Department of Computer Science and Engineering

# Interaction and Visualization Methods in Teaching Spatial Algorithms and Analyzing Spatial Data

Jussi Nikander

Aalto University

DOCTORAL DISSERTATIONS

Aalto University publication series
**DOCTORAL DISSERTATIONS** 134/2012

# Interaction and Visualization Methods in Teaching Spatial Algorithms and Analyzing Spatial Data

**Jussi Nikander**

Doctoral dissertation for the degree of Doctor of Science in Technology to be presented with the due permission of the School of Science for public examination and debate in auditorium T2 at the Aalto University School of Science (Espoo, Finland) on the 9th of November, 2012, at 12 noon.

**Aalto University**
**School of Science**
**Department of Computer Science and Engineering**

**Supervising professor**
Professor Lauri Malmi

**Thesis advisor**
Professor Kirsi Virrantaus

**Preliminary examiners**
Professor Clifford A. Schaffer, Virginia Tech., USA
Professor Jason Dykes, City University London, UK

**Opponent**
Dr. Anthony Robinson, Pennsylvania State University, USA

Aalto University publication series
**DOCTORAL DISSERTATIONS** 134/2012



ISBN 978-952-60-4826-0 (printed)
ISBN 978-952-60-4827-7 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-60-4827-7

Unigrafia Oy
Helsinki 2012

Finland

Publication orders (printed book):
This dissertation can be read at
http://otalib.aalto.fi/en/collections/e-publications/dissertations/

NORDIC ECOLABEL
441 697
Printed matter

Aalto University

# Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

**Author**
Jussi Nikander

**Name of the doctoral dissertation**
Interaction and Visualization Methods in Teaching Spatial Algorithms and Analyzing Spatial Data

**Publisher** School of Science

**Unit** Department of Computer Science and Engineering

**Series** Aalto University publication series DOCTORAL DISSERTATIONS 134/2012

**Field of research** Software systems

**Manuscript submitted** 17 April 2012 **Date of the defence** 9 November 2012

**Permission to publish granted (date)** 14 September 2012 **Language** English

☐ **Monograph** ☒ **Article dissertation (summary + original articles)**

**Abstract**

This thesis investigates the benefits of using interactive techniques and visualization in GIS. The thesis consists of two studies: in the first study the topic is teaching spatial data algorithms, and in the second performing spatial analysis.

In the first study, the main research question was how software visualization can be used to help geoinformatics students learn spatial data structures and algorithms. In order to test this question, the TRAKLA2 learning environment for data structures and algorithms was expanded to include spatial data algorithms. Using the system, the students practice how spatial algorithms and data structures work by simulating the modifications an algorithm does to a data structure.

The system has been used in an actual spatial data algorithms course, and its effect on the students' learning results has been evaluated using both quantitative and qualitative methods. Quantitative methods were used to compare the students' learning results in the system to their results in exams; qualitative methods were used for content analysis and analyzing student interviews using semi--structured interview method. The results gained indicate that visual learning environments can be used to help teaching spatial data algorithms.

The topic of the second study is exploratory methods, their use in solving suitability problems, and the new information that can be gained by using them. In a suitability problem the goal is to categorize locations based on how well they fit the criteria for some activity. To test these questions a prototype for a flexible spatial analysis was implemented. The prototype is based on the ideas of exploratory analysis and visual analytics, and contains simple analysis tools.

The developed system has been used to analyse the off-road mobility of vehicles using an armored personnel carrier as the example vehicle. The results have been evaluated by comparing them to an existing mobility model created by Finnish Defense Forces, and by expert evaluation of the analysis results. The results indicate that exploratory methods work just as well as traditional model-based approaches.

**Keywords** education, interaction, information visualization, software visualization, spatial analysis, spatial data

**ISBN (printed)** 978-952-60-4826-0 **ISBN (pdf)** 978-952-60-4827-7

**ISSN-L** 1799-4934 **ISSN (printed)** 1799-4934 **ISSN (pdf)** 1799-4942

**Location of publisher** Espoo **Location of printing** Helsinki **Year** 2012

**Pages** 255 **urn** http://urn.fi/URN:ISBN:978-952-60-4827-7

Aalto-yliopisto

**Tiivistelmä**

**Aalto-yliopisto, PL 11000, 00076 Aalto** www.aalto.fi

**Tekijä**
Jussi Nikander

**Väitöskirjan nimi**
Vuorovaikutus ja havainnollistaminen sijaintitietoalgoritmien opetuksessa ja paikkatiedon analyysissä

**Julkaisija** Perustieteiden korkeakoulu

**Yksikkö** Tietotekniikan laitos

**Sarja** Aalto University publication series DOCTORAL DISSERTATIONS 134/2012

**Tutkimusala** Ohjelmistojärjestelmät

**Käsikirjoituksen pvm** 17.04.2012 **Väitöspäivä** 09.11.2012

**Julkaisuluvan myöntämispäivä** 14.09.2012 **Kieli** Englanti

☐ **Monografia** ☒ **Yhdistelmäväitöskirja (yhteenveto-osa + erillisartikkelit)**

**Tiivistelmä**

Väitöskirja käsittelee vuorovaikutteisten menetelmien ja havainnollistamisen käyttöä geografisen datan käsittelyssä. Väitöskirja koostuu kahdesta tutkimuksesta, joista ensimmäisen aihe on sijaintitietorakenteiden opetus, ja toisen aihe paikkatietoanalyysi.

Ensimmäisen tutkimuksen aihe oli ohjelmistojen havainnollistamisen käyttö auttaa sijaintitietorakenteiden ja -algoritmien opetuksessa. Kysymystä tutkittiin tuottamalla TRAKLA2 oppimisympäristöön sijaintitietorakenteita käsittelevä laajennus. Ympäristön avulla opiskelijat saattoivat harjoitella algoritmien toimintaa simuloimalla algoritmin tietorakenteeseen tuottamia muutoksia.

Ympäristöä käytettiin opetuksessa ja sen vaikutus opiskelijoiden oppimistuloksiin arvioitiin sekä määrällisiä että laadullisia menetelmiä käyttäen. Määrällisenä menetelmänä oli opiskelijoiden oppimisympäristössä saamien tulosten tilastollinen vertaaminen heidän tenttituloksiinsa; laadullisena menetelmänä olivat sisältöanalyysi ja puolistrukturoidut haastattelut. Tutkimuksen mukaan vuorovaikutteisista oppimisympäristöistä on hyötyä sijaintitietorakenteiden opetuksen tukena.

Toisessa tutkimuksessa aiheena oli tutkivan havainnollistamisen käyttö paikan soveltuvuusanalyysin tekemiseen. Tutkimuksessa toteutettiin ja testattiin analyysityökalun prototyyppi. Prototyyppi sisälsi vuorovaikutteisten havainnollistamismenetelmien lisäksi yksinkertaisia laskennallisen analyysin työkaluja.

Prototyyppiä käytettiin ajoneuvojen maastoliikkuvuuden tutkimiseen, käyttäen esimerkkiajoneuvona panssaroitua miehistönkuljetusajoneuvoa. Tuloksia analysoitiin vertaamalla niitä puolustusvoimien olemassaolevaan maastoliikkuvuusmalliin, sekä käyttämällä asiantuntija-arvioita. Tutkimuksen mukaan tutkivan havainnollistamisen menetelmillä voidaan saada tuloksia, jotka ovat vertailukelpoisia perinteisen mallipohjaisen analyysin tuloksiin.

**Avainsanat** Opetus, vuorovaikutus, tiedon havainnollistaminen, ohjelmistojen havainnollistaminen, paikkatieto, paikkatietoanalyysi

**ISBN (painettu)** 978-952-60-4826-0 **ISBN (pdf)** 978-952-60-4827-7

**ISSN-L** 1799-4934 **ISSN (painettu)** 1799-4934 **ISSN (pdf)** 1799-4942

**Julkaisupaikka** Espoo **Painopaikka** Helsinki **Vuosi** 2012

**Sivumäärä** 255 **urn** http://urn.fi/URN:ISBN:978-952-60-4827-7

# Preface

The journey that led me here really started when Professor Lauri Malmi hired me as a teaching assistant for the basic data structures course at Helsinki University of Techonlogy. I was, at that time, a second-year student at the Computer Science department. That way I got to join his research group at the university and participate in the development of the TRAKLA2 system. During that time Lauri, and docent Ari Korhonen helped and instructed me a great deal. I must especially thank Ari for his habit of questioning everything - it forced me to develop my abilities for critical thinking and justifying my arguments. After finishing my master's thesis, the focus of my work changed, when Professor Kirsi Virrantaus and her geoinformatics research group stepped into the picture. If not for her, I doubt I would ever had done anything involving spatial data or maps. If not for these three people, I probably would not be here. Thank you for your support and guidance.

There are also other colleagues who deserve a mention. Juha Helminen and Tanja Kantola coded more lines to the TRAKLA2 spatial extension and the Analysis Prototype respectively than I did. Petri Ihantola, Ville Karavirta, and Otto Seppälä helped me in many ways during the years with TRAKLA2 and computer science education research. Eiri Valanto helped me a lot when I first familiarized myself with spatial data algorithms and designed the spatial extension to TRAKLA2, and Paula Ahonen-Rainio helped me in various ways during my time at the geoinformatics research group. Thank you all.

I would also like to thank the pre-examiners, professor Clifford A. Schafford and professor Jason Dykes, for their comments, which helped me to improve this thesis considerably. I also thank Dr. Anthony Robinson for acting as my opponent.

Finally, there is of course my family. Back when I was a young boy I

got to see how my father tried to finish his own dissertation and, I knew my brother was working on his. Both defended their dissertations long ago. Because of their example, it was only natural for me to try become a Doctor of Science. Special thanks go to my wife Satu, who spent many hours poring through my work and fixing numerous errors in my English. And finally there is my daughter, whose mere existence has brought so my joy into my life. My deepest thanks for you all.

Espoo, October 10, 2012,

Jussi Nikander

# List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.

**I** Jussi Nikander, Ari Korhonen, Eiri Valanto, and Kirsi Virrantaus. Visualization of Spatial Data Structures on Different Levels of Abstraction. *Electronic Notes in Theoretical Computer Science*, Vol 178: the Proceedings of the Fourth Program Visualization Workshop (PVW 2006), pp. 89–99, July 2006.

**II** Jussi Nikander and Juha Helminen. Algorithm Visualization in Teaching Spatial Data Algorithms. In *11th International Conference Information Visualization IV2007*, Zürich, Switzerland, July 2007.

**III** Jussi Nikander, Juha Helminen, and Ari Korhonen. Experiences on Using TRAKLA2 to Teach Spatial Data Algorithms. *Electronic Notes in Theoretical Computer Science*, Vol 224: the Proceedings of the Fifth Program Visualization Workshop (PVW 2008), pp. 77-88, July 2008.

**IV** Jussi Nikander, Juha Helminen, and Ari Korhonen. Algorithm Visualization System for Teaching Spatial Data Algorithms. *Journal of Information Technology Education*, vol 9, pp. 201-225, 2010.

**V** Jussi Nikander and Kirsi Virrantaus. A New Exploratory White Box Method for Spatial Data Analysis. In *Cartography and Geoinformatics for Early Warning and Emergency Management: Towards Better Solutions*, Prague, Czech Republic, January 2009.

**VI** Jussi Nikander, Tanja Kantola, and Kirsi Virrantaus. Exploratory vs. Model-Based Mobility Analysis. Accepted for publication in *Nordic Journal of Surveying and Real Estate Research*, August 2012.

# Author's Contribution

### Publication I: "Visualization of Spatial Data Structures on Different Levels of Abstraction"

This paper introduces the theory of algorithm visualization abstraction levels. The paper also describes how the Matrix framework and the TRAKLA2 algorithm visualization environment based on it can be extended to the visualization of spatial data structures and algorithms.

All authors contributed evenly to this paper.

### Publication II: "Algorithm Visualization in Teaching Spatial Data Algorithms"

This paper discusses the spatial data algorithm extension to the TRAKLA2 system. The paper covers how the spatial extension has been implemented, and introduces both the modifications done to the Matrix framework, and the spatial exercises. The paper also includes first impressions gained by using the system in teaching.

I was the lead author of this paper and the leading developer of the system described in the paper. My co-author assisted in the system implementation and in writing the paper.

### Publication III: "Experiences on Using TRAKLA2 to Teach Spatial Data Algorithms"

This paper describes the first results of using the spatial data algorithm extension to the TRAKLA2 system. The system and the results are discussed from three different points of view: the developer's, the instruc-

tor's, and the student's. Using the different points of view we can examine the system implementation and its effect on teaching and learning from a broader perspective.

I was the lead author of this paper and the main developer of the system described in the paper, while my co-authors assisted in the system development. I did the quantitative analysis of the results while my co-authors were responsible for gathering the qualitative analysis. All authors contributed evenly to the interpretation of the results.

### Publication IV: "Algorithm Visualization System for Teaching Spatial Data Algorithms"

This paper brings together elements from the first two papers as well as expands the analysis of the learning results from the third paper. The focus is on the students' attitudes and learning results gained using the TRAKLA2 system.

I was the lead author of this paper and responsible for quantitative analysis and many parts of the qualitative analysis described in this paper. My co-authors assisted in the analysis and in analysis interpretation.

### Publication V: "A New Exploratory White Box Method for Spatial Data Analysis"

This paper describes the initial ideas and theoretical background we had for exploratory and visual spatial data analysis methods. It also introduces the model for the spatial analysis process described in this thesis.

The work in this paper was evenly distributed by the two authors.

### Publication VI: "Exploratory vs. Model-Based Mobility Analysis"

This paper describes the theoretical model for the exploratory and visual spatial data analysis process. It also introduces the prototype application we implemented, and the results of applying the prototype to an actual data analysis process.

I was the lead author of this paper. I was responsible for the analysis process model described in the paper. I was the leading developer of the prototype described here. One of my co-authors assisted in the prototype

development. I was responsible for analyzing and interpreting the results.

# Contents

# 1. Introduction

This thesis consists of two multidisciplinary studies that apply computer science methodology and techniques to solving problems in geoinformatics. In both studies the aim of the research is to examine the use of interaction and visualization methods in geoinformatics. The two topics of the research are teaching spatial data algorithms and developing spatial analysis methods. The two topics may seem, at least at first, to be almost completely unrelated. Both learning and analysis are, however, interactive and often iterative processes, where the user – learner or analyst – tries to test their mental model of a situation by using given external data. These similarities between the processes give rise to the possibility that we can gain new information by comparing the processes.

The goal of teaching data structures and algorithms is to have the students learn how these constructs work on a given level of abstraction. The levels of abstraction vary. The lowest level is knowing how to implement a data structure or an algorithm on a specific programming language. Higher levels include conceptual knowledge of different problem solving strategies that can be used to design data structures and algorithms. This study concentrates on the conceptual knowledge level.

Spatial algorithms are algorithms designed to store and manipulate coordinate data and spatial relationships, such as topology. As such they tend to be conceptually more complicated and harder to grasp than regular data structures and algorithms. Thus, teaching spatial data algorithms is a challenging topic. Results of the work done on the development of spatial data algorithm teaching at Aalto University have been detailed in [84].

Spatial analysis is a process where coordinate data, such as maps of different types, are analyzed using various methods in order to solve a problem or help decision making. The data analysis example in this research

is off–road mobility analysis. In off-road mobility analysis, the terrain is categorized to different suitability categories according to how easy the movement outside roads is for a specific vehicle type.

## 1.1 Background and Motivation

The original idea that led to this research was the desire to improve the teaching of spatial data algorithms at Aalto University (previously known as Helsinki University of Technology). This topic was taught at the Department of Surveying in a single course aimed at third year geoinformatics students. The teacher in charge of the course was interested in adding interactive web–based exercises to the course. The idea was to use visualizations, since geoinformatics uses a lot of illustrations in the form of maps and figures, and therefore the students were already familiar with visualizations.

Thus, the initial idea was to add exercises that use *software visualization* to the spatial data algorithms course and investigate how they affect the students' learning results. This aim required two topics of research. The first topic of research was whether it is possible to create computerized, visual exercises on spatial data algorithms. There had been previous work at creating spatial data visualizations and some individual exercises, but no comprehensive exercise sets existed. The second topic of research was the effect such exercises would have on learning.

Later, the research was expanded to include the use of visualization in spatial analysis. This broadened the research from software visualization to the more general category of *information visualization*. The spatial analysis problem selected for the project came from our research partners in the Finnish Defense Forces. The Defense Forces had developed terrain analysis methods for analyzing mobility, fortifiability, and other topics relevant to defense [86]. The original work by Defense Forces concentrated on traditional mathematical models that could be used automatically. In order to use a given model, the user gathered the required input data, preprocessed it into a specific format, and inserted the processed data into the model. As output, the user would get a mobility map for the area depicted by the input data. However, these models are valid only with a given, strictly–defined input, and only for Finnish terrain. Thus, the goal of this work was to investigate if there were ways to create more flexible analysis methods that employed visualizations and interactive methods.

The method selected was exploratory analysis techniques.

## 1.2 Fields of Science in This Study

This work is a multidisciplinary study, and as such several fields of science are relevant to this work. This section briefly introduces the different elements used. The study draws mainly from *computer science* and *geoinformatics*. Elements from *information visualization* are also an important part of the thesis.

### 1.2.1 Computer Science

*Computer science* is the study of the theory and application of computational problem solving, and related disciplines. The field contains numerous topics from the study of computational complexity, i.e. how much resources solving a given problem takes, to the design of programming languages. This work draws mainly on two fields of computer science. The first field, *algorithmics*, is the design and study of algorithms. An algorithm is a finite number of precise steps that solve a well-defined problem, or prove that no solution is possible. Practically all computerized problem solving is based on algorithms. The second field, *software visualization*, is a branch of software engineering focused on the illustration of software. Software visualization seeks to create useful and understandable visualizations that help the viewer understand certain aspects of the software. These visualizations can be pictures, picture series, or animations. A visualization may allow the user to interact with it in order to make it easier to focus on interesting aspects of the visualization.

### 1.2.2 Geoinformatics

*Geoinformatics* can be defined as being any aspect of the capture, storage, integration, management, retrieval, display, analysis, and modeling of spatial data [120]. Geoinformatics can be approached from many points of view. In this work, two approaches are considered. The first approach is to consider GIS as the application of different methods in order to accomplish an objective. From this point of view, how the methods work is not interesting, and the user is more focused on what can be accomplished with the tools at hand. Thus, geoinformatics can be seen as a way to solve spatial problems using available tools and frameworks. The second

approach is to consider GIS as the application of different computerized methods for solving spatial problems in novel ways. From this point of view, the details of the methods are a point of interest. These methods are typically applied for creating new solutions to existing problems or for solving newly discovered problems. From this point of view, geoinformatics is the application of computer science methods for solving spatial problems in novel ways.

Most institutions teaching geoinformatics lean towards the first point of view. Students are taught how to use the various existing tools and methods, such as desktop GIS environments and map algebra, for solving geoinformatics problems. The underlying data structures and algorithms used are typically taught only on a very basic level. At Aalto University, however, *spatial data structures and algorithms* are an important part of the curriculum.

### 1.2.3 Information Visualization

*Information visualization* is a field of science that studies how complex large–scale collections of information can be visualized in an understandable manner [113]. The goal is to create visualizations from which the viewer can gain additional value and information. Information visualization techniques are used to illustrate, for example, organization structures, network relations, databases, or program code. In the broadest sense, any illustration of complex data or information can be viewed as information visualization.

One application of information visualization is *multivariate data visualization*, where the goal is to comprehensibly visualize n-dimensional data elements. One multivariate data visualization method, the parallel coordinates plot (PCP) [52], is used in this work. In a PCP visualization the data dimensions are depicted as parallel lines and data elements are visualized as polylines with vertices on the parallel lines. An example of a parallel coordinates plot can be seen in Figure 1.1, where a PCP view is shown together with a map view representing the same data. The figure also shows another often-employed visualization method: the use of multiple, simultaneous linked views. Both views in Figure 1.1 show the same set of data using different visualizations. The views are linked together: if a modification such as highlighting some data items is done in one of the views, this is also reflected in the other view.

### 1.2.4 Visualization in Geoinformatics

The most common visualizations in geoinformatics are different types of maps. Most people are familiar with common maps, which are often used in everyday life. In geoinformatics many types of thematic maps are also used. A thematic map is used to emphasize the spatial pattern of one or more geographic attributes [108]. A thematic map can show, for example, population density, annual rainfall, or different soil types. There are numerous different types of thematic maps, such as choropleth maps, dot maps, and dasymmetric maps. In a choropleth map, for example, the attribute value is shown separately for each measurement region, and different values are differentiated using, for example, different shades or colors. The regions are selected by the map designer and can reflect, for example, municipalities.

Often, however, using only map visualizations is not sufficient. For example, if there are several independent attribute data variables, it becomes hard to include all of them in one map. Thus, when the number of interesting variables grows, multivariate visualization methods are required. Information visualization views combined with map visualizations allow the user to gain insight on the data that is hard to grasp from just map views. Techniques used include bar charts, parallel coordinates plots, Chernoff faces [22], and star diagrams [21]. Some of these can be overlaid on the map, while others need a separate view. An example of two linked views can be seen in Figure 1.1, where spatial data has been clustered using three input data layers. In the figure, the clusters are shown both using a map view and a parallel coordinates plot view. In both views one cluster has been highlighted using magenta color. The map view on left side of the image shows the spatial distribution of each cluster, and the right side shows the data value distribution of the cluster using parallel coordinates plot.

### 1.2.5 Spatial Data and Spatial Data Algorithms

*Spatial data* is data that is located in a multi-dimensional space [68]. In geoinformatics, spatial data typically models geographic locations on Earth's surface. Such data has either two or three spatial dimensions – x– and y–coordinates, and possibly elevation – and may also have time as an additional dimension. Thus, each data item contains *spatial information* that defines the location of the data item in space, and possibly in

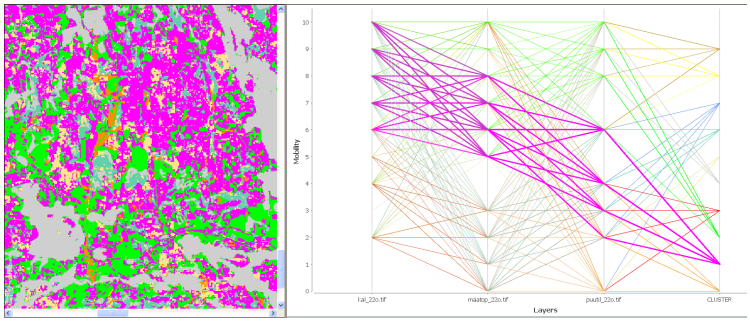


**Figure 1.1.** Simultaneous map and multivariate visualization view of the same data. The data represents a clustering result for three attribute dimensions. The map shows the geographic extent of each cluster, while the PCP view shows the attribute extent of each cluster.

time, and associated *attribute data* that defines what the data item represents. Spatial data is manipulated using both regular non-spatial data structures and algorithms, and specialized *spatial data structures and algorithms* that have been designed to take the multidimensional nature of the data into account.

Spatial data structures are based on non-spatial structures, such as arrays or trees, and algorithms for manipulating these structures. However, taking the multidimensional nature of the data into account makes the design and implementation of efficient spatial structures a complex task. For example, multidimensional data does not have any one unambiguous order. Thus, in order to linearize spatial data, methods such as space-filling curves or spatial location code, are needed [118].

### 1.2.6 Spatial Analysis

Spatial analysis is the formal study of spatial data. The term is not very well defined and may have widely different meanings in different contexts. O'Sullivan and Unwin [87], for example, use the term spatial analysis in four different contexts: *spatial data manipulation*, *spatial data analysis*, *spatial statistical analysis*, and *spatial modeling*. However, they also mention that it is typically hard to distinguish between the different approaches, and in many situations all four are required. Furthermore, the data analysis, statistical analysis, and modeling definitions are closely related. In general, spatial analysis is the investigation of spatial data (using the definition given in section 1.2.5) using various qualitative and quantitative methods. The aim is to investigate a phenomenon or solve a problem.

There are numerous different methods for spatial analysis. The best methods for any given situation depend on the problem at hand and the data used in the analysis. For example, there are *exploratory* methods that are used to investigate the datasets interactively in order to gain further knowledge of their qualities. There are also *statistical* methods that calculate statistics of the data, taking into account their spatial characteristics. These statistics can be either global, where the statistics represent some features of the whole data set, or local, where the statistics vary spatially across the data.

When doing spatial analysis, the analyst needs to select the methods used depending on the situation, the available data, the spatial phenomena being studied, and the desired solution or results. If the analyst uses exploratory, visual approaches, then they are constantly interacting with various visualizations. These visualizations can potentially show the analyst tremendous amounts of information, which in turn can make the visualizations completely incomprehensible. Thus, in most visual analysis the basic approach taken can be summarized using the Visual Information Seeking Mantra [103]: "Overview first, zoom and filter, then details-on-demand".

### 1.2.7 Software Visualization

Software Visualization (SV) is a branch of software engineering, which aims to use graphics and animation to illustrate the different aspects of software [110]. In [94] SV is divided into two subcategories: program visualization (PV) and algorithm visualization (AV). PV is the use of various visual techniques to enhance the human understanding of computer programs. It is typically used to illustrate actual, implemented programs. AV, on the other hand, illustrates abstractions of algorithmic concepts and is often independent of any actual algorithm implementation. Of the two subcategories, only algorithm visualization is relevant for this thesis.

Education is one of the primary applications of SV – numerous SV systems have been developed for teaching purposes. It has been noted in [49], that merely introducing SV to teaching does not seem to improve learning results. In order to benefit from the use of SV, the learner must become an active participant in the learning process by interacting with the visualizations. They must, for example, construct an animation or simulate the work flow of a data structure. Thus, modern educational SV systems typically offer high level of interaction for the learner. One way to do this

is to use exercises where interaction with the algorithm visualizations is required for solving the problem. For example, the learner might manipulate data structures through graphical user interface in order to simulate the modifications a real algorithm might do [73]. Another possible option is to create a system, where the user needs to construct algorithm animations, and thus explore how the algorithm works [48].

## 1.3 Interaction and Visualization as Part of a Process

Both spatial analysis and learning processes are *interactive* and often *iterative*. During the process the user (either analyst or learner) interacts with the data, typically using various visualizations. Furthermore, both the analysis process and the learning process can result in an untenable situation. In analysis the result might not contain the information required, or in learning the learner's understanding of the topic can turn out to be unusable. Thus, both processes may require the user to return to an earlier phase of the process and start again from there, trying something different. Thus both processes can be – and often are – iterative.

### 1.3.1 Teaching Process

This work is based on the idea that learning happens using a *mental model* [12], which is a person's internal understanding of a given topic. A mental model is viable, if it can be used to correctly solve problems related to the topic at hand. The model can be refined using many methods. This work discusses exercises, where the learner needs to apply their mental model. If the model is not viable, the learner will encounter situations where their mental model leads to incorrect answers. These *conflicts* then cause the learner to refine their mental model, and through refinement the model can become viable.

The refinement of mental models requires interaction with the topic, and the process is often iterative in nature. After a conflict has been discovered, the learner must try to solve it. One way to do this is to find the point where the mistake was made, and try and solve the problem again from this point, fixing the mistake. Such interactive and iterative process is possible in a classroom. There, the students can interact with the course staff and get feedback. Outside the classroom setting other methods for gaining feedback need to be devised.

One way to gain feedback regardless of the time or location is to use exercises that can be *automatically assessed*. This way, the solutions the learners submit for exercises can be checked without involvement from the course staff. Furthermore, if the automatic assessment system is connected to the Internet, the learners can return their solutions anywhere and at any time, and thus do not need to come to the classroom at a given time in order to submit their solutions.

For teaching data structures and algorithms, automatic assessment can be combined with software visualization techniques in order to make it possible for the learner to interact with the data structures relevant to a given problem. In order to be effective, the software visualizations used should be interactive and make the learner an active participant in the learning process [49]. One way to do this is to use *visual algorithm simulation* [60], where the learner simulates the behaviour of an algorithm by manipulating a set of data structure visualizations. Using such exercises the learner can see whether their mental model of a given algorithm is viable, and try to refine the model when conflicts are encountered. The exercises can introduce conflicts either immediately when the learner makes a mistake, by giving feedback after each simulation step, or after the learner has completed the exercise and submitted it for assessment. Then, after discovering the cause for the conflict and trying to improve the mental model, the learner can try the exercise again in order to see whether the refined mental model is viable. This way both interaction and iteration can be used in a learning process.

### 1.3.2 Spatial Analysis Process

The idea of exploratory spatial analysis [9] exemplifies both interactivity and iterativity. The process is also called exploratory spatial data analysis, or exploratory data analysis, when the spatial nature of the data is not emphasized. Here the term "exploratory analysis" will be used. The idea of exploratory analysis is that the analyst is interacting with the data by using various visualizations and tries to find new information and insights by exploring the various aspects of the data. Typically the main goal of exploratory analysis is the formulation of hypotheses that can then be tested using other methods, such as spatial statistics.

Exploratory analysis has influenced the creation of other data analysis methodologies. One such related methodology is *visual analytics*, which is a method of analytical reasoning facilitated by interactive visualiza-

tions [114]. The idea of visual analytics is that the analyst interacts with, and explores, data sets using interactive visual interfaces in order to gain deeper insight. This newly created understanding can then be used to advance the problem solving process. Visual analytics in spatial analysis is closely related to exploratory analysis. In both cases the analyst explores the data using visualizations. In exploratory analysis the emphasis is on exploration, and in visual analytics it is on the visualizations.

Since spatial datasets typically have two or three spatial dimensions and a number of attribute data dimensions, multivariate visualization methods are often required in order to be able to interpret the data in any meaningful manner. Furthermore, in many cases it must be possible to control the amount and type of data being visualized in order to be able to comprehend visualizations. This ties back to the information visualization mantra [103].

Without sufficiently informative visualizations, it is not possible to see patterns in a dataset that has numerous data elements in many dimensions. Furthermore, it has been shown that while many variables can be visualized on the same map, such maps quickly become hard for the viewer to understand [108]. Thus, when multiple attribute data values need to be shown simultaneously, methods such as interactivity or multivariate visualizations are required in addition to maps.

A spatial analysis process can also alternate between exploratory phases, where the analyst uses various means to interact with the data in order to gain new insight, and confirmatory phases, where other spatial analysis methods are used to test these insights. Exploratory analysis can be defined to include only the use of interaction and visualizations. In this work, however, it is assumed that exploratory phases can also contain the use of computational methods. For example, several data sets can be combined using map algebra [115].

Visualizations are not only required in the exploratory phase, but also in the interpretation of the computational spatial analysis methods. Most spatial analysis methods give a spatially varying output. Examples of spatially varying methods are the kernel density function, the various different variations of Voronoi–diagram, spatial interpolation methods, and numerous different field model calculations. Thus, maps are required in order to visualize the results of the analysis for further interpretation. Therefore, visualization is present in spatial analysis from the very beginning to the end.

For example, the kernel density function is a method for analyzing point data density. It creates a field representation from point data, where the field value at each location is dependent on the number of data points that are closer than given distance $r$, which is often called the *bandwidth* of the kernel estimation. In a simple, naive kernel density, the number of points inside the bandwidth is calculated, and each point is given the same weight. In a more complex analysis the distance to the point is taken into account using a kernel function. When using a kernel function, the more distant points are given less weight than points that are close by. [87].

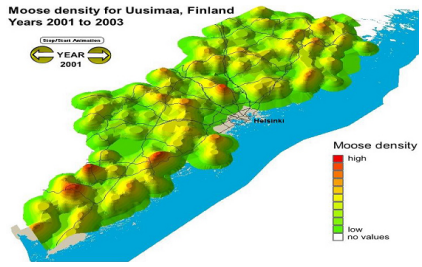


**Figure 1.2.** Kernel Density Function. Originally presented in [65], used with permission of the original author

Figure 1.2 shows an example of kernel density function analysis. Kernel density is typically used to calculate the density of data points, and it has several applications. At Aalto University, for example, kernel density has been used to estimate crime and rescue operation densities [119], space-time trajectories [31], and moose behavior and density [65].

There are also methods, like G– and K–functions, or the misclassification matrix, that give non–spatial output. In such situations it is possible to draw some conclusions about a dataset without using map visualizations. The output of a G– or K–function is a two–dimensional line chart, where the value of the function is given depending on the distance. The G–function, for example, measures the shortest neighbor distances between elements in a point set. The value of a function at a distance $d$ is the fraction of the points in the set for which the distance to the nearest neighbor is less than, or equal to, $d$.

To summarize, spatial analysis can be considered to consist of two phases: an exploratory phase where the analyst familiarizes themselves with the data, and a confirmatory phase where insights gained during the exploratory phase are tested. Furthermore, in practice these phases can be hard to separate from each other due to the interactive and iterative

nature of the analysis process.

## 1.4 Research Problems

The focus of this research are the two following processes: the process of learning spatial data structures and algorithms, and the process of analyzing spatial data. The interaction and visualization methods employed by the two processes are the main topics of investigation in this work. For the questions "what is the best visualization", and, "what is the best interaction method", there are unlikely to be any general answers. The information communicated by a visualization, and the extent and type of interaction required, is task–specific, and also depends on the data. Furthermore, for a given situation, different people have different opinions about the best visualization or interaction method.

The research is constructive in nature, and in both processes existing visualization and interaction methods are used. The goal is to investigate how these methods can be used in novel ways to solve problems. Thus, the basis of the research is a literature review of existing methods. Some of the methods are then selected and included in a prototype. The prototype is tested, and the data gathered is analyzed to measure the usefulness of the prototype.

The two processes are different, but they share several similarities. Thus, a comparison of the processes was done as a part of the research. There are two research questions concering the comparison: *Are there similarities between the learning and the analysis process?* and *Can either of these processes be enhanced by using knowledge gained from comparing the two processes*?

### 1.4.1 Teaching SDA

The initial motivation for this research was improving the teaching of spatial data structures and algorithms at Aalto University by using software visualization. Software visualization was decided on because of the intrinsically visual nature of geoinformatics. Students are already familiar with visualizations such as shown in Figure 1.1. Therefore algorithm visualization was hypothesized to be a good learning tool for teaching SDA. The TRAKLA2 system [73], originally created for teaching basic data structures and algorithms, was thus expanded to cover spatial data.

The main research problem in this part of the work is *How can software visualization be used to help geoinformatics students to learn spatial data structures and algorithms*? Most of the research on educational use of algorithm visualization has focused on basic data structures and algorithms. This research examines the use of visualization in teaching a completely different area of algorithmics.

Geoinformatics is an intrinsically visual field, and therefore the students can be assumed to be familiar with the concept of visualization. However, most visualizations they are familiar with are cartographic visualizations or information visualizations. Furthermore, the students cannot be assumed to have the same level of background knowledge in computer science as CS majors, and many of the data structures and algorithms discussed on the course are quite complicated. Therefore, it is not certain what kinds of visualizations the algorithms require, and whether the visualizations affect learning or not.

This research problem can be divided into a number of smaller problems.

1. *How can spatial algorithms be visualized*? Most software visualization techniques are focused on showing the details of the data structures. In such visualizations data elements are typically shown using key values that identify each data element. Key values are one-dimensional.

   The data used in geoinformatics is spatial, and therefore simple key value visualizations are inadequate for conveying how different data items relate to one another. However, merely drawing two-dimensional data elements to the 2D plane makes it hard to grasp how the elements are stored in the data structures. Therefore, more complex solutions are required. This question can then be divided further into smaller subquestions:

   (a) *What kinds of algorithm visualization elements are available for spatial visualization*? Spatial data has always at least two dimensions, the x- and y-coordinates. Thus, traditional one-dimensional key value visualizations are not sufficient, and *multidimensional* data visualization views are required. Furthermore, these views should be such that they can convey how spatial data elements relate to one other. Thus, not all multidimensional visualizations are appropriate for visualizing spatial data, and specific data views designed for illustrating spatial data are required.

(b) *How does spatial data affect algorithm visualization design*? After spatial data visualization methods have been reviewed, the next step is to use them to design algorithm visualizations that illustrate how spatial data structures and algorithms work. Most algorithm visualizations are focused on visualizing basic data structures and algorithms containing one-dimensional data elements. Spatial data structures and algorithms, which handle multidimensional data, may require different design principles.

(c) *How can comprehensible spatial algorithm simulations be implemented using TRAKLA2*? Prior to this research, the TRAKLA2 system did not have any semantics for multidimensional data items, and no means of presenting multidimensional views. The visualization of two-dimensional data is more complicated than one-dimensional data. The relationships between items are more complicated, and a computer screen only has two physical dimensions. Such complications make the implementation of useful visual algorithm simulation exercises a challenging task.

2. *Do the implemented TRAKLA2 spatial exercises promote learning* ? Merely implementing an algorithm visualization system for multidimensional data is not sufficient. If the implemented system does not help students to learn, either by affecting the learning results or by improving the student attitudes, there is no clear advantage in using it.

   Therefore, in this research, the aim is evaluating the effect of TRAKLA2 spatial visualizations on learning. Both the students' learning results and attitudes are examined. Thus, this research question can be divided into two subquestions: *Are there quantitative effects* and *are there qualitative effects*.

### 1.4.2 Spatial Analysis

Spatial analysis is the analysis of spatially varying phenomena with the aim of producing new information. There are numerous different tasks where spatial analysis can be applied, for example decision support and planning. Perhaps the most well-known spatial analysis method involving multiple map layers is the map overlay. In a map overlay several input maps are added together into one output map that contains a combination

of the input map information. A very simple map overlay method is sieve mapping, where the goal is to find areas suitable for a given task. Initially the whole area is considered suitable. Then, a number of input layers containing criteria that make some area unsuitable are laid over each other. In the output map, only areas that have no disqualifying criteria are considered suitable. [87]

Sieve mapping is an example of suitability analysis, which is the process of determining the feasibility of a given area for a given activity. Suitability analysis can answer various questions, for example, how safe or risky a certain area is, or what are the best locations for field hospitals, communication links, command posts, or helicopter landing areas. The output of a suitability analysis is a map that splits the given area into suitability categories according to the characteristics of each point. The map visualization can also be combined with other information visualization views that give different perspectives to the results of the analysis and thus enable the analyst to, for example, see why a certain point has a certain suitability value.

The research group on Geoinformatics at the Department of Surveying at Aalto University, together with the Finnish Defense Forces, has developed several methods and tools for terrain analysis [53, 104, 117]. At Aalto, the tools typically reach a prototype stage, and further software development is handled by the Defense Forces. The collaboration with Finnish Defense Forces has continued in this work.

One topic of research in suitability analysis is troop *mobility* in the field. The mobility of troops depends on several factors, including the troop type, the season, the soil type, the type and coverage of vegetation, roads, and buildings. Traditionally the analysis is done using a mathematical model that has been developed for the purpose. The model gives each pixel of the area a mobility value.

The problem of verified models is that they only work with specific input data. Thus, if one or more of the input data sets are missing, or the contents of a data set differ from what is assumed, the model is not usable. For example, the soil type input data is assumed to conform to the soil type classification used by the defense forces. This classification is designed for Finnish terrain, and thus may not be usable in other countries. Similarly, the vegetation layer assumes that the amount of vegetation is given using cubic meters of wood per hectare. If some other measure for the amount of vegetation is used, or the vegetation is given as different

vegetation types, the model cannot use the input. Thus, since the model is developed for Finland, it cannot be used in places where the circumstances are radically different from Finnish terrain.

Therefore, there is a need for flexible and general analysis methods for situations where traditional models can not be used. For example, in crisis management situations the analyst often needs to find a solution using whatever data is available. There are no guarantees about the quality, type, or contents of the available data, and thus explicit models are likely to be unusable. Using exploratory techniques, analysis can be attempted with whatever data happens to be available. Thus exploratory analysis may work in situations where traditional models are unusable. Suitable methods and visualizations depend on the situation and the data available. Possible analysis methods include the use of map algebra [115], or data mining methods such as clustering.

In clustering, the data is divided into several different categories (clusters) based on the similarity of the data points. Possible ways to calculate the similarity of data items are distance between items or the density of data items in the data space. Perhaps the most well–known clustering method is the distance–based $k$-means method [71], but numerous others have also been developed. In this work, k-means and the density-based *DBSCAN* [33] are used.

The main research question of this part of the work is *How can exploratory analysis methods be used to enhance suitability analysis*? The model-based suitability analysis method has several limitations that prevent its usage in many situations. Therefore other, more flexible methods should be investigated for possible advantages. This problem can further be divided into smaller subproblems as follows:

1. *How can suitability analysis be solved using exploratory methods*? One of the main goals of this work was applying exploratory analysis to suitability problems. This research question can be divided into smaller subproblems:

   (a) *What kind of visualizations are available for suitability analysis*? The first subproblem was selecting and researching suitable visualizations for solving the problem. There are numerous different visualizations for depicting spatial data. The usefulness of each visualization depends on the problem at hand, the data used for solving a specific

problem instance, and the analysis methods used.

(b) *How can available visualizations be applied to suitability analysis*? The visualizations must be integrated into the data analysis process in order to help the analyst in their task. However, the visualizations, and the analyst's interaction with them, must be implemented in a way that assists in the analysis process. Therefore, care must be taken when selecting the visualizations and interaction methods used in the analysis process.

(c) *How can exploratory methods be used to solve the cross-country mobility problem*? Mobility is a problem that may need to be solved in various different environments, using whatever data happens to be available. Since the existing model-based approach to the problem requires specific input. Thus, flexible, exploratory methods can offer a more general alternative. Therefore, a way to solve this problem using exploratory analysis is desirable. Furthermore, such a solution can work as an example on how to use exploratory methods for solving suitability problems.

(d) *How can the exploratory suitability analysis process be modeled*? An exploratory analysis process is generally described as a data-centered, interactive process, where the analyst uses various means to delve into the data. A general model of this process is required in order to expand the work from the cross-country mobility example used in this research.

2. *How exploratory methods enhance cross-country mobility analysis compared to the existing model-based approach*? Problems can be solved using different methods. These methods have various advantages and disadvantages. These differences can be found, for example, in the process itself, in the metadata created by the process, or in the final output of the process. The focus of this part of the research are these three aspects of the analysis. Thus, this research question can be divided into subproblems.

(a) *How does the output gained from an exploratory process compare to the output of the model-based approach*? By comparing existing,

model-based mobility maps of a given area to maps of the same area created using exploratory process it is possible to see if there are differences in the output given by the two approaches. If the model-based map is considered to show mobility for the area accurately, then the exploratory process should be capable of creating a comparable map in order to be useful. More detailed analysis, where differences between the two approaches are examined in more detail, could also be possible.

(b) *Is there new (meta)data or information that can be found using the new process*? The result of both model-based and exploratory approach on suitability analysis is a suitability map. However, for the model-based approach this is the only information that is gained during the process. The details of the analysis, such as the metadata used during the process or the reason why certain point got a certain mobility value, are not preserved. In an exploratory approach, however, such data can be preserved and thus new information can be gained from the process.

(c) *Does the process based on exploratory methods offer improvements over the model-based process*? The two processes have a completely different approach on solving suitability problems. Thus, the processes need to be compared in order to see what advantages or disadvantages they have.

## 1.5 Structure of This Thesis

This thesis overview is divided into an introduction and three parts. The first part discusses the teaching of spatial data algorithms using web-based visual exercises, the second part discusses spatial analysis using exploratory methods, and the third part contains discussion and conclusion. The first part consists of four chapters, numbered from 2 to 5. Chapter 2 gives an in-depth introduction to the topic and chapter 3 discusses related research. Chapter 4 shows the visualization theory used in the work, and describes the implementation of spatial data algorithms learning environment. Chapter 5 discusses the results of using the learning environment on a real course on spatial data algorithms.

The second part consists of four chapters, numbered from 6 to 9. Chapter 6 introduces the topic, and chapter 7 discusses related research and other exploratory analysis systems. Chapter 8 describes the spatial analysis process, and how it can be modeled. Chapter 9 introduces the prototype spatial analysis software implemented in this project, and describes the results of using the prototype in a real problem solving situation.

The third part contains discussion of work done in parts one and two, as well as a comparison of the two processes. In the comparison, the goal is try and find similarities and differences between the processes and how this new information can be used to improve the processes. Conclusions and future work end the overview.

At the end of the overview there are two appendixes related to the spatial data algorithm teaching. Appendix A describes some spatial data algorithm exercises in more detail, while appendix B gives a list of implemented exercises.

# Part I

# Teaching Spatial Data Algorithms

# 2. Introduction and Background for Part I

The initial motivation for this research was improving the teaching of spatial data structures and algorithms at Aalto University. The SDA course has been a part of the school curriculum for years, and experience has proven it to be difficult for many students.Thus, the teacher in charge of the course wished to improve the student learning results on the course by applying new teaching methods.

## 2.1 Teaching Spatial Data Algorithms

The spatial nature of the data used in geoinformatics makes teaching the topic more complicated than teaching, for example, basic data structures and algorithms. Since the data items are multidimensional, the relationships between items can become rather complex. Data items can overlap, intersect, and the distance between items can be measured using several different metrics. Therefore, in order to show how spatial data items are related to each other, a multi-dimensional illustration that shows the data in their respective coordinates is required. Such views, however, are typically not very good at showing how the data structures used to store the data are composed. Therefore, in order to demonstrate how spatial data structures and algorithms work, multiple, simultaneous, and synchronized views are required. One view shows how the data items are related to one other, and another shows how the data structure used to store them is composed.

Multiple, simultaneous visualizations are commonly used in literature describing spatial data algorithms. Such views can be found in all relevant text books, as well as in several scientific articles, for example in [27, 34, 42]. For the learners, this means they must be able to connect several different views of the same data structure together in order

to be able to comprehend how the data structure is arranged. However, with just static pictures of the data, it is often hard to grasp how the data structure is modified as data is added or removed, or how the structure behaves with different input data. Such dynamic visualizations can be achieved by utilizing *algorithm visualization* to create *animations*.

## 2.2 Software Visualization in Teaching Data Structures and Algorithms

Algorithm visualization (AV) is a part of *software visualization* (SV), a branch of software engineering that uses graphics and animation to illustrate the different aspects of software [110]. Algorithm visualization can be used to illustrate how algorithms work through animations depicting how the associated data structures are modified during the algorithm's execution. Animation in AV is typically divided into two categories: smooth animation, where state transitions are shown explicitly by data elements moving around, and stepwise animation, which consists of a series of snapshots during the algorithm's execution. Both types of animation are used in teaching.

There is, however, strong evidence that the type of animation used in teaching is not as important as the *level of engagement* the learner has with the visualizations [49]. For learners to gain benefit from the use of software visualization, it must be used in a way that activates the learners and enables them to construct and refine their knowledge. Therefore using just passively viewed algorithm animations is not a very good way to improve learning.

Learners can be activated by exercises where problem solving is based on interacting with visualizations. For example, the learner could manipulate data structure visualizations in order to simulate how an algorithm modifies the data structures during execution. We call such exercises *Visual algorithm simulation exercises* [60].

We have implemented visual algorithm simulation exercises using the TRAKLA2 learning environment [73]. TRAKLA2 is a web-based learning environment where learners solve exercises through the web. The system uses *automatic assessment* to give the learners immediate feedback on their solutions without the need for instructor participation.

TRAKLA2 has been shown to be an effective learning tool for teaching basic data structures and algorithms [67, 75]. The spatial data algorithm

extension to TRAKLA2, introduced in [84], is the first time the system has been used to teach something beyond basic data structures or algorithms.

## 2.3 Contributions in This Part of the Thesis

Previously, the TRAKLA2 system included only basic data structures and algorithms, which manipulate one-dimensional data elements, such as numbers or strings. In order to include spatial data structures and algorithms, the system must be able to manipulate and visualize multidimensional data and data items. For spatial data, the most important data elements are two-dimensional points, lines and polygons. Such items can be visualized as key values, but for easier understanding multidimensional views should also be used.

This part of the thesis contains a theoretical framework describing how to visualize data structures on different levels of abstraction. The framework is based on four different levels of abstraction, and the concept of *canonical views* for basic structures. Canonical views are commonly used data structure visualizations that are familiar from practically all data structure textbooks and scientific papers. Such views can be used to visualize *any* data structure. However, canonical views are not very good for visualizing all aspects of data structures. For example, canonical views assume that data key values are described as text, which makes it hard to grasp how, for example, two-dimensional data items are arranged.

Some applications require other data structure representations. Such representations still contain some information about the data structures used. For example, when visualizing how two-dimensional points are structured in a quadtree [34], the two-dimensional view would show how different parts of the tree cover different subsections of the area. However, information about the data structure can be completely removed from the view in order to create domain-specific visualizations. Examples of such views are cartographic visualizations that show how the data is structured but do not contain any information about the data structures used.

This part of the thesis describes the spatial data structure extension to the TRAKLA2 environment. The extension contains a number of new data structures for storing multidimensional data, as well as visualizations for illustrating the data structures. A number of TRAKLA2 exercises have been implemented using these structures and visualizations. The exercises have been put into use in the spatial data structures and

algorithms course at the Aalto University. The learning results and student attitudes towards the system have been evaluated, and evaluation results are reported.

# 3. Related Work

## 3.1 Software Visualization

The use of software visualization in teaching began in the early 1980s, with the development of the Sorting out Sorting video [10]. The development of this one non-interactive animation took years [11]. Since then, there have been numerous algorithm visualization systems developed for educational purposes. Early systems concentrated on developing clear and attractive animations. The focus was on how well different aspects of visualization had been implemented, how versatile the graphical representations were, and whether the animation was smooth or step-wise. This is reflected in the early taxonomy by Price et al. [94]. Prominent systems from this period include Balsa [18] and its successor Zeus [19], Tango [109], and several others.

More recent research has revealed that the level and type of interaction enabled by the system is much more important for learning than the attractiveness or the level of detail of the visualizations [49]. The more recent systems have, therefore, been more focused on creating possibilities for learner interaction than developing extremely polished representations. Also recently, more and more systems have been designed to work in web-based environments, such as [39, 45, 82, 92]. There have also been efforts to create guidelines for the creation and evaluation of visualization systems [81]. Another aspect, which has become important, is the amount of work and effort required for taking a visualization system into use on a course, as well as the amount of effort required for creating new visualizations using a particular system [51]. Interaction between different systems has also been studied [54]. A reasonably recent overview of the algorithm visualization field can be found in [105].

### 3.1.1 TRAKLA2 System

TRAKLA2 is a learning environment for data structures and algorithms developed at the Department of Computer Science and Engineering at Helsinki University of Technology [73]. The system was originally implemented for teaching basic data structures and algorithms to undergraduate students. Since then, the system has been used in numerous institutions in both Finland and abroad [67]. The spatial data algorithm expansion is the first extension of the system beyond basic data structures and algorithms.

The TRAKLA2 system is the successor of the highly-successful automated assessment system TRAKLA, which was developed at TKK in the early 1990s [50]. Initially, the original TRAKLA system did not contain any algorithm visualization functionality. The system contained a number of *algorithm simulation exercises* that the learners received via email, solved using pen and paper, and submitted solutions via email. The email interface used a specified text format to represent the data structures. To solve the exercises, the learners had to simulate the manipulations an algorithm would do to a data structure using a given input. Typically pen and paper were used in this simulation process. Later, a web-based algorithm visualization system was added to TRAKLA, enabling students to solve the exercises graphically [59]. The visualization system was merely a graphical overlay on the original TRAKLA system: the exercises were still returned by the web applet using the same text format than was used with the email submissions.

As the limitations of the original TRAKLA system became more apparent, a completely new algorithm visualization system was designed. The new system was based on Matrix [61], a general purpose framework for creating algorithm visualizations and animations. Matrix is based on a very small number of fundamental visual components that are used to implement all visualizations used in the system. Unlike original TRAKLA, Matrix also allows the user to modify actual, implemented data structures through manipulation of the visualizations. Therefore, using Matrix, it is possible to create visual algorithm simulation exercises where the user-submitted simulation sequence can then be compared to a sequence created by a real algorithm. In TRAKLA, the correctness of algorithm simulations was checked by taking one or two snapshots of the algorithm simulation sequence.

Furthermore, the Matrix framework allows for multiple synchronized visualizations of the same data structure. Therefore, it is possible to, for example, view a binary heap both as an array and as a binary tree at the same time. Modifications done to one visualization will then be reflected in all views. Moreover, the Matrix framework allows hierarchical visualizations. For example, a B-tree can be visualized by using array visualizations to show how data elements in each tree node are arranged. The arrays are contained inside the visualizations of the tree nodes.

Based on the Matrix framework, a new learning environment called TRAKLA2 was implemented in of the beginning 2000s [73]. The system replaced the old TRAKLA in 2003. Since inception, the TRAKLA2 system has been improved and expanded constantly.

## 3.2 Teaching Geoinformatics

In many institutions, teaching geoinformatics is focused on how to use available tools and techniques to solve geographical data problems. Therefore, courses tend to concentrate on teaching how to use common geoinformatics tools such as MapInfo, ArcGIS, or some other GIS software, and how to apply techniques such as numerical modeling [55], simulations [26], or exploratory data analysis [6]. The requirements for understandable and clear maps are also taught [108]. In addition to common geoinformatics tools, internet and eLearning environments have also been used [29, 95].

There are few institutions that have courses with a technical point of view on the techniques and algorithms used in geoinformatics, but even when spatial data algorithms are included, the course probably does not use algorithm visualization systems. The only other large collection of spatial data structure and algorithm visualizations the author is aware of, is the VASCO Spatial Index Demo collection of Brabec and Samet[1], which is related to a number of books by Samet [99, 100, 101]. Unfortunately, this collection has not been under active development in the recent years.

There are, however, several algorithm visualization systems that illustrate some data structures or algorithms used in geoinformatics. For example, there are several systems that visualize geometric algorithms or Voronoi diagrams [35, 45, 46, 107]. Such systems seem, however, to be aimed at computer science curriculum, and none of them come even close

[1] available at http://donar.umiacs.umd.edu/quadtree/

to covering the basics of spatial data structures used in geoinformatics. There does not currently seem to be much active research interest in the topic. A 2008 paper on the MAVIS system had one spatial algorithm example [58], but the author is unaware of any other recent progress in the field[2].

### 3.3 Visualization in Geoinformatics

In geoinformatics, the use of maps is ubiquitous, making it an intrinsically visual field. Therefore, the creation of clear and understandable maps is an important, and much studied topic. The creation of maps has been studied for a long time and the basics are typically covered in every GIS textbook. There is a wealth of literature on map making, including textbooks on how to create maps, and scientific literature discussing map making concepts. See [17, 32, 66, 98, 108], for some examples, and [77] why care must be taken in map-making. The basics of cartographic visualization appear to be well established these days. Thus, research tends to focus on new visualization strategies for specific purposes or situations.

One active area of research is the visualizations used in exploratory analysis of geodata [116]. Exploratory data analysis is an analysis method where the user explores the data by using various visualizations, in order to find hypotheses to test, for example, or assess assumptions about the data. The exploration requires dynamic, interactive visualizations that may differ greatly from traditional printed maps [37]. In many cases using only maps, even dedicated and customized thematic maps, is not sufficient for showing all relevant aspects of the data in question. Therefore, geoinformatics applies several other types of information visualization methods [63, 70, 106]. Exploratory analysis uses several analytical methods, many of which are also used in data mining [16]. The use of exploratory analysis in actual problem solving has also been evaluated [6].

Research on geovisualization touches many topics, ranging from data uncertainty [70, 122] to ontological similarity [38] and distributed modeling [47, 89]. Applications range from evaluation of the quality of life [97] and human activity patterns [96] to data quality assessment [40], forestry [6], and symbology [62]. In addition to visualization, other elucidation methods, such as auralization, have also been researched [20].

[2]A spatial algorithm visualization example was one of the winners of AlgoViz awards in 2010. That, however, is not a scientific contribution.

# 4. Spatial Extension to TRAKLA2

## 4.1 Levels of Algorithm Visualization Abstraction

In algorithm visualization, we are interested in illustrating the organization of a data structure. In order to be effective, visualizations must be clear and understandable, so that the viewer can comprehend what the visualizations are depicting. One way to achieve this is to use widely-accepted and commonly used visualizations, which we call *canonical views*. Examples of these views are shown in Figure 4.1. Such views of data structures can be found in practically all books and scientific papers discussing the topic. For example, most figures depicting data structures in the Introduction to Algorithms book [25], which is used as course book on the first data structure course at Aalto University, are variations of canonical views.

Canonical views, however, are not always sufficient for depicting all aspects of a data structure or algorithm. For example, when using just canonical views to visualize spatial data structures, it is hard to grasp how two-dimensional data items are related to each other. When the data is drawn to a two-dimensional plane, the relations between data items can easily be seen, but the details of the data structure are harder to understand. For example, both Figure 4.2 and Figure 4.3 are required to grasp how a quadtree is arranged, and how the data stored in the quadtree is arranged in two dimensions. Therefore, there are several cases were multiple simultaneous views are useful for representing a data structure or algorithm more understandably. Furthermore, these views may be on *different levels of abstraction* in order to bring different aspects of the structure into focus. In the following, we define levels of abstraction for the visualization of data structures and algorithms.

The levels of data structure abstraction were first introduced in Publication I. A slightly revised version, where the names of a few abstraction levels were made more descriptive, was described in Publication IV. The following will briefly go through the levels of abstraction using quadtree [34], a spatial data structure, as an example. In the publications, two one-dimensional data structures, binary heap and B-tree, were used as examples.

There are four levels of abstraction in the model. They are, from the lowest level of abstraction to the highest, *elementary structure level*, *data structure level*, *representation level*, and *domain level*.

### 4.1.1 Elementary Structure Level

Most data structures are created from simple, generic structures, such as arrays, trees, lists and graphs. These structures are archetypes, or reusable, basic building blocks for creating data structures. We call them *elementary structures*, and visualizations of these structures are at the elementary structure level. This level was called basic structure level in Publication I.

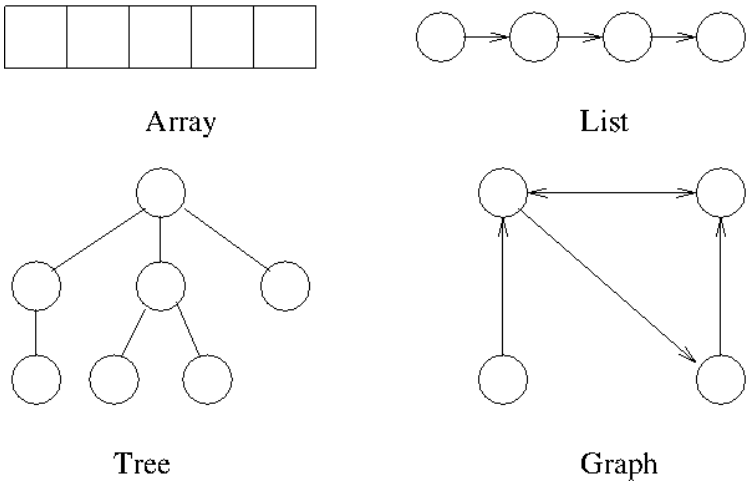


**Figure 4.1.** Canonical views of elementary structures.

Since elementary structures are used in the implementation of most data structures, all of them have well-known visualizations, or *canonical views*, shown in Figure 4.1. The canonical views are rather close to the actual structure implementation, but they are still abstractions. An array, for example, is visualized as a set of boxes arranged side-by-side to represent the different array indices. An actual array implementation,

however, is typically a continuous block of memory locations for storing a specific type of variable.

Being general building blocks, the elementary data structures have no set semantics. The definition of an array does not dictate the data type used, the order of, or the values for the items inside the array. For effective data structures, however, we need to impose a set of semantics on the data structure in order for algorithms to work efficiently. Similarly, the visualizations need to reflect these semantics. In many cases, this just means that the data in the structure is arranged in a specific way. However, illustrating some data structures, like the quadtree, may require additional visual cues. In the case of quadtree, we need to label nodes in order to know which quadrant of the parent node each child covers.

### 4.1.2 Data Structure Level

At data structure level, the focus is on visualizing the internal composition of a data structure. On this level the data structures are typically implementations of an abstract data type (ADT). The ADT defines the operations that the data structure must conform to, while the implementation of the operations depends on the data structure. The visualizations on this level use canonical views to present the physical configuration of the data structure. The layout of the visualization can, however, be customized in order to show some aspects of the data structure more effectively. For example, if the data structure was a graph, where the vertices have geographical locations, the vertices could be drawn in their correct coordinates.

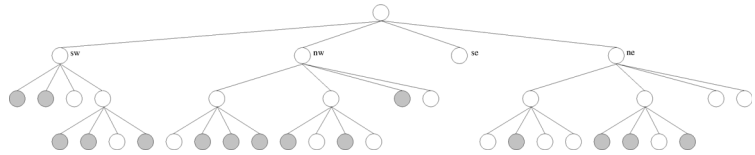


**Figure 4.2.** Tree view of an region-quadtree.

An example of a data structure level visualization can be seen in Figure 4.2, which shows a region-quadtree using tree visualization [34]. Region-quadtree is a data structure that covers a two-dimensional area. All data values in the tree are stored in leaf nodes, and internal nodes of the tree merely contain information about the area the particular node covers. Each node of the quadtree divides the area it covers into four quadrants, and each quadrant is covered by one of the children of the node. The di-

vision continues recursively until a node contains at most one data value. In this case, the region-quadtree is used to store a binary raster, where different values are shown in leaf nodes using white and gray. In general, quadtree nodes can contain a certain, set number of elements in each node, and the structure can be generalized into more than two dimensions. Detailed discussion about quadtrees can be found in the works of Samet [101, 100, 99].

While the figure accurately depicts how data is arranged in the quadtree, a human viewer cannot really comprehend what the data represents. However, a skilled observer might be able to see certain details. For example, the width and height of the area - in pixels - could be estimated from the height of the tree. Still, a completely different view is required for seeing what the data depicts.

### 4.1.3 Representation Level

A single data structure can be presented in various ways in order to emphasize different aspects of the structure. A binary heap, for example, could be presented as an array in order to show how the heap is actually implemented, or as a binary tree in order to show how the logical structure of the heap maintains the heap property.

On *Representation level*, we use different data structure visualizations in order to emphasize different aspects of the structure. There is no longer a need to maintain conformance to the data structure implementation, and visualizations other than canonical views can be used. While all data structures can be visualized using canonical views, such views are not always the best ones for illustrating all aspects of the structure.

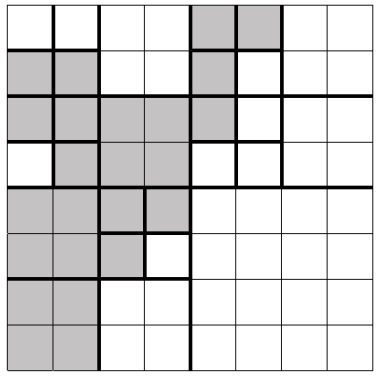

**Figure 4.3.** Area view of a region-quadtree.

In Figure 4.3 the area covered by the region-quadtree shown in Fig-

ure 4.2 is visualized. The emphasized lines represents how the quadtree divides the area into smaller pieces. From this area view of the data, it is easy to see how the data items are arranged. However, the exact composition of the data structure is harder to see, and the order of the child nodes is impossible to ascertain from the view.

### 4.1.4 Domain Level

All previous levels of visualization have included some details of the data structure. At the topmost level of the visualization, the *domain level*, we visualize the data in domain-specific ways that may exclude all details of the underlying data structure. At this level, we are moving from using algorithm visualization to using information visualization techniques. For example, a raster map could be indexed using a region-quadtree, but when the map is viewed using a GIS software, this fact is not shown to the user.

## 4.2 Visual Algorithm Simulation Exercises

Problem solving in TRAKLA2 is based on the concept of *visual algorithm simulation*, first introduced in [60]. In visual algorithm simulation the user simulates an algorithm by modifying data structures the same way an actual algorithm would do. The modifications are done through manipulation of data structure visualizations. Currently, the TRAKLA2 system contains two different types of visual algorithm simulation exercises.

In *tracing exercises*, the learner is given an input sequence, data structures, and an algorithm to simulate. The goal is to manipulate the data structures in order to faithfully duplicate the modifications a real algorithm would do. Such exercises are assessed by comparing the manipulations the learner made to actual algorithm execution. An example of a tracing exercise is simulating the build-heap algorithm for constructing a binary heap.

In *open tracing exercises*, the learner is given an input and the desired output. The goal is not to duplicate the execution of any specific algorithm. Instead, the intention is to create a specific output. The learner may do any manipulation enabled by the exercise, and the correctness of the submission is assessed by comparing the simulation's final state to the correct solution. An example of an open tracing exercise is coloring an AVL-tree into a red-black tree.

Tracing exercises are currently much more common than open tracing exercises. In the spatial data extension, there are 10 tracing and 3 open tracing exercises.

### 4.3 Implementation of the Extension

The exercises implemented in this work can be accessed as Java applets through the web[1]. Those interested in the full TRAKLA2 learning environment should contact the Learning + Technology research group at Aalto University[2].

The implementation of the spatial data structure extension was first described in Publication II. The implementation consists of three different parts: spatial data elements, spatial data visualization, and spatial exercises.

The spatial implementation is the first expansion of the TRAKLA2 system beyond basic data structures and algorithms. The new exercises are based on the same basic look and feel, features, and graphical user interface as older TRAKLA2 material. However, they have been modified for the requirements of spatial data and data structures.

Exercises in TRAKLA2 are solved by manipulating data structure visualizations via an applet. Figure 4.4 shows a screenshot of a TRAKLA2 exercise page. On the left side, at the top of the figure, there are buttons with which the user can show or hide the explanatory text and the pseudocode, as well as open them in a separate window. On the top right are buttons for going to the next and previous exercise. Below those is the explanatory text which tells what the student is supposed to do in the exercise. In a separate tab are more detailed instructions on how to use the exercise applet. Below those is the pseudocode for the exercise and the exercise applet.

Using the exercise applet, a student can solve TRAKLA2 exercises by manipulating the data structure visualizations. The exercise in the screenshot is simulation of the Douglas-Peucker line simplification algorithm. The algorithm simplifies a polyline by removing unneeded points from the line. The student can solve the exercise by dragging and dropping points from the area visualization to the stack, or to the linked list representing the simplified polyline. In the area visualization, white lines are parts

[1] http://www.cse.hut.fi/en/research/SVG/TRAKLA2/exercises.shtml
[2] http://www.cse.hut.fi/en/research/LeTech/

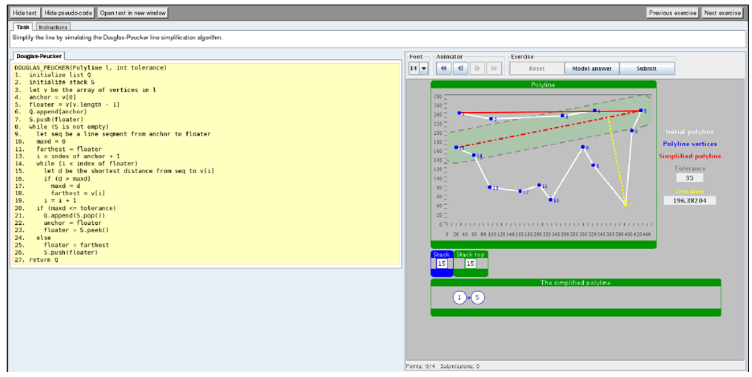

**Figure 4.4.** Screenshot of a TRAKLA2 exercise page.

of the original polyline, red lines are parts of the simplified polyline, and the red dashed line is the next candidate line for the simplified polyline. The candidate line is surrounded by a buffer zone. The point, for which the distance from the candidate line is currently measured, is indicated by the yellow dashed line.

At any time during an exercise, a student can use the 'model answer' and 'submit' buttons. The model answer button opens the model answer for the exercise. For tracing exercises this is an algorithm animation solving the problem instance, and for open tracing exercises a visualization of the correct solution. Opening the model answer will disable the submit button, which is used to send the student's submission to the TRAKLA2 server. Pushing the submit button will also give the student feedback on their solution, including the number of correct simulation steps executed and the points gained from the exercise. After submitting the exercise or viewing the model answer, the student can use the reset button to initialize the exercise with new input. The number of times an exercise can be submitted may be limited or unlimited.

### 4.3.1 Spatial Data Elements

Each two-dimensional point needs to store at least three primitive data values: the x- and y-coordinates, and the data value associated with the spatial item. If the spatial data item is more complicated than a point, other spatial attributes of the item have to be stored as well. A polygon, for example, can be stored as a series of points, each corresponding to one polygon corner.

Spatial data items are therefore not primitive data, but are instead compound *structures* that contain a number of primitive data values with a

given semantics. In TRAKLA2, this is accomplished by a generic structure object that can be extended to contain any number of primitive data values and given a set of semantics. Structures are used to implement the three important spatial data elements: points, lines and polygons.

### 4.3.2 Spatial Data Visualization

Spatial data items can be visualized in at least three ways. First, we can visualize the structure as a set of independent values, each of which has a label that indicates the semantics associated with the value. Second, we can visualize the structure using a key value that identifies it without showing the associated primitive data elements. Third, we can visualize the data items in the space they occupy. For spatial data elements in TRAKLA2, all three visualizations are possible.

The first visualization is implemented using a specific structure visualization, where the fields of the structure are named. A structure is then visualized as a compound object, where it is possible to manipulate the values of individual fields. The second visualization is the same key visualization which is used for basic data items in the system. These two visualizations can be used on any abstraction level.

The third visualization is a separate *area view*, designed for visualizing two-dimensional areas and spatial data items stored in the associated data structure. The area view can also contain conceptual elements associated with an algorithm, such as the sweep line of a line sweep algorithm. Such elements do not explicitly exist in normal algorithm implementations. The area view is a representation level visualization.

An example of the different levels of visualization is shown in Figure 4.5, which depicts an exercise where the student simulates the line sweep algorithm [13] for finding line segment intersections. On the top-left in the figure is a binary heap that stores points depicting either line segment endpoints or intersections. The structure visualization is used for these points in this view. For each point, the visualization contains the point's key value, the x-coordinate, and the y-coordinate. On the top-right of the figure, the points are visualized in two-dimensional area using a labeled point visualization. In this visualization, each point is drawn to its proper coordinates, and the key value is shown next to the point. A key value visualization for points can be seen the two bottom views of Figure 4.4.

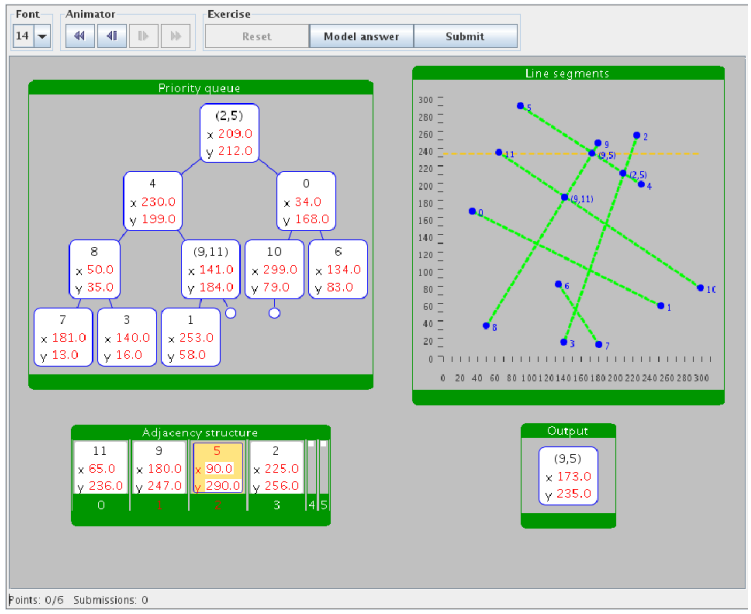


**Figure 4.5.** Screenshot of a TRAKLA2 line segment intersection exercise showing 2D points visualized using both point and structure visualizations

### 4.3.3 Spatial Exercises

Using the spatial data elements and visualizations together with normal data and data structure level visualizations of the various data structures, it is possible to create spatial TRAKLA2 exercises. On a conceptual level, the spatial exercises differ from other exercises in the system only by using spatial data. There are, however, some implementation level details, which are different between non-spatial and spatial exercises. This section discusses some of the differences and difficulties encountered when developing the exercises.

The most important difference is that multidimensional data requires more complicated visualizations than one-dimensional data. Basic data structure or algorithm exercises can be implemented using only canonical views. Furthermore, most of the time the canonical visualizations are used on the elementary and data structure levels. In spatial exercises, on the other hand, area views are required. Therefore there are representation level visualizations used in each exercise. Moreover, most exercises require the use of multiple views on combined levels in order to visualize all relevant aspects of the data and data structures.

Another important difference between spatial and non-spatial exercises is the generation of random input data. The creation of random, one-

dimensional, data seldom has many constraints. Therefore, most non-spatial exercises can receive their input from the same source of random data. For two-dimensional data, however, the input must be such that it can be visualized in an informative manner as well as manipulated in the area view. Furthermore, what is considered “informative”, and how the data needs to be manipulated, depends on the exercise. Therefore, for 2D data, there is no common data source that can be used to generate input data. Each exercise requires its own input generator.

Two-dimensional visualizations often require some extra effort to make them as understandable as possible. For example, the area view for a quad-tree contains information about both data items stored in the tree and how the area is divided into smaller subareas by the tree. Similarly, in a line sweep exercise, a clear difference must be made between the sweep line and the line segments that are the input data.

A total of 13 spatial exercises for TRAKLA2 were implemented in this project and used on the spatial data algorithms course. Some of the exercises were deemed successful in depicting how the particular algorithm works, and have a great potential as learning aides. Others were less successful. Some example exercises are described in Appendix A.

# 5. Use of Spatial TRAKLA2 in Teaching

At Aalto University the spatial extension to the TRAKLA2 system was first taken into use in the spatial data algorithms course in spring 2007. For this research, the use of the system was monitored during the spring 2007 and spring 2008 courses. Some information from spring 2009 is also included. During that time, the system was expanded to include new exercises, old exercises were updated with better user interface, and a large number of software bugs were fixed. Most of the modifications were done during the 2007 course (mainly bug fixes) and between the 2007 and 2008 courses.

Use of the TRAKLA2 spatial extension in teaching is described in Publication IV. The learning results gained using the system, and the students' attitudes towards it, were first reported in Publication III. The results were analyzed in more detail in Publication IV.

## 5.1 The Spatial Data Algorithms Course

The spatial data algorithms course is taught at Aalto University by the Department of Surveying. The goal of the course is to teach some of the basic data structures and algorithms required in geoinformatics. The structures are typically taught at a rather high level of abstraction. The goal of the course is not to teach how to implement the data structures and algorithms. Instead, after the course, the students should be able to compare different problem solving methods, know how several common geoinformatics problems are solved by a computer, and be able to select the correct problem solving method for a given situation.

For years, the course was taught using the same format. Lectures were four-hour sessions that started with a short traditional lecture of approximately one hour. After that, the students did group work for two hours,

familiarizing themselves with one data structure or algorithm. At the end of the session, one hour was reserved for the groups to present their work to other students. After the lecture, each group had to write a one- or two-page summary of their work. The course typically consisted of six or seven lectures, after which the students had their first chance to take the final examination. After the lectures, the students also made a small programming exercise either alone or in pairs. In the exercise they implemented one of the algorithms discussed on the course.

The TRAKLA2 SDA extension was first taken into use in spring 2007. The system was added to the course without making any other modifications, and was used the same way in 2008. In 2009 the course lecturer changed, and the course underwent major changes. At the same time the way TRAKLA2 was used on the course changed.

In 2007 and 2008 TRAKLA2 exercises were a compulsory part of the SDA course. The students needed to gain at least 50% of the maximum points from TRAKLA2 in order to pass the course. No further benefits could be gained by getting more points. In 2009, TRAKLA2 was no longer a compulsory part of the course. Instead, the students could gain 2 additional points for the final examination by solving exercises. The maximum points for the final exam, excluding these bonus points from TRAKLA2, was 24 points.

Between the 2007 and 2008 courses, several bugs in the exercises were fixed, two new exercises were added to the course, one exercise was removed, and one exercise was completely redesigned. Between 2008 and 2009, one new exercise was added to the course.

### 5.2 TRAKLA2 Results for the SDA Course

The results of using the system were assessed in two ways. The first point of the assessment was the students' learning results, which were measured by comparing TRAKLA2 results to exam results. The second point of assessment was the student attitudes towards the system, which were measured using course feedback forms and interviews. Feedback forms were used each year, and interviews were conducted after the 2008 course. The following will give a brief summary of the learning results. Detailed description and analysis of the learning results can be found in Publication IV.

The learning results were evaluated using both quantitative and quali-

tative methods. The quantitative method used was linear regression. The linear regression analysis was used to see the correlation between the students' TRAKLA2 results and the exam results. It was also used to see to what extent the TRAKLA2 results could explain the exam result variation.

The qualitative analysis method used was content analysis [64]. It was used to investigate how the students' experiences with TRAKLA2 exercises were reflected in their exam answers. It was also used to see what misconceptions the students had about data structures and algorithms.

### 5.2.1 Quantitative Learning Results

Table 5.1 contains basic course information and the students' TRAKLA2 results. The table shows how many students started the course each year, and how many of them participated in the first examination. Also, the table shows the number of TRAKLA2 exercises on the course, how many of those were spatial exercises, as well as the total number of exercise submissions, and the average score given to the students. The table contains information for all three years.

**Table 5.1.** Basic course data for the Spatial Data Algorithm course

| year | # students | # in exam | # exer. (SDA) | # subs | avg. score |
|---|---|---|---|---|---|
| 2007 | 16 | 10 | 15 (9 SDA) | 723 | 67% |
| 2008 | 20 | 16 | 16 (10 SDA) | 1036 | 83% |
| 2009 | 13 | 10 | 14 (11 SDA) | 662 | 80% |

The linear regression analysis was done twice. The first analysis was for all TRAKLA2 exercises and the whole exam; the second analysis was for the TRAKLA2 exercises and exam questions that covered R-trees [42]. Table 5.2 contains the results of the linear regression analysis. The table shows the number of students included in the analysis as well as the $\rho$ (correlation), adjusted $R^2$ (strength of relationship) and p (statistical significance) for both the whole exam and the R-tree questions. Table 5.2 contains results only for years 2007 and 2008. In the 2009 course there were no exam questions concerning R-trees and the way TRAKLA2 was used on the course was radically different from the previous years. Therefore the 2009 results would not be comparable to the previous years.

The quantitative analysis was actually done twice, and the results were first reported in Publication III. The analysis was later revisited in Publi-

**Table 5.2.** Learning results on the spatial data algorithms course

| **Course info** | | **Whole exam** | | | **R-trees** | | |
|---|---|---|---|---|---|---|---|
| Year | N | $\rho$ | adj. $R^2$ | $p$ | $\rho$ | adj. $R^2$ | $p$ |
| 2007 | 10 | 0.74 | 0.50 | 0.01 | 0.90 | 0.78 | $< 0.01$ |
| 2008 | 16 | 0.48 | 0.18 | 0.058 | 0.55 | 0.25 | 0.03 |

cation IV. The results in Tables 5.1 and 5.2, are those reported in Publication IV. Results from the 2009 course were not available when the results were originally written, and were added to Table 5.1 in this work. The analysis was redone since some of the data was actually missing in the analysis reported in Publication III. The analysis was corrected for Publication IV. There were no statistically significant differences between the results in the two publications.

The analysis indicated that the use of TRAKLA2 in SDA teaching had similar results to using the system in teaching basic data structures and algorithms, as reported in [75]. The $\rho$ values, which indicate correlation between the TRAKLA2 exercises and the exam results, are over 0.7 for the 2007 course, and above 0.48 for the 2008 course, and the p-values, which indicate the strength of the statistical relationship, are well under the 0.05 mark, with the exception of the whole course in 2008. This indicates that students who did well in TRAKLA2, also did well in the exam.

This does not prove that TRAKLA2 promotes learning. However, that is one explanation for these results. TRAKLA2 is a tool the students can use to learn how spatial data algorithms work. Those students, who learn how to solve TRAKLA2 exercises, can be assumed to know SDA better, than students who do not learn how to solve the exercises. The students, who can solve TRAKLA2 exercises well, typically also do well in the exam. Thus, it can be argued that TRAKLA2 can promote student learning.

However, the class sizes on the SDA course were so small that any statistical results should be assumed to be merely indicative instead of definitive. Both the 2007 and 2008 courses have sufficient number of data points for statistical analysis. This does not, however, change the fact that number of data points is small and thus it is easy to induce bias in the results. Thus, one should be very careful when drawing conclusions from these results.

The exam results from the years 2004-2009 were also compared. The exams used in the years 2006 and 2007 had a maximum of 30 points,

whereas the other exams had a maximum of 24 points. For the comparison, these two exams were scaled to 24 points. After this, the Shapiro-Wilks test was used to confirm that all data sets were normally distributed. Welch's t-test and Mann-Whitney U test were used to see if there were any statistical differences between data sets.

No significant differences were found between data sets. This was, however, to be expected. The exams used on the course typically had four questions where essay-type answers were expected. In 2006 and 2007 there were five questions. The exams were graded by the same person (with the exception of the year 2009 exam, when the course lecturer changed) without using any external, objective assessment criteria. The exams from different years cannot be compared. The assessment for each year reflects the variation between exam answers for that year, and does not contain any knowledge of the previous years' exams. Therefore, the results merely show that the data from different years is not comparable and indicates nothing about whether TRAKLA2 affects learning.

Interestingly, the data from the year 2005 had statistically different mean compared to 2006 and 2009 data ($p=0.047$ and $p=0.049$, respectively) shown by Welch's t-test. Mann-Whitney U test did not, however, show significant differences between these data sets ($U=83.5$ $p=0.064$ and $U=60.5$ $p=0.075$, respectively). The hypothesis is that exam grading on the 2005 course was more lenient, although other causes are also possible.

This shows, again, how difficult it is to find the influence of any particular learning method or other variable without proper experimental setup.

### 5.2.2 Qualitative Learning Results

The qualitative results were first described in Publication IV. The method used was content analysis, and two questions were considered: how students' TRAKLA2 experiences were reflected in their exam answers, and what kind of misconceptions the students had. Each exam question was considered separately, and the analysis was done by question and not by student. Therefore the results reflect, for example, what kinds of misconceptions students have about a given algorithm, and not what kinds of misconceptions a single student has.

Depending on the exam question, there was none, some, or extensive influence of TRAKLA2 in the exam answers. The influence was primarily observed in the diagrams students drew in the exams. TRAKLA2 also

had some influence on which parts of the data structure or algorithm the students' answers concentrated upon. Therefore, it is clear that the students gained at least some knowledge from using the TRAKLA2 system, and are able to apply this knowledge in the exams. While the evidence of the influence of TRAKLA2 is far from universal among the students' exam answers, it indicates that TRAKLA2 does play some part in the students' learning process.

Several common misconceptions were also discovered in the R-tree questions in the exam. For example, several students thought that R-tree nodes can hold only two data elements (like the example in [121] had), or that an R-tree is created by recursively dividing the area into two subareas covered by the nodes' children. In reality R-trees, like B-trees they are based on, can have many data elements in each node, and R-trees are created by repeatedly adding new data elements to the structure.

The misconceptions clearly show that many students' understanding of how the R-tree data structure is created and how it works was still fundamentally flawed at the time of the exam. Furthermore, these misconceptions are such that the use of TRAKLA2 should remove them. In the TRAKLA2 R-tree exercise, the students are required to insert polygons into an R-tree. The tree nodes hold a maximum of three elements, and the number of polygons to be inserted is sufficient to create a situation, where the root node has more than two child nodes. However, there were students who did such mistakes despite the presence of a tool which should have exposed such misconceptions. It was not investigated whether the students who held these misconceptions were able to solve the TRAKLA2 R-tree exercise or not.

### 5.2.3 Student Attitudes

The student attitudes towards the TRAKLA2 system were collected using two methods, a course feedback form at the end of each course, and a number of interviews after the 2008 course.

The course feedback *questionnaire* had a number of questions on the TRAKLA2 system. The students were asked to grade the system as a whole, as well as give their opinion on the usefulness of various features of TRAKLA2 ranging from pseudo-code to the user interface of the applet. The questionnaire also had questions where the students were to grade the teaching methods used on the course, learning materials used, and other aspects of the SDA course. The students could also tell which parts

of the course they liked and which they disliked. The student attitudes were first reported in Publication III. Some further analysis was done in Publication IV.

According to the feedback questionnaires the student attitudes to the TRAKLA2 system changed radically between the two years. In 2007 TRAKLA2 was ranked as the second-lowest of all teaching methods on the course, while on 2008 and 2009 it was the highest-ranked method on the course. Furthermore, in 2007 many students stated that TRAKLA2 was their least-favorite part of the course. In 2008 and 2009 no student regarded TRAKLA2 as the least-favorite part of the course, and to some it was the favorite. It should be noted, however, that in 2009 the number of feedback questionnaires received was much smaller than in the previous years.

The main difference in the TRAKLA2 system between the year 2007 and the later years was that a number of irritating bugs that had not been noticed in testing were corrected and that a couple of badly implemented exercises were removed. Thus, this again shows how small things can affect the users' opinion radically. Most of the TRAKLA2 was the same in all years. Thus, it was likely the small and easily corrected, but annoying bugs in the 2007 version, which made it so unpopular with the students. Furthermore, most of the bugs had been removed by the end of the 2007 course, but the students still held a low opinion of the system. Thus, it has been shown again, that the initial impression is important.

A total of four *interviews* were conducted after the 2008 course. The interviews followed the interview guide approach [90]. The approach states that an interview is conducted using a general outline of topics, but the interviewer is free to vary the wording and order of the questions to some extent. The interviewees consisted of one Finnish male, one Finnish female, one foreign male, and one foreign female student. The ages of the interviewees were between 22 and 28 years, and their backgrounds varied. A detailed report of the interviews can be found in Publication III.

Two main paths of questions were explored. First, what was the student's subjective opinion of the system, and second, what they thought about it compared to other teaching methods and learning materials. The interviews revealed both strengths and drawbacks of the system.

The interviewees found the system to be beneficial and thought that it was an important learning tool that should continue to be utilized on the course. Also, by using the TRAKLA2 system, the interviewees felt they

could grasp details of the algorithms that did not come up in the lectures. Furthermore, interviewees felt that by doing algorithm simulations it was easier to remember the details than by using other learning methods.

The interviewees criticized the graphical user interface, and the feedback gained from most exercises. The required GUI actions used to solve an exercise varied between different exercises. Therefore students had to spend some time to learn the user interface first, before they could start solving the exercise. The feedback on most exercises was just the number of points gained, and the number of correct simulation steps done. The interviewees would have wanted more detailed feedback. The interviewees also thought that it was cumbersome to go through the simulation steps in order to find the error. Also, the fact that the system did not give any points after the first error was thought to be unfair.

According to the interviews, the students' overall attitude towards the system seems to be more or less positive. However, there are still many things in the system that could be improved. Perhaps the most important finding for this particular study was the students problems with the GUI. It is an indication of problems with the system design. Either the point-and-drag GUI semantics of TRAKLA2 should be expanded in some manner, or more attention should be given for designing an understandable and consistent GUI for spatial exercises.

# Part II

# Spatial Analysis

# 6. Introduction and Background for Part II

The original motivation for this research was the limitations of traditional model-based spatial analysis. Models are typically tied to a specific situation. A model takes certain, specified input and produces an output. This means that the model cannot work without the given input being present, and often the process of how the input is combined to a specific output cannot be examined. Thus, a model easily turns into a black box, where it is hard to verify the correctness or usefulness of the output. Exploratory methods, where the process can easily be traced and verified, offer an alternative to modeling.

The ability to trace and verify an analysis makes the process *transparent*. Besides making it possible for the analyst to later review the analysis, it also makes it possible for other people to familiarize themselves with the analysis and thus verify whether the analysis output is useful. Such a feature could be useful in, for example, multinational crisis management. Crisis management exercises have shown that in a crisis management situation with numerous different organizations, these organizations often do not trust analyses done by other organizations. Therefore, during an exercise many organizations request that others give them only the raw data instead of any analysis results. This situation and its consequences are described in Publication V and Publication VI.

There are at least two reasons for refusing analysis results created by other organizations. First, different organizations have different values and knowledge, and their analysis results reflect this. Thus, an analysis done by another organization may not reflect the needs of the recipient organization. Second, the details of a particular analysis process are typically not revealed to other parties. Thus the recipient organization has to use the analysis results without detailed knowledge of its creation. Without this knowledge the recipient organization can not infer how the pro-

cess, and the values and knowledge used in it, have affected the result, and thus may feel that they can not trust the result.

Furthermore, in crisis management, it is impossible to know beforehand what sort of data will be available. Often the data on the crisis area needs to be compiled from a variety of sources, including publicly available maps, satellite images, on-site measurements, and such. It is similarly not possible to know beforehand what kind of problems need to be solved. This leads to a situation where general methods, which can be used to solve a wide range of problems and work with whatever data happens to be available, are required.

Thus, the analysis methods selected for use in such a situation should be *free of values*, so that they do not inherently show bias towards any organization and therefore can be accepted as tools by several organizations, *transparent*, so that anyone can trace the details of the analysis process and therefore validate the results of an analysis, and *general*, so that they can be used in many situations, using whatever data is available.

The problem used as a case example in the research is off-road mobility analysis, where the goal is to create a map that shows how difficult it is for a specified vehicle to advance over terrain. Typically, in crisis management, the goal of mobility analysis is to create a mobility map that depicts the maneuverability of the terrain in the operational area. Mobility in such a map is divided into a number of mobility categories. Each category represents how difficult it is to travel over area covered by the category. Mobility can, for example, be divided into three categories: NO GO for areas that cannot be crossed, GO SLOW for areas where maximum practical speed is slow, and GO for areas where it is possible to drive fast.

### 6.1 Suitability Problems

Dividing an area into mobility categories is an example of a *suitability problem*. In general, the goal of suitability analysis is to find a location, or locations, best suited for a given activity, or to categorize locations according to their suitability. The off–road mobility problem belongs to the second type of suitability problems.

How different locations are assigned to categories depends on the input, the type of vehicle, and several other considerations. However, in the end, all locations belonging to a category have similar overall suitability scores. This means that these locations are somehow similar, and there-

fore *similarity measures* can be used for solving the problem. Dividing the area into suitability categories is a general method that can be used to solve also many suitability problems.

In this work, an analysis process, which can be used to solve many types of problems, has been developed. Off-road mobility is used as the example problem, and the main analysis methods used are *exploratory data analysis*, which is an interactive, data-driven method of data analysis [9],and *visual analytics*, which is a method of analytical reasoning facilitated by interactive visual interfaces [114]. When using exploratory analysis, the user explores the data using various interactive means in order to gain new insight and find new information; in visual analytics, the user explores the data sets using various interactive visualizations, and thus gains deeper insight on the data. Using the new information gained from this interaction, the user can then make decisions and solve problems.

Suitability problems can typically be solved by creating a scale of suitability for each available input data set, combining the datasets, and then comparing locations against the scale. This also means that the suitability of locations can be compared by measuring their similarity; locations that are similar typically have the same suitability. Thus, methods that use the similarity to group locations into categories, can be used to solve this problem.

*Clustering* is the task of dividing a set of data items into subsets according to their similarity. Elements in the same subset are similar to each other, and are as different as possible from the elements in other groups. In cluster analysis, the goal is typically to see whether the data can be divided into natural subsets, which are clearly distinct from each other. Thus, the goal is to discover something about the nature of the data being analyzed [43]. In the case of off-road mobility, the goals could be: to see whether there is a subclass of good mobility that is clearly distinct of classes of bad mobility; to see whether there is a subclass or subclasses of fair mobility, and what are the differences between these; to see whether there are clearly distinct subclasses of bad mobility, and what prevents mobility in these classes.

The input for a clustering method is a set of data vectors, where each data vector is a multidimensional set of data elements. In the case of suitability problems, each data vector represents the input data in a given geographic location. The data vectors are compared for similarity, and similar vectors are combined into clusters. Similarity in clustering is

measured using a function, which takes two data vectors as input and returns a similarity value for them. The most common, and most basic, similarity measure is simple Euclidean distance between the data vectors in attribute space. In many cases, however, other measures can be more useful, for example when the data has numerous dimensions. One such distance measure is the Mahalanobis distance, which takes into account the correlation between input data dimensions. The correlation is given by the covariance matrix of the data set [72].

Clustering, as well as many related *data mining* methods, are data-driven analysis methods, where new information rises from the data [43]. This allows the analyst to find new points of view and gain insight into data that might not otherwise be found. For example, when combining data from several input layers into one suitability score, data mining methods can help in discovering how the different input layers interact, and to discover unusual, or unexpected, divisions in the source data. This can lead, for example, to the discovery of areas where assigning suitability would require more detailed analysis or additional information.

Thus, data-driven methods can give more information to the analyst than a direct suitability categorization would. Therefore, the results can be of more use in further analysis and decision making than the direct suitability scores, which do not allow such information discovery. Furthermore, in many cases it could be hard for the analyst to decide the best criteria for different suitability scores without first familiarizing themselves with the data. Data–driven methods can show the analyst possible criteria by showing how the data can be arranged.

## 6.2 Requirements of the Process

Several requirements of the data analysis process were derived from the background and motivation of this research. The purpose of these requirements was to guide the development of the analysis process and a prototype implementation.

The first requirement was for the analysis process to be *free of inherent values*, and employ only well-known and general visualizations and computational methods. This makes the process to be usable by several actors, who may have incompatible goals, values, or knowledge they wish to use when doing their own analyses. Furthermore, it enables the methods to be used to solve several different problems.

Thus, the methods selected must not contain values or knowledge specific to any organization, or for solving any specific problem. However, it is impossible to solve a spatial problem without using any values or knowledge in the process. Thus, the user needs to explicitly, and often also implicitly, insert the required knowledge and values to the analysis process. Explicit insertion of knowledge into the process also makes it possible to keep track of the values or knowledge used while solving a certain problem. For example, when analyzing mobility, the analyst needs to insert the knowledge of how each input layer affects mobility into the process.

The second requirement was for the analysis process to be *transparent*. This requirement may have two different meanings. First, the analysis process in general can be transparent, so that any user or organization can examine the process and make sure there are no hidden values or knowledge in the process. Second, the analysis of a specific problem can be transparent so that other users can review the analysis process and see what values and knowledge have been inserted in order to arrive at the given conclusion. A transparent process can later be reviewed by different actors in order to ascertain whether a specific analysis result is usable for them.

This work has facilitated the first meaning of transparent by using well-known and general visualizations and computational methods. Furthermore, in the prototype application, these methods have been implemented independently, or by using open source software in the implementation whenever possible. Unfortunately, outside considerations forced the use of the closed source ESRI ArcEngine platform as the GIS framework.

The third requirement was for the analysis process to be *general*, so that it can be used to solve a wide range of problems. This research has concentrated on suitability problems, where an area needs to be divided into categories according to their suitability for certain activity. Numerous problems belong to this category, since selecting the best position for something is an important and common GIS problem. The approach used in this work is exploratory analysis combined with visual analytics, where interactive visualizations are used in order to examine the data and draw conclusions from it. Thus, the analysis process can be used to solve a wide range of problems, and is therefore general.

The prototype application constructed as a part of this research is aimed at solving the problem of cross-country mobility. Due to time and resource

considerations it uses a limited number of visualizations and computational methods. It is, however, capable of accepting a wide range of inputs.

Finally, there is one requirement for the process that did not arise directly from the needs of international crisis management, but from the goals of this research and conscious decisions. The focus of this research is the development of analysis processes, not methods development. Therefore, basic, *well–known techniques and methods* are sufficient for this research. This way it is possible to ascertain that the methods employed in the process – the visualizations, the interaction methods, and the computational analysis methods – are effective. Thus, there is no need to spend time and effort on establishing whether the individual methods are of use. The focus of the work is on analyzing the process itself.

# 7. Related Work

## 7.1 Exploratory Analysis and Visual Analytics

The idea of *exploratory data analysis* (EDA) was first proposed by John Tukey to complement *confirmatory data analysis*, which is concerned with testing hypotheses or assumptions about the data [116]. According to Tukey, EDA is not a set of specific tools or processes, but more an attitude or philosophy about how data should be analyzed. The goal of EDA is to explore data, to detect and describe patterns, trends or relationships in the data, and to summarize them in an easy-to-understand form. The process can be used, for example, to formulate new hypotheses about data, or to support the selection of appropriate statistical methods and tools. It is therefore not a replacement for confirmatory data analysis. Instead, it represents a completely different approach. Other authors have later expanded EDA to cover different types of analysis and different data set types. For example, Andrienko and Andrienko have written on how exploratory analysis can be applied to spatial data sets [9].

EDA has also influenced the development of other data analysis methods, such as *visual analytics* (VA). Visual analytics is "a method of reasoning facilitated by the use of interactive visualizations" [113]. It is a multidisciplinary field that focuses on analytical reasoning techniques, visual representations and interaction techniques, data representations and transformations, and techniques that support the production, presentation and dissemination of analysis results. Visual analytics tools and techniques can be used to "synthesize information and derive insight from massive, dynamic, ambiguous, and often conflicting data" [113]. In essence, it is a set of visualization and interaction tools and techniques meant to assist analyzing large datasets. Thus, the field is related to

both *information visualization* and *scientific visualization*, two interrelated fields of science studying the use of visualizations for representing large datasets. There is still, however, debate on the definition of the term, and how it relates to other fields of science; a definitive consensus is yet to be reached [5].

Exploratory data analysis methods and interactive visualizations are often used together for problem solving. Thus, in many cases it is possible to argue that the process depicted is both an example of EDA and an example of visual analytics. Examples of cases where exploratory and visual approaches have been used to analyze spatial data include ecological network planning [65], vehicle safety [78], forestry [6], public health [97], and eye movements [23]. There has also been a number of works that study the use of exploratory analysis and interactive visualizations. The topics of these papers range from combining exploratory visualizations with data mining [16] to the fundamental problems of exploratory analysis [37], from evaluating exploratory and visualization methods [63] to clustering methods [79] etc. However, it seems that there are still numerous challenges in using visual analysis and exploratory methods with geodata [7].

## 7.2 Visualization

Visualization is the act of representing something using images or illustrations. *Information visualization* is the study of how to visualize complex large-scale collections of information in an understandable manner. The goal of information visualization is to create visualizations, from which the viewer can gain additional value and information. There are numerous different visual metaphors that can be used to depict data. Most methods work for hundreds or thousands of data elements, while some techniques can scale up to millions of elements. The best method for a given situation depends on the size and type of the data, and the goal of the analysis process. Furthermore, in many cases multiple information visualization methods can be used simultaneously or in succession. The visual information seeking mantra states "overview first, zoom and filter, then details-on-demand" [103]. In cases where there are numerous data elements, the user might want to use different visualizations for showing the overview and for the details. If, for example, the whole set is hundreds of thousands of elements, then a different view could be more useful there

than when examining the details of a dozen data elements.

Information visualization techniques are required for the visualization of *multivariate data*, where each data element consists of several attribute values. In case of *spatial* multivariate data, each data element also has a geographic location associated with it. Thus, the data has at least two spatial dimensions and a number of attribute dimensions. In such case, the spatial distribution of the data can be visualized by using a map, and the attribute distribution by using multivariate data visualization methods.

There are numerous visualizations that can be used to represent large, multivariate datasets in an understandable manner. The different visualizations all have advantages and disadvantages, and therefore it depends on the situation, which method is the best one to use. Solutions include scatterplot matrices [4], star plots [21], glyphs such as Cernoff faces [22], reordeable matrices [15], parallel coordinate plots [52], and many others.

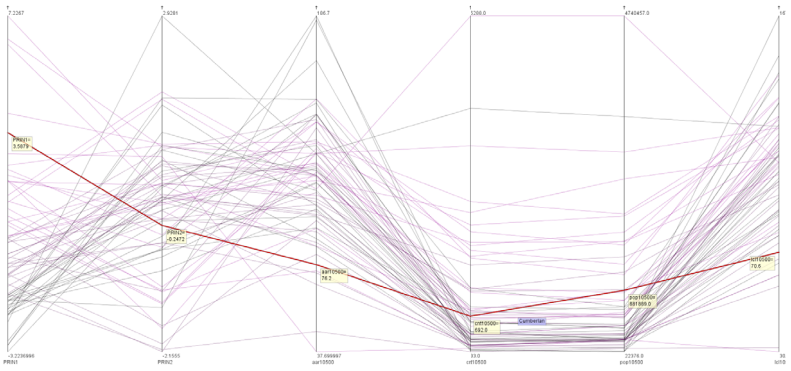

**Figure 7.1.** A parallel coordinate plot example

Parallel coordinate plot (PCP), shown in Figure 7.1, was selected as the primary information visualization method used in this work. PCP is a multivariate visualization method [52]. It displays $n$–dimensional data in two dimensions using $n$ parallel axes. The axes are arranged either horizontally or vertically, and data elements are visualized as polylines that traverse through these axes. A polyline representing a particular data element intersects each axis at a point that represents the data element's value in the dimension represented by the axis. PCP has been found to be a very useful tool, which works well in combination with maps [30]. The main limitation of PCP is that its effectiveness is limited to situations where less than 1000 data elements need to be shown on the screen simultaneously [57]. When more elements need to be shown simultane-

ously, other visualization methods are typically more effective. Some techniques, such as dense pixel displays, can visualize up to millions of data elements [56].

### 7.3 Spatial Analysis Methods

Spatial analysis is the formal study of spatial data. The term can have multiple meanings, depending on the context. O'Sullivan and Unwin give four different contexts: *spatial data manipulation*, *spatial data analysis*, *spatial statistical analysis*, and *spatial modeling* [87]. Of these four contexts, spatial data analysis corresponds to exploratory analysis of spatial data. It is in contrast with statistical analysis of spatial data, in which various statistical methods are used to test hypotheses and see whether a given data set conforms to a statistical model, and spatial modeling, in which models are constructed to predict spatial outcomes. In exploratory analysis, a number of different methods can be employed. The analyst can use various visualizations to explore the data set. The analyst can also use computational methods to manipulate the data, in order to see new aspects or find new information.

One computational method that can be used in EDA is data mining. It is the process of discovering patterns or relationships in large data sets [43]. One often–used data mining technique is clustering, which is the process of dividing a set of data elements into subsets, where elements in the same subset are similar to each other, and elements in different subsets are distinct from each other. There are numerous clustering algorithms that can be divided into different models, including: hierarchical clustering methods, where clusters are created by combining or breaking up existing clusters; distance–based methods, where data elements that are closest to a given cluster center point belong to the same cluster; and density–based models, where elements that create sufficiently dense areas in the data space belong to the same cluster. In this work, two clustering methods were used: k–means clustering and DBSCAN.

K–means is a well–known and widely used clustering method originally proposed in the 1960s [71]. K–means divides a set of data elements into $k$ clusters, where the number of clusters needs to be given beforehand. The clusters in k–means are defined by cluster centers, which are calculated iteratively: the center of each cluster is the mean of its elements, and each data element is assigned to the cluster with the nearest mean. K–

means has been applied to numerous problems and has a large number of improvements and variations. For details, see for example [14]. In the context of geoinformatics, k–means has been used for finding locations for facilities [69], landslide hazard prediction [41], and analyzing space–time paths [106].

DBSCAN, or Density–Based Spatial Clustering of Applications with Noise, is also widely used [33]. DBSCAN creates clusters from data elements that constitute sufficiently dense regions in the data space. In DBSCAN, density is defined using two parameters: distance value $\epsilon$ and minimum number of neighbors $p$. If a point has at least $p$ other points closer to it than $\epsilon$, the point, and all its neighbors, belong to a cluster. Data elements that do not have $p$ other elements inside the $epsilon$–distance, and are not inside the $\epsilon$–distance of any point belonging to a cluster, are marked as noise. DBSCAN has been used for finding clusters in road networks [111] and earthquakes [91]. Also, the algorithm has variations. One example is GDBSCAN, which generalizes the concepts of density and neighborhood [102].

## 7.4 Related Systems

GIS systems, whether commercial products such as ArcGIS, or free software products such as GRASS, traditionally have had rather limited support for exploratory analysis or visual analytics. A typical GIS system is more focused on the visual representation of maps and traditional confirmatory data analysis. Thus researchers interested in using EDA or VA methods typically need to either implement their own systems from scratch, or build them on top of existing GIS framework. Therefore, a number of EDA and VA systems that explicitly support spatial data have been implemented in academic institutions.

### 7.4.1 GeoVISTA

GeoVISTA is currently an active research project at Pennsylvania State University [2]. The project has developed a number of visualization tools for GIS. One is the GeoVISTA Studio software framework [112] and a number of software projects derived from it, such as the GeoViz Toolkit [44].

GeoVISTA Studio software framework allows users to create a wide range of different data analysis processes via a graphical programming

environment. GeoVISTA contains a wide range of visualizations and computational analysis methods that can be combined into new analysis processes in a custom visual programming environment. The system is implemented using Java and JavaBeans, and new functionality can be added to it by adding new beans. The system is general and flexible, but by our own experience it has a rather steep learning curve. In the beginning, a new user can a have hard time getting started, because there is so much functionality presented. The system is developed as free software and uses the GNU Lesser General Public License. Last public release of the GeoVISTA Studio system was in September 2007. The system has apparently been developed further, even if there have been no further releases of the framework. This is demonstrated by, for example, the recent paper describing the GeoViz Toolkit developed using the system [44].

GeoViz Toolkit is a visual analytics toolkit based on GeoVISTA Studio. The system contains a large number of different visualizations that can be used to view the data at hand. The views are interactive and linked together, and thus a change done in one of the views is reflected in all other views. An example of GeoViz in action is shown in Figure 7.2. The Figure shows, from left to right, the attribute selection window, star plot window, map window, and PCP window, with animator window and table view window below them. One element is highlighted in all data windows.

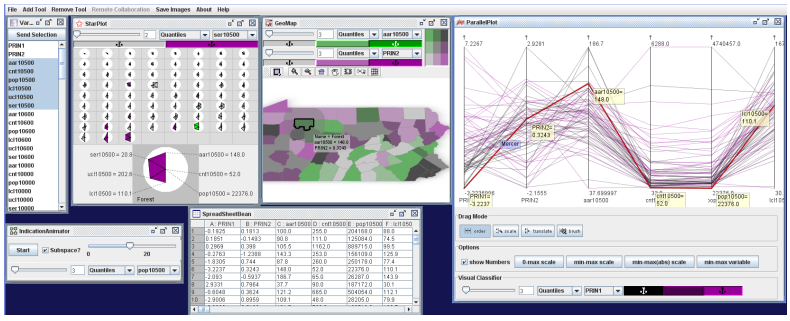

**Figure 7.2.** View of the GeoViz Toolkit containing a number of visualization windows.

The GeoVISTA Studio is a mature, robust, and versatile software package. A large part of the functionality required for this work has been implemented in GeoVISTA. Furthermore, the system has been published as free software, and thus should be easy to take into use and to modify.

### 7.4.2 CommonGIS

CommonGIS is a java–based software for visual analysis of spatial and spatiotemporal data. The system has been developed at the Fraunhofer Institute for Intelligent Analysis and Information Systems [1].

The system offers a large number of different information visualization views that can be combined with a linked map view for visual analysis of the data sets. The system can also calculate a number of analyses and statistics from the data. The system can also handle time–series data, and has visualizations specific for such datasets. CommonGIS is implemented in Java, and can be used both as a java application or as an applet on a web page. The CommonGIS software is based on previous systems named IRIS and Descartes [3], and the initial development on the system was done around the turn of the millennium [8].

The main window of the CommonGIS system resembles the main view offered by several desktop GIS applications. This window is shown in the top-right corner of Figure 7.3. On the left–hand side of the map window is a view showing all data layers that have been loaded. The layers can either be shown or hidden on the map. On the right–hand side of the map window is a view that can be used to fine–tune the layer visualizations. On the top of the main window are menus, from which the user can call functionality, such as the calculation of statistics.

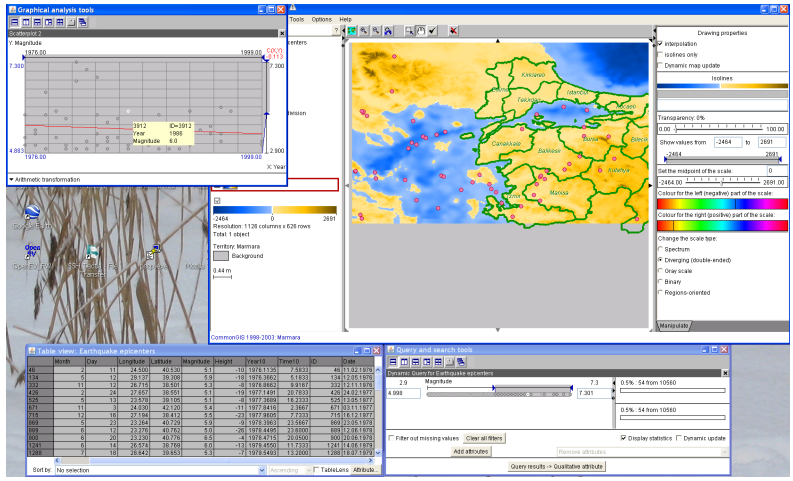

**Figure 7.3.** View of CommonGIS containing a number of visualization windows.

Figure 7.3 shows several CommonGIS windows representing earthquakes around the Marmara region of Turkey. On the top left is a scatterplot matrix showing the year and the magnitude of each earthquake. Only

quakes of magnitude 5 or larger are shown. Next to the scatterplot is the map view showing the location of each earthquake. Bottom left is the table view of the earthquake data, and next to it is the dynamic query window used to filter out the smaller earthquakes.

CommonGIS is a versatile software package that offers a lot of GIS functionality common to many desktop programs, as well as a large number of information visualization views. The software is available without a cost for educational institutions.

## 7.5 The ArcGIS Environment

The ArcGIS platform is a family of commercial GIS products developed by ESRI. The system was initially released in 1982 as ARC/INFO, and the graphical desktop version in 1995 as ArcView. The number of included software products was subsequently expanded, and these days the ArcGIS software family includes a large number of GIS software. Included are, for example, the desktop GIS environment ArcMap, data management software ArcCatalog, and software development environment ArcObjects.

The ArcGIS desktop environment, ArcMap, is a flexible software product, which can be used both for map construction and spatial analysis. The spatial analysis functionality in the software is included in the Spatial Analyst extension to the software. The extension contains a large number of different analysis tools that range from the specific, such as calculating a slope layer from an elevation layer, to the generic, such as the map algebra tool.

The focus of the analysis tools in the ArcMap environment is on traditional, confirmatory spatial analysis, although several tools can be used as part of an exploratory process. There is, however, little support for exploratory analysis of spatial data, and especially the support for different types of information visualization is limited. Typically, a spatial data layer in ArcMap can be visualized as a map, as a table of values, and as a histogram or a scatterplot. The different views are linked.

Figure 7.4 shows the ArcMap software with several visualizations representing the slope degree around the city of Lahti in Finland. Different degrees of slope are shown using different shades of blue. Flat area is shown in white, and extreme slopes in red. In addition to the map view, the data is also visualized using the table and the histogram views. On the left side of the screen is the view used to control data layers, and on

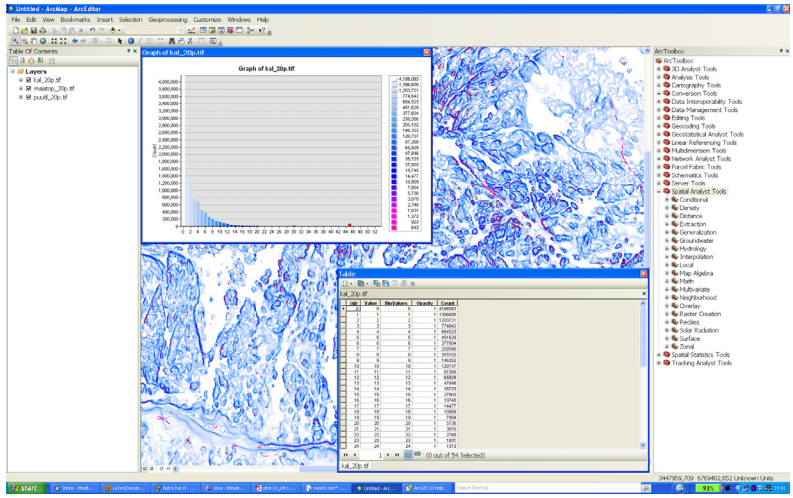

**Figure 7.4.** View of ArcGIS containing a map, table and histogram views of the data.

the right side are the different tools available for the user.

ArcGIS also includes a software development environment called ArcObjects. The environment can be used to implement new functionality or new applications using the ArcGIS functionality. The functionality available to ArcMap or ArcCatalog through GUI is also available through ArcObjects. The environment is available in .NET, Java, C++, and Python languages.

*ArcGIS in This Project*

In this project two ArcGIS products, the ArcMap desktop environment and the ArcObjects programming framework were used. The ArcGIS platform was selected as the implementation platform since it was the sole GIS system used by the Finnish Defense Forces, who funded the initial research on this topic. Thus adopting ArcGIS made it easier to share the ideas and concepts developed in this project. Because ArcGIS had been used at the Department for several years, our research group had extensive experience with the environment. There was also concerns about the amount of documentation and support available for the established research prototypes. When the project was started, the research prototypes were still mainly used in the institute that implemented the system. Thus, it was decided that using an established, commercial system was a better choice, since support was available on demand.

It was clear, that research prototypes such as GeoVISTA would have provided more exploratory and visual analytics functionality than ArcGIS. However, in this case other considerations won over the provided func-

tionality, and thus it was decided that ArcGIS was the platform of choice for this research.

# 8. Analysis Process

There are numerous ways to analyze spatial data. This work concentrates on exploratory methods, combined with visual analytics and clustering for analyzing suitability problems. Off-road mobility analysis is used as the main case example. The process developed in this project was first described in Publication V. It was elaborated further, and results on using the process were described in Publication VI. Publication VI also includes a detailed example of how the process can be used to solve the cross–country mobility problem.

The data used in this study is assumed to contain no spatial dependencies, and thus can be analyzed without taking spatial autocorrelation into account. This means that a value at a given location can be measured without having to take into account the location's neighborhood. Examples of such phenomena are soil type, amount of vegetation, amount of rainfall, elevation, etc. Examples of phenomena that are dependent of the neighborhood are the viewshed, the catchment, and the distance to the nearest road.

Spatially dependent phenomena can also be used in the analysis discussed here, if the phenomenon can be transformed into a format where it can be expressed as a simple attribute value in each location. For example, the distance to the nearest road or the amount of catchment can be calculated for each position and expressed as a number. In this form these phenomena can be considered spatially independent as long as the data does not change. A viewshed, on the other hand, cannot be a simple attribute value. The output of a viewshed analysis is a map that shows all locations visible from the given location.

The spatial analysis process described in this work is based on the concepts of exploratory analysis and visual analytics. This means that the user controls the process and therefore needs to have quite a lot of expert

knowledge about the situation at hand. Furthermore, the process itself does not contain any values or knowledge, and these need to be inserted by the user during the analysis. Thus, the user's expert knowledge about the problem at hand, the input data used, and the analysis process are vital. Several times during the process the user's knowledge is either used to control the process or inserted into the process in other ways.

The expert knowledge required in this kind of process can be divided into two categories. *Domain knowledge* is knowledge about the problem at hand and the factors that affect that problem; *GIS knowledge* is knowledge about geoinformatics and GIS, and how GIS can be used in problem solving [65]. Both types of knowledge are required for solving spatial analysis problems. In cases where no single person has sufficient knowledge both about the domain in question and the GIS techniques, two users can work together in order to solve the problem. In such cases, a domain expert handles the tasks that require domain knowledge, and a GIS expert handles the GIS tasks.

## 8.1 The Spatial Analysis Process

Overview of the spatial analysis process described in this work can be found in Figure 8.1. The boxes in the figure stand for different phases of the analysis process. The process moves top-down t and consists of four main phases, labeled from A to D in the figure. Inside each phase are a number of smaller sub-phases. Orange arrows represent steps forward in the process, and blue arrows represent steps backwards. The backwards arrows represent iterative steps where the process is continued from an earlier phase. Each backwards arrow also contains a short explanation for the reason for backtracking. It should be noted that there are no backwards arrows from phase C, since computational data analysis always requires interpretation before the user can decide whether they need to backtrack. Output interpretation is in phase D of the process, and it is the first instance where it is possible to backtrack, once phase C has been entered.

The analysis process consists of four phases: *A: Preparation for the Analysis*, *B: Visual Data Exploration*, *C: Computational Data Analysis* and *D: Interpretation of Output*. In Publication VI phase A was called "analysis design" and phase B "Data Exploration and Preparation".

In phase A, the analyst gathers input data layers for the analysis, and

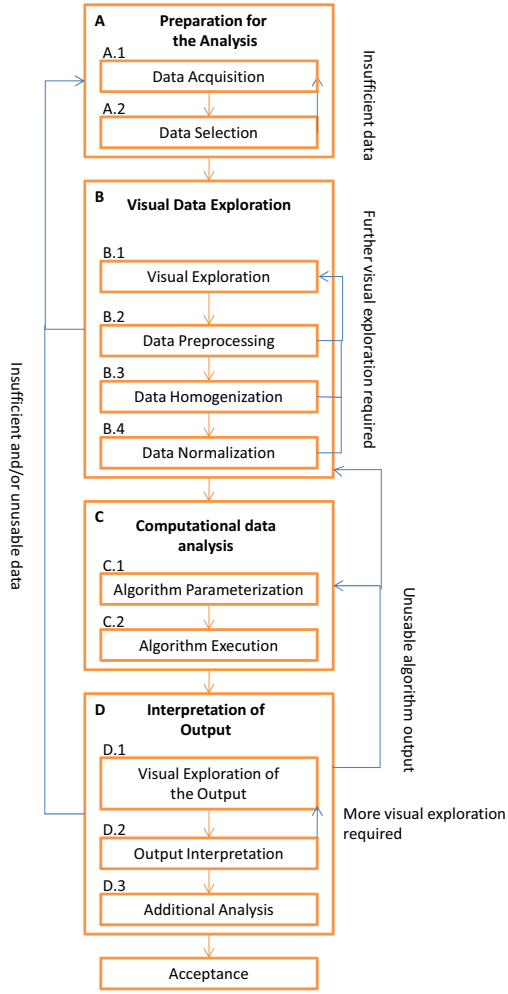


**Figure 8.1.** Overview of the spatial analysis process. Figure originally published in Publication VI.

designs how each layer is used during the process. In phase B, the analyst investigates the data further, and processes it in order to make the different input data layers comparable. In this phase the analyst also converts the data into a format that can be given to a computational data analysis method. Phase C parameterizes and executes the method, and its output is interpreted and further analyzed in phase D. If the output of phase D is acceptable as an analysis result, the process terminates; otherwise the analyst goes back to a previous phase to continue the process. The details of the process are first described in Publication V and further elaborated in Publication VI.

### 8.1.1 Iterative work

Typically, a spatial analysis process does not proceed smoothly from one phase to the next. Instead, during the process various problems and mistakes are discovered, such as having insufficient or unusable data in some phase of the process. When this happens, the user needs to go back to a previous phase of the process. Often this iteration needs to be done several times before an acceptable solution can be found. Figure 8.1 shows this using blue backwards arrows.

Each of the arrows is labeled with the most probable reason for backtracking. If an arrow starts in a given phase, it means that it is possible for the backtracking to start from any sub-phase. If an arrow ends at a given phase, it means that it is possible for the backtracking to end in any sub-phase. Arrows starting or ending in a particular sub-phase mean that backtracking starts or ends at that particular sub-phase.

On the bottom of the figure are arrows that backtrack from phases B and D of the process to phase A. These depict situations where the user discovers that the input data used for the analysis is either insufficient for the analysis, or otherwise unusable for some reason. For example, the attribute data format in an input layer may be such that it cannot be used for the intended purpose. For example, a vegetation layer that shows different types of vegetation may be unusable, if an analysis requires the amount of vegetation to be shown.

On the top of the figure are arrows that backtrack to and from specific sub-phases of the process. The first arrow from the top goes from phase A.2 to A.1. This depicts a situation where, while selecting and categorizing the data, the user realizes that some required input data layers are still missing, and the gathered data is thus insufficient for the analysis.

The second arrow on the top goes to phase B.1 from the later sub-phases of the Visual Data Exploration phase. This depicts the situation where data preparation for computational analysis requires more visual exploration before the user can make a decision on how to proceed in particular part of the analysis. This also means that it is possible to go from B.1 to any of the subsequent sub-phases inside phase B of the process. These jumps forward are not depicted in the figure in order to simplify it. The jumps also depict that the data preparation is a very interactive task, where the user often needs to use various visualizations to see, how the data layers are arranged, and how the data is spread in a given layer.

The third arrow on the top at the right side of the figure depicts the most common iterative loop of the analysis process. This arrow starts at phase D of the process, and ends in either phase C or phase B. It covers the situation where the output of the computerized analysis method is deemed unusable, and the user returns to a previous part of the process in order to get a new result. In practice, this particular loop often ends in phase C.1, where the user gives the algorithm new, different parameters, and runs it again in order to see the new result. For example, in the k-means clustering the user must give the number of clusters for the algorithm to use. The output is heavily dependent on the number of clusters, and if there are too few clusters the output might be such that all of the clusters contain data elements depicting bad mobility. Thus the user may need to run the algorithm several times before gaining an acceptable output. The other possible end point of this backtrack are the sub-phases of phase B. In this situation, the user needs to modify the input given for the algorithm. For example, the normalization used could be such that it does not differentiate between different types of terrain sufficiently well, and thus there are no discernible differences between clusters.

Finally, there is a backwards arrow going from phase D.2 to D.1, which depicts the fact that the output interpretation is a very interactive task where the user applies various visualizations in order to interpret the algorithm output. In essence, the user examines a cluster or a number of clusters, and then gives these clusters a suitability value. After this, the user examines new clusters in order to discern their suitability.

### 8.1.2 Limitations of the Process

The process described in this work is aimed at solving suitability problems. So far, it has been extensively used to solve the cross-country mobil-

ity problem, and small experiments have been done with other suitability problems, such as suitable locations for artillery pieces and helicopter landing areas. The most common limitation in suitability analysis are the often strict requirements for input data. In the current version of the process described in this work the input must be such that spatial autocorrelation can be ignored, and the characteristics of each location can be expressed by a simple numerical value. Thus, for example, selecting good locations for microwave communication link masts is a difficult problem to solve using this method. The location of each communication link affects the optimal locations of every other link, and two masts must always have a line of sight between them. Thus, when the locations of two masts are selected, the LOS between each two candidate positions must be considered. The process, as it currently stands, cannot support such analysis.

Up until now the topic of each experiment with the analysis process has been a suitability problem. However, it should theoretically be possible to apply the process to any problem, where the effect of each input layer on the problem can be characterized independently of other inputs, the effect of each input layer can be characterized using a simple, numeric value on each location, the output is some kind of combination of the input values at each location, and the neighborhood of each location has no effect on the analysis. The data used in the process must also be of sufficiently good quality. The quality aspects are, however, out of the scope of this work.

Furthermore, in order to use the process, the analyst, or analysts, must be familiar with exploratory analysis techniques and with the GIS problem at hand, as well as the problem being solved. Without a GIS expert and a domain expert at hand, the process cannot be used.

The current prototype application used to test the process in real problem solving situation, described in Chapter 9, currently supports only parts of the process, and contains only limited visualization and computational analysis functionality. It is by no means a general or industry-ready implementation of the process.

# 9. The Prototype Application and Results

In order to validate the analysis process described in Chapter 8, it was applied in a real spatial analysis situation. Tools available in common GIS tool sets such as ArcGIS, MapInfo, or GRASS actually cover large parts of the analysis process. Phases A, B, and D.3 (additional analysis) can typically be accomplished by the functionality already included in a good GIS system. Still, current GIS software does not typically include clustering algorithms or visualizations required for interpreting the clustering result. Thus, existing GIS software were not sufficient for the needs of this research.

As described in Chapter 7, current GIS software systems that incorporate comprehensive visual analytics or data mining tools tend to be research prototypes. Some of them, like GeoVista are quite mature, while others tend to be rather limited. Another consideration was the fact that this research was done in cooperation with, and partially funded by, the Finnish Defense Forces. The FDF had, before the start of the project, decided to adopt ArcGIS as their exclusive GIS platform. Because of this ArcGIS was adopted as the sole GIS platform of this project and a prototype application was built on top of ArcObjects framework. The prototype application covers phases B.4 (data normalization) through D.2 (output interpretation) of the analysis process, and contains three data analysis methods.

There were several reasons for including these phases and only these phases in the prototype. The by far most important reason was the the prototype described here is a research prototype designed to verify the usefulness of the approach described in Chapter 8 and act as a reference for future development. It was not meant to act as a basis for the development of a complete software product. Thus, since the ArcGIS desktop included the functionality required for the other phases of the process,

replicating it in a prototype would be unnecessary extra work. Also, since the functionality in phases C.1 through D.2 is not included in ArcMap, these phases needed to be implemented. Since the results of a clustering are highly dependent of the clustering algorithm used, its parameterization, and the input data, one hypothesis was that iteration between phases B.4 and D.2 is the most common iteration in the process. Thus, while phase B.4 could also be done with the functionality offered by ArcMap, this phase was included in the prototype.

Three data analysis methods were included in the prototype. Two of the methods were clustering methods: k-means and DBSCAN. Both clustering methods use euclidean distance between data vectors as the distance measure. The output of these methods is a number of clusters that have a spatial distribution and a data space distribution. These can be visualized using a map for the spatial distribution and a parallel coordinates plot for the data space distribution. The third analysis method was map algebra local multiplication, where the values are normalized to an interval from 0 to 100. The output of this method is a raster map depicting the different multiplication results. It can be visualized using a map for the raster image and a histogram for the data values.

The prototype application has been implemented using Java, the ArcObjects framework for Java, and the JCharts and JFreeChart diagram libraries for creating the information visualization views. Map algebra was provided by ArcObjects, and k-means and DBSCAN were implemented using common pseudocode and algorithm descriptions as references.

## 9.1 Prototype User Interface

The prototype consists of two user interface windows, the main window and output interpretation window. The main window, shown in Figure 9.1, is opened as the prototype is launched. The visualizations in the main window are standard ArcGIS functionality. On the left-hand side is a frame listing all open input data layers, and on the right-hand side is a map frame that visualizes the currently selected input layers.

In the main window, the user can select the input layers for the computerized data analysis, normalize the layers, and parameterize and run analysis methods. Analysis methods can be run only for layers that have been normalized. The results of an analysis are visualized in the output interpretation window, which is opened automatically once the analysis is

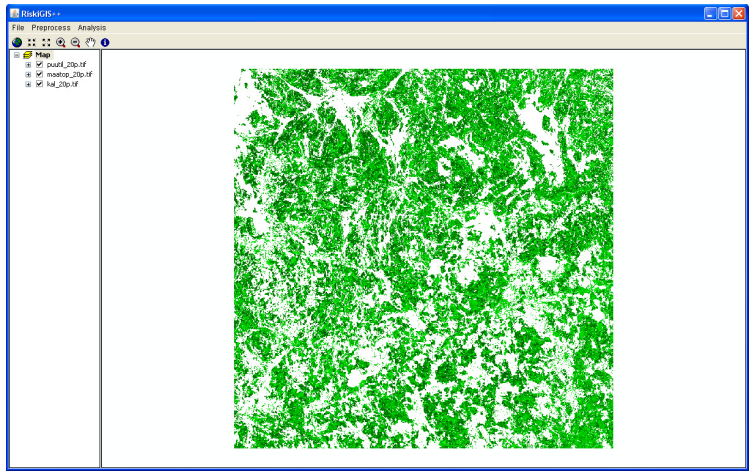


**Figure 9.1.** Main Window of the Prototype.

ready.

Only the functionality that is absolutely essential for the analysis process is included in the main window. Therefore, there are no means in the prototype for using information visualization views to further explore the details of the input data layers, and the means of how to apply, for example, the normalization of an input layer is rather crude. Since the goal was not to create a polished software product, but to validate a concept, this was deemed sufficient for the purposes of this research.

Figure 9.2 shows the output interpretation window for k-means clustering. At the top of the window are tools for manipulating the map view. Below the map tools the window is divided into four frames. In the top row, on the left, is a map visualization frame, and on the right, an information visualization frame showing the parallel coordinates plot. The map visualization frame shows different clusters using separate colors. The user can pan, zoom, and select clusters or individual pixels from the map for highlighting or more detailed analysis. The information visualization frame contains a PCP, which has four axes that stand for slope, soil type, amount of vegetation, and cluster number. The user can select clusters from the PCP visualization for highlighting. One of the clusters is highlighted in both views. Each time the highlighted cluster is changed, this change is reflected in both views.

In the bottom row, on the left, is the output interpretation frame, and on the right a frame showing the cluster details. The output interpretation frame contains functionality for assigning suitability values to the clusters. The cluster details frame shows the centroid of each cluster and all

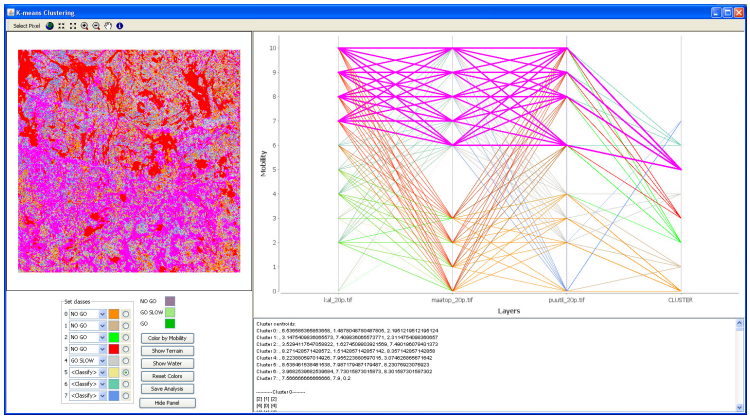


**Figure 9.2.** K-means clustering interpretation window of the Prototype.

data vectors belonging to each cluster. There is no special functionality in this frame.

Using the output interpretation window the user can view the details of each cluster. They can see how the cluster is distributed on the map, how it is distributed in the data space using the PCP, and view the data vectors belonging to the cluster. Using this information the user can assign each cluster a mobility value. Once all clusters have been given a mobility value, the user can click the 'Color by mobility button' to see the color map, and can save the data using 'Save analysis' button.

The functionality provided by some of the visualization views is rather limited. The PCP view, for example, is missing some common functionality, such as the ability to modify or rearrange the axes. When the prototype was built, it was made for solving the cross-country mobility problem. It was assumed that each input layer is normalized the same way, and thus each dimension should be visualized the same way. Furthermore, since the number of input layers used in mobility analysis is typically rather small, it was assumed that rearrangement of the axes would not be a very important feature to include. It is, after all, typically used in order to be able to examine specific dimensions of a large-dimensional data set. Such bare-bones functionality was thought to be sufficient for the purposes of this research.

The frames in the output interpretation window are fully customized according to the analysis method used in a particular situation. Figure 9.3 shows the output interpretation window for map algebra local multiplication. Instead of a parallel coordinates plot, there is a histogram that shows how many pixels have a given mobility value. In the interpreta-

tion frame the user can set the lower limit for each of the three mobility classes, and divide the area that way. The map view and its interaction methods are unchanged from the interpretation window for clustering, except for the different color scheme for the map. This is due to the way ArcGIS assigns map colors. The user can interact with the histogram by selecting one or more bars. These bars and the corresponding areas on the map are then highlighted.

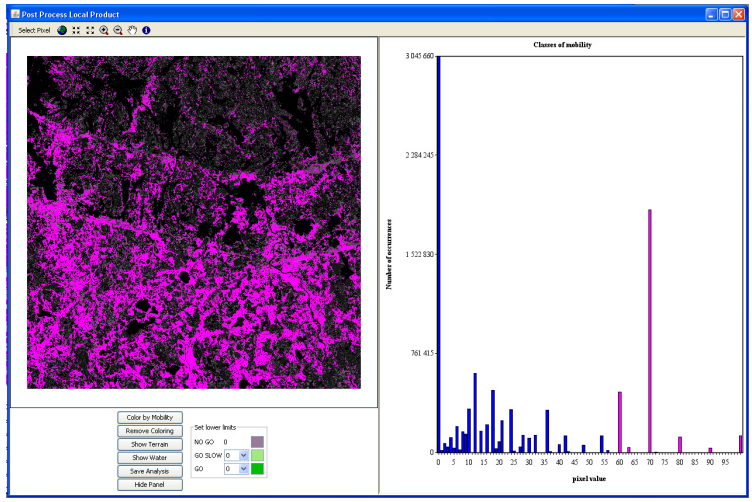


**Figure 9.3.** Local product interpretation window of the Prototype.

The histogram visualization was selected as the information visualization view used for the interpretation of the local multiplication due to the fact that the output of the local multiplication method gives each pixel a numerical value representing its mobility compared to other pixels. Thus, a histogram view of the output map is a natural way to represent this data, and shows to the user how many pixels of a certain value there are, and if there are, for example, natural breaks in the data that could indicate separation between different classes of mobility.

In a typical use case, the user starts the analysis using a normal desktop GIS such as ArcMap. When entering phase B.4 of the process, the analysis moves to the prototype application, where the analysis continues until phase D.2 has been completed. While testing the system it was noticed that in a typical analysis scenario, there are several iterations between phases C and D before an acceptable output is found. After this, the rest of the analysis process can be done using the selected desktop GIS program. In a real application, the functionality of the prototype should be integrated into the desktop program. For the purposes of this research, a separate prototype was deemed sufficient.

## 9.2 Experimental Verification

The the usefulness of the prototype was tested by applying it to off-road mobility analysis in Finland. Two experiments were conducted using two data sets. In the first experiment the goal was to ascertain that the approach can be used for spatial analysis, and that the results are useful. This was done by trying to replicate the model–based mobility analysis done by the Finnish Defense Forces by using exploratory techniques and visualization. The results were verified by expert evaluation and by comparison to the FDF mobility model. This experiment used the data sets originally produced by FDF for their mobility analysis. The details of this experiment and its results have been published in Publication VI.

In the second experiment the goal was to ascertain if it is possible to do the entire analysis process, starting from input data acquisition, sufficiently fast to support the decision making. The data for this experiment was gathered independently from available sources. The details of this experiment have not been published previously.

The vehicle for which the mobility was created was the same in both instances: the Patria Pasi armored personnel carrier. The Pasi was selected as the example vehicle, because the full metadata used in the creation of the FDF mobility model for this vehicle type was available.

### 9.2.1 First Experiment: Feasibility of the Analysis Results

The results of the first experiment were reported in Publication VI, and the following is only an overview of the experiment.

The input data used in the first experiment was soil type, amount of vegetation, slope data, road data, and building data. Soil type, vegetation, and slope layers were used throughout the analysis, while road and building data were integrated to the solution in the additional analysis phase. In this experiment the vehicle mobility was divided into three categories: GO, GO SLOW, and NO GO. The FDF model divided mobility into seven categories. Thus, for comparison, the categories in the FDF model were generalized into three categories in order to make the two results comparable. The input data and data normalization used were the same as what had been used in building the FDF model. The data normaliza-

tion was between 0 and 1 with one-digit precision, where zero depicted no mobility and 1 perfect mobility.

The experts in FDF suggested the use of three categories in the experiment. The categorization comes originally from NATO usage, and one of the goals of the FDF was to create mobility maps with this categorization. The generalization of the existing FDF mobility model was done by using expert knowledge. The most important expert knowledge was the understanding of how the FDF model-based approach works, and from this deriving an acceptable generalization of existing seven categories.

The area used in the experiment was a part of central Finland that had both wilderness and urban areas. The wilderness in the experiment area was divided into two clearly distinct subsections by the Salpausselkä ridge system. The area was 80km X 80km in size, and pixel size was 25m X 25m, making the raster images used in the analysis 3200 X 3200 pixels in size. The analysis was done several times using both k–means and DBSCAN clusterings. In order to simulate quick, ad–hoc analysis with limited information available for the analyst, all roads were categorized as good mobility (GO) and urban areas were categorized as no mobility (NO GO). Where the two areas overlapped, roads were given precedence. It should be noted that in reality datasets had sufficient information for more detailed categorization, and the urban dataset covered, for example, airplane runways, which offer good mobility.

The output of the analysis was a number of k–means and DBSCAN clustering results. Eventually three results were deemed the most promising and were selected for comparison with the FDF data. Two of the results were gained by using k–means, and one by using DBSCAN. Further evaluation revealed that the k–means results were variations of the same k–mean clustering result, with the values of two clusters reversed. The two results had been gained using the same input data and the same value of $k$, which lead to identical algorithm output, since the implementation of k–means used in this work is deterministic. Thus, only the one, which better conformed to the FDF analysis result, was described in Publication VI. The two analyses were made on different days, and the k–means output had been interpreted differently. In the following these are referred to as k–means1 and k–means2 outputs.

Detailed results of the experiment can be found in Publication VI. There, the results have been analyzed using expert evaluation and a misclassification matrix. The results of the experiment indicate that exploratory

analysis and visual analytics can be used to arrive at a result similar to the one gained by using traditional modeling techniques. The misclassification matrix analysis showed that the k-means output was a much better match for the FDF model than the DBSCAN output.

The area in the DBSCAN output, which was labeled GO, was over twice the GO area in the FDF model. The DBSCAN output covered most of the GO area in the model, but also included most of the model's GO SLOW area, as well as considerable amounts of NO GO area. In comparison, the corresponding area in the k-means output covered most of the GO area in the model, as well as some of the model's GO SLOW areas, with no NO GO areas covered.

The likely reason for the result of the DBSCAN is in the distribution of the data elements in the attribute space. Since DBSCAN is a density-based clustering algorithm, all data element that are sufficiently densely packed are put in the same cluster. In the experimental data, one such cluster contained all data elements that depict good mobility, and also contained data elements, where the slope degree prevented good mobility. This can be explained by the fact that the experimental terrain contains rather steep changes in elevation in places where the other input data values do not change (the soil type and amount of vegetation stay the same). The k–means clustering, which is distance-based, does not suffer from the same problem with this data. One of the cluster centers was in the attribute space location which depicted very good mobility, and thus one cluster included most of the good mobility data elements.

*Comparison of the Two K-means Results*

The following discussion will concentrate on clusters k–means1 and k–means2, and the differences between the two, as these were left out of Publication VI due to space constraints.

The k–means1 interpretation was similar to the FDF mobility map, while k–means2 had significant differences. Specifically, k–means2 had GO SLOW areas that were different from those in the FDF map. GO areas in the two interpretations were identical.

A further comparison of the two interpretations revealed that the difference between the two was in two clusters, referred in this discussion as cluster A and cluster B. In k–means1 result cluster A was labeled GO SLOW and cluster B as NO GO; in k–means2 A was NO GO and B was GO SLOW. Figure 9.4 shows a PCP visualization of the two clusters. Cluster

A is on the left, and cluster B on the right in the figure.

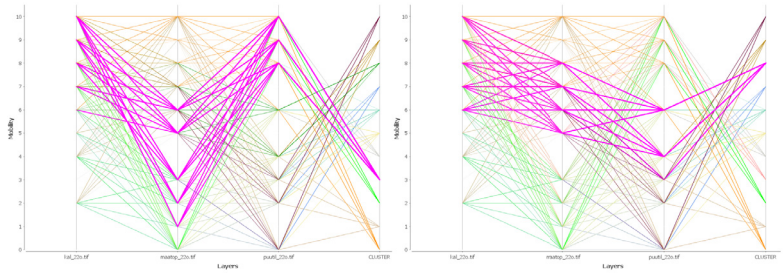

**Figure 9.4.** Comparison of clusters A and B in k–means interpretations using the PCP visualization on normalized cluster input data. Cluster A is on the left and cluster B on the right. The two clusters represent potential GO SLOW areas. The data axes are, form left to right, slope, soil type, vegetation, and cluster number.

As can be seen from the figure, cluster A has rather good values on slope (>= 0,6) and vegetation (>= 0,8), while soil type values are between 0,1 and 0,6. Cluster B has mostly good values on slope (>= 0,6) and fair values on soil type (between 0,5 and 0,8). Vegetation, on the other hand, has values between 0,3 and 0,6.Thus, in cluster A the mobility is mainly prevented by the soil type, and in cluster B, both soil type and vegetation make mobility worse. GO SLOW in cluster A covers most of the area marked GO SLOW in the FDF map, while the GO SLOW area in cluster B is mostly NO GO in the FDF map. Therefore, if we accept that the FDF map is an accurate depiction of mobility in the area, the k–means1 result depicts mobility better than k–means2.

### 9.2.2 Second Experiment: Feasibility of the Analysis Process

The second experiment was a part of the TiTiMaKe–project coordinated by the University of Eastern Finland. One of the goals of the project was to create a number of on–line information services that can serve an environmental monitoring system used, for example, in crisis management. One of the services was a reachability service, which answers the question of how far a vehicle can move from a given position inside a given time frame. In order to implement a reachability service, a mobility map is required. This mobility map was created using exploratory analysis methods.

Thus, the goal of the second experiment was to use the analysis process in a real situation. This would tell whether it would be feasible to try and implement the analysis process in an actual application. Thus, the

second experiment would also answer whether expert users could use the the analysis process in a real problem solving situation. In experiment one, it was known that the data used was such that it could be used to solve the problem. In the second experiment, raw data was gathered from existing sources without a priori certainty of their usefulness.

The input for the experiment was gathered from various public sources, and consisted of road, soil, elevation, forest, and swamp data. Building data was not used in the experiment. Since the experiment area and the surrounding countryside was sparsely built, leaving the building data out did not affect the results in any significant manner. In order to ascertain that knowledge gained during experiment one was not used in this experiment, an FDF Pasi driver was used as a domain expert. He assisted in analyzing the input data and constructing the normalizations required for further analysis. The expert was not involved in the implementation of the mobility model used by FDF, and thus could give an opinion independent of the FDF model for the mobility of the Pasi vehicle. Therefore, he needed to insert his own knowledge into the process instead of copying the knowledge contained in the FDF mobility model.

Map algebra local multiplication was used instead of clustering to combine the input layers in order to get a more detailed mobility map. The output was then generalized into 10 off–road mobility values, and road and water data was overlaid to the map. The best off–mobility was estimated to be half the speed that could be achieved while on road, and water blocked movement completely.

The whole process of map creation was less than 8 hours of work for two people. Both were expert users, one a GIS expert and the other a domain expert. Most of the time was actually spent in collecting the data and preparing it for the analysis. The normalization, computational analysis, and additional analysis steps took only a small fraction of the total time, although several iterations of the computational analysis and its interpretation were required before acquiring an acceptable analysis result. Most of the time was spent on gathering and preprocessing the input data. Thus, reusing the same data for some other analysis would take only a fraction of the time used in the experiment.

The map produced in the experiment was examined using expert evaluation. No glaring problems were found in the test area, and the map was thus deemed usable for the purposes of the TiTiMaKe project. Since the main focus of the project was combining the various services together, the

accuracy of the map was not extensively evaluated. In some parts outside the test area the map had gross mistakes. The most conspicuous problem was that the entire center of the city of Kuopio was apparently covered in water. The reason for this was found in the input soil type data. The soil type for the city center was omitted from the input, and omitted areas had the same numerical soil type code as water areas. Thus, when the vector soil type map was rasterized, omitted areas turned into water.

The experiment demonstrated that it is feasible to create an analysis from scratch using exploratory methods in a reasonable amount of time. While the output was not particularly accurate in some places, it was sufficient for its purpose. Together, the two experiments have demonstrated that experimental methods can be used as an alternative to the traditional model–based approach. The output, when properly prepared, is of similar quality to using an existing model, and thus exploratory analysis is a valid alternative to model-based analysis. Furthermore, as has been shown in Chapter 8, it is possible to formalize exploratory analysis into a well-defined process. Such a model can be used, for example, to review specific analyses, or develop the analysis process further. However, it should be noted that the model described in this work is only one way to formalize spatial analysis. It is by no means the only way, nor should it be considered definitive.

# Part III

# Discussion and Conclusions

# 10. Discussion

## 10.1 First Part: Teaching SDA

The main research question in the first part of the thesis was *"How can software visualization be used to help geoinformatics students to learn spatial data structures and algorithms"*. Spatial data structures and algorithms have been visualized for decades. For example, the original R-Tree paper by Guttman [42] contained both a tree view and an area view visualizations of the R-tree. Similarly, there are algorithm animations for spatial data, such as the animations related to the works of Samet [99, 101]. However, to my knowledge, these spatial visualizations and animations are typically implemented without assessing or taking into account the pedagogical effectiveness of the output.

Hundhausen et al. [49] have shown that meaningful interaction with visualizations is a requirement for a useful learning tool. Different types and levels of interaction have been formalized in the engagement taxonomy [83] and its extension [80]. According to the engagement taxonomy, the higher and more involved the interaction between the learner and the system is, the more likely the visualization is to facilitate learning.

The TRAKLA2 exercises described in this study are one implementation of interactive visualizations with a high level of engagement. The results gained from their use on the spatial data algorithms course at Aalto University indicate, that the TRAKLA2 system can promote learning SDA the same way it promotes learning basic data structures and algorithms. However, due to the small class sizes the results cannot be considered definitive. Moreover, as seen in Appendix A, creating such exercises can be difficult. Thus, some spatial algorithms may require techniques and approaches different from the ones investigated in this study.

The main SDA teaching research question was divided into two smaller problems: *"How can spatial algorithms be visualized"* and *"Do the implemented TRAKLA2 spatial exercises promote learning"*.

### 10.1.1 How Can Spatial Algorithms Be Visualized?

The first of these research questions is further divided into three subquestions: *"What kinds of algorithm visualization elements are available for spatial visualization"*, *"How does spatial data affect algorithm visualization design"*, and *"How can comprehensible spatial data simulations be implemented using TRAKLA2"*. In effect, the topic of these three subquestions is what kinds of visualizations can be used in this situation, how they differ from one-dimensional visualizations, and how such visualizations can be implemented in an on-line learning environment.

*What Kinds of Algorithm Visualization Elements are Available for Spatial Visualization?*

The literature that discusses spatial algorithms, such as [27, 28, 25, 34, 35, 36, 42, 46, 76, 99, 101], contain illustrations that use a two–dimensional view to show the spatial data elements. Therefore, it can be assumed that an area view is typically assumed to be required in order to visualize spatial data algorithms effectively. In many cases this *area view* is the only visualization used.

However, the area view omits the exact details of how the data structure is arranged, and thus hides information from the viewer. In some cases, it can be useful to decrease the amount of details shown at once, and thus perhaps make the visualization easier to comprehend. Decreasing the amount of details can, however, also hide important information from the user, and thus hinder their understanding of the topic. Therefore, in many cases, both an area view, and a more traditional data structure view are used together in order to give the user a more complete picture of the situation. Examples of such can be found both in scientific papers [34, 42] and textbooks [27].

Whether it is better to only use area views, or combine them with data structure views, depends on the algorithm or data structure in question, as well as on the level of detail it is described in. In the case of multidimensional trees, such as Quad-tree [34] or R-tree [42], it is reasonable to use both views. An area view can be used to show how the data is distributed spatially and how the data distribution affects the tree structure.

It is hard, however, to use such a view to show certain details of the tree, such as how the tree nodes are arranged and how data is arranged inside a given tree node. Thus, a data structure view of the tree is required in order to show the structure and the details of the tree.

In other cases, the data structure view can either be used, or left out. For example, in the case of line segment intersection [13], the data structure level can be included, as demonstrated in [27], or left out as in [25]. The data structure view included in [27] shows a binary tree. The tree is used to store the order in which the sweep line intersects the current working set of line segments, and the visualization shows how the elements are arranged in this tree. As the order is defined by the position of the sweep line, it is possible to understand how the algorithm works without knowing the details of how the line segments are stored. Therefore, it depends on the author whether the data structure view is required. By using the view, the author shows how the algorithm works in more detail, but at the same time gives the reader more information to comprehend.

Thus, when designing a spatial algorithm visualization, the author must consider the algorithm in question and the level of abstraction the visualization is on. After such details have been decided, it is possible to select the views used in the visualization. The more important the internal arrangement of a data structure is for comprehending how the algorithm works, the more important it is to include a data structure view to illustrate the structure. However, adding such a view can sometimes be detrimental for the clarity of the visualization. If the focus of the visualization is on a very high level of abstraction, the details of most data structures are typically not essential for understanding the topic.

Similarly, in creating algorithm visualizations for spatial data, the use of an area view is essential. Without it, it is not possible to show the two-dimensional distribution of the data, and thus the viewer cannot hope to comprehend what the data set represents. This is unlikely to change even if interactive visualizations are used instead of static illustrations. Depending on the situation, data structure views may also be required. In TRAKLA2 both views are often used. This makes it possible for the learner to see both how different data elements relate to one other and how they are arranged in a data structure.

In most cases, the data structure view can be based on the canonical views described in Section 4.1.1. The two–dimensional area view shows the area covered by the data, and often includes details about the data

structure. For example, when visualizing a line sweep algorithm, the area view also includes a visualization of the conceptual sweep line. However, since a computer monitor has only two physical dimensions, it is sometimes difficult to distinguish between data elements and other algorithm artifacts. Methods such as color or emphasis can be used to distinguish between the two.

*How Does Spatial Data Affect Algorithm Visualization Design?*
The two-dimensional nature of the data complicates the creation of spatial data structure visualizations. This ties to the second subquestion *"How does spatial data affect algorithm visualization design"*. The first obvious difference between one-dimensional and two-dimensional algorithm visualizations is the use of an area view to show how spatial data elements are arranged. The second obvious difference is the use of area and data structure views together in order to show the data from two different points of view. However, it must be mentioned that multiple views can also be useful in the visualization of certain data structures or algorithms handling one-dimensional data. The use of multiple views can cause additional problems in the visualization design. For example, if the amount of space available for visualizations is restricted, it may be difficult to place multiple visualizations in the available space.

In case of interactive algorithm visualizations, such as the ones used in TRAKLA2, the user interface design for spatial algorithms can be more complicated than for one-dimensional algorithms. Many one-dimensional data structure exercises can be implemented using one algorithm visualization view. With spatial data algorithms, this is often impossible. The user has to interact with data elements at a given geographic position by using an area view, and in a given position in the data structure by using a data structure view. This requires separate views for the two types of interaction.

Sometimes the interaction can be even more complex. For example, in the line sweep exercise described in Appendix A, the learner must be able to remove elements from the priority queue, add line segment intersections to the queue, and modify the order of the line segments in the adjacency structure. In order to be able to do all this, the learner must interact with three separate data structures: the queue, the line segments, and the adjacency structure. Therefore, at least three different views are required.

Furthermore, in many spatial exercises multiple views of the same data are required in order to enable sufficient interaction. For example, one view can be used to demonstrate the details of the data structure and another to demonstrate how the data elements are related in two dimensions. The first view could then be used to modify the manner the data is arranged in the structure and the second to pick data elements according to their spatial characteristics. Of course, depending on the exercise, the total number of views required can vary. Several exercises include many data structures that need to be shown to the learner in order to solve the exercise. As shown by the closest pair of points exercise described in Appendix A, including too many views and too complex interaction can make the exercise unusable. Some exercises, on the other hand, can be implemented by using only one visualization, as shown by the Delaunay Triangulation exercises described in Appendix A.

The more complex user interfaces caused by the increased number and complexity of visualizations can also affect the GUI semantics. In most TRAKLA2 exercises with one-dimensional data, the semantics of clicking or dragging are the same: a click selects an element and a drag moves, copies or swaps it. Spatial exercises often require the modification of the semantics. Unfortunately, this forces the learners to master the user interface semantics for each exercise separately, which increases the cognitive load each exercise imposes.

This is shown, for example, in the expanding wave exercise in Appendix A, where the semantics of clicking on a node can be changed using a radio button. With both radio button settings, the main semantics of click are to select. The selection, however, has side effects, which depend on the setting. With "connect" -setting, selection adds the selected point to the triangulation; with "calculate angle" -setting, selection calculates the angle created by the selected point and two other points. These side effects of selection are unique to the expanding wave exercise, and the learner cannot anticipate them before starting the exercise. There are side effects in most of the exercises. Therefore, each time the learner starts solving a new exercise, they are exposed to new side effects.

*How Can Comprehensible Spatial Data Simulations Be Implemented Using TRAKLA2?*

The third subquestion on spatial data visualization was *"How can comprehensible spatial data simulations be implemented using TRAKLA2"*. For implementing such exercises, TRAKLA2 required new data elements

that can store multidimensional data, a new two-dimensional area visualization, and an implementation of required data structures, algorithms, and exercises.

New data elements were required in order to display two-dimensional data items. The new elements were implemented as *structures*, or collections of primitive data elements that have been given semantics. A point, for example, is implemented as a pair of floating point numbers representing coordinates, and a string representing a name. The coordinates define the location of the point in two dimensions, and the name gives it an identity.

The spatial data elements were used in the implementation of spatial data structures and algorithms. These, in turn, were used to construct model answers for exercises, which were used both for checking the correctness of a submission and for giving feedback to the students. Visualizations were used to construct the graphical user interface for the exercises. An area visualization was required for visualizing the spatial data elements in two dimensions. The area visualization was implemented to contain both actual data elements stored in the structure, as well as visual cues of more abstract algorithmic concepts, such as a sweep line or the minimum bounding rectangle of a polygon. Each exercise also required a generator of random input data.

Due to the two–dimensional nature of the data, input data generation was often more complex than in one-dimensional cases. The input data had to be comprehensible in the area view. Thus, the data items were required to be, for example, sufficiently evenly distributed but still random. The exact requirements for input data varied between exercises.

As shown in Appendices A and B, a number of exercises has been implemented in this work. However, as shown in Appendix A, not all of these implementations were deemed successful. The selection of appropriate visualizations and user interface semantics for each exercise were perhaps the two most challenging problems during the implementation process. The unsuccessful exercise described in Appendix A.3 fails on both of these accounts: there are too many visualizations, and GUI semantics for the exercise are very cumbersome. This particular exercise also has several other problems, which are discussed further in the Appendix.

The problems with the user interface semantics tie to one of the main design principles of TRAKLA2: a mouse click selects the clicked element, and a mouse drag swaps, moves or copies the dragged element into a new

position. For spatial exercises this design principle was broken, mainly by giving the actions exercise-specific side effects. A selection, for example, could add, remove, or change the data, highlight or create new visualizations, or have other unique effects. Thus, in effect, each exercise used whatever semantics seemed most appropriate for the situation. At the start of the project, there was insufficient information about the implementation of spatial exercises in TRAKLA2 for designing expanded semantics in a systematic fashion. Now, perhaps, it could be possible to map the semantics down and see if there is a uniform way to approach the problem in all exercises. One way to do this would be to document the different types of semantics used in the exercises, and to try to see if they can somehow be arranged into categories. The categorization can then be, for example, used in the documentation or system tutorial to show the learners what kinds of user interface semantics they can expect.

### 10.1.2 Do the Implemented TRAKLA2 Spatial Exercise Promote Learning?

The final research question for teaching was *"Do the implemented TRAKLA2 spatial exercises promote learning"*. Due to the small class sizes, it was not possible to give definitive answers to this research question. Quantitative data implies, however, that the exercises have an effect on student learning. Previous research on basic data structures and algorithms has shown that such exercises can be as effective as class-room exercises. These results hint that the same can hold true for teaching spatial data structures and algorithms. Further investigation with a better experimental setup and a larger number of participants is, however, required before anything definitive can be said.

Qualitative data shows that the students use the system, and this is reflected in their exam answers. Therefore, at least some students remember what they learned in using the system, and are able to recall and apply these lessons in an exam situation. Furthermore, the students enjoy using the system. However, the system must work sufficiently well before students feel comfortable using it. The interviews highlighted many aspects of the system the students felt uncomfortable with. These could use some further work in order to make the system more user-friendly.

The use of automatic assessment also has many other advantages, which are not directly related to the students' learning results on the course. A large advantage for course staff is that the use of automatic assessment

can help to free resources. Classroom exercises can be replaced by an automated system, if the two methods have the same effect on learning results. This can free additional human resources for other teaching activities. Furthermore, after a set of exercises has been developed, the amount of effort required for including them in a course is independent of the number of students on the course. Therefore, automatically assessed exercises are most useful in courses with a large number of students.

From the students' the point of view, web-based automated exercises give opportunities not present in traditional class-room exercises. The students can submit their exercises whenever and wherever they wish, and can be given immediate feedback. In addition, automated systems allow for resubmission, which can promote learning [74]. Since the system assesses the exercises automatically, it does not require any additional resources from the course staff.

### 10.1.3 Ethical Considerations of The First Part of the Research

The first part of this research used students on a compulsory course as test subjects. Although the goal of the study - and the initial hypothesis of the researchers - was that the use of the system improves learning, the students were not given a choice whether they wished to participate in the experiment. This raises the question whether it was ethical to use the students in the research.

When the research was conducted, there were no formal ethical guidelines, nor a committee of research ethics at the Helsinki University of Technology. Therefore, the decisions of how to conduct the research and how to use students as participants in the experiment were made together by the author and the supervising professor.

When the decision to include the TRAKLA2 system as an obligatory part of the SDA course was made, it was the honest opinion of both the researcher and the supervisor that including the system is an honest attempt of improving the quality of the teaching. Furthermore, the amount of work included in the course was such, that the inclusion of TRAKLA2 did not cause undue amount of extra effort for the students. After the inclusion, the total estimated workload was still well inside the guidelines given by the university. Furthermore, the students were not placed into an unequal situation, since everyone had to use the system. Thus, all students on the same course were given the same teaching, and the same opportunities for learning.

Modifications to courses are periodically done by teachers. Therefore, the inclusion of a new learning method in the form of TRAKLA2 can be considered as normal part of trying to develop a course for the better. In all such cases, there is a chance that the change is for the worse. This does not mean, however, that courses should never be changed. Instead, change should be encouraged, since all students have the right to the best possible teaching they can get. Thus, there were no ethical problems in the research.

## 10.2 Second Part: Spatial Analysis

The main research question for this part of the work was *"How can exploratory analysis methods be used to enhance suitability analysis"*. The idea of exploratory analysis was proposed in the 1970s [116]. Thus, the idea of data-driven examination of data in order to find novel information is not new. The term visual analytics, on the other hand, was developed around the turn of the millennium [114], but the idea of using information visualization, interaction, and related techniques as part of exploratory analysis is older. For example, parallel coordinates plot was developed in the 1980s [52], and research on the topic has been going on from at least since the 1970s [4, 22].

Exploratory analysis methods have been applied to several GIS problems [6, 23, 65, 78, 97], and systems designed for the use of exploratory methods have been developed [8, 44]. There have also been attempts to create general guidelines of how to apply exploratory analysis for spatial data [9]. Therefore, there is more than sufficient evidence to claim that exploratory analysis and visual methods are an accepted part of GIS problem solving.

In this work, exploratory methods are used to solve the cross-country mobility problem, which is an example of a suitability problem. The goal of the suitability analysis is to categorize given locations according to their suitability for a given task, in this case movement. The problem has traditionally been solved by using methods based on mathematical modeling [53, 104, 117]. In this work, it has been shown that exploratory analysis methods can be used to achieve comparable results. In addition, using the exploratory analysis process can preserve and create metadata and other information not present in the model-based method.

The main research question about spatial analysis was divided into two

questions, both of which were further divided into subquestions. The first research question was *"How can suitability analysis be solved using exploratory methods"*, and the second research question was *"How exploratory methods enhance cross-country mobility analysis compared to the existing model-based approach?"*

### 10.2.1 How can suitability analysis be solved using exploratory methods?

This research question was further divided into four subquestions: *"What kind of visualizations are available for suitability analysis"*, *"How can available visualizations be applied to suitability analysis"*, *"How can exploratory methods be used to solve the cross-country mobility problem"*, and *"How can the exploratory suitability analysis process be modelled"*. The topic of these subquestions is what kinds of visual tools there are available for suitability analysis, and how the suitability analysis is structured.

*What Kind of Visualizations Are Available for Suitability Analysis?*
There are numerous different information visualization methods that have been developed. One way to assess these methods is to review how many distinct data dimensions and simultaneous data elements they can support. Some visualizations, such as Chernoff faces [22] or star plots [21] typically depict only very few elements in a single visualization, and are capable of handling only a relatively small number of data vectors. Other methods, such as the parallel coordinates plot [52] can support numerous data dimensions, and a larger number of data vectors. There are also methods that can handle larger number of data elements and more dimensions than the PCP. [57]. Depending on the context, the data in question, and the preferences of the analyst, the best view for each given situation can vary. The PCP and histogram views were chosen for this research.

The information visualization views must be combined with map views in order to understand the spatial nature of the data, since a non-spatial information visualization view cannot adequately convey how the data elements are arranged spatially. There are numerous different ways to visualize spatial data [108]. For suitability analysis, befitting thematic maps need to be selected.

*How Can Available Visualizations Be Applied to Suitability Analysis?*
For several suitability problems the input is limited to less than a dozen layers. Furthermore, at least according to the experiences gained from this project, each layer can be normalized to a limited number of suitability values without losing any vital information. Thus, the maximum number of distinct data elements in the attribute space is the number of data layers times the number of suitability values. Or, if there are $k$ input layers, and $s$ distinct suitability values for each, the maximum number of data elements is $ks$. In practice, the actual number of distinct data elements is typically much lower, since not all suitability combinations are actualized. Thus, for many suitability analysis problems, the number of data dimensions and elements is such that the parallel coordinates plot is a good initial visualization method.

The parallel coordinates plot can be used to visualize the input data in the attribute space. In this project, PCP is typically used for normalized data. In many cases, PCP could also be used to visualize the initial input data before normalization. If the computational analysis method used in the process is such that it gives multidimensional data elements as output - such as clustering - then PCP can also be used to visualize the output of the data analysis process. In cases where the output of the computational analysis is one-dimensional - such as local multiplication - some other visualization is more appropriate. The histogram view used in this work is one such visualization.

In addition to information visualization views, a suitable map view is required in order to understand the spatial aspect of the data. Each input layer needs to be visualized separately or, if the user wishes to visualize them using a single map, a suitable map visualization method has to be used. In this work, each input layer is visualized separately, and one map is used for the output. The variable selected to be visualized on each map view depends, again, on the computational method used. For clustering algorithms, the cluster number is shown, and for the local multiplication, the output of the multiplication is shown. These variables are chosen, because they characterize the output and are also used in the construction of the suitability map.

Separately, the map view and the information visualization view give an imperfect picture of the data under scrutiny. However, by linking the views together, the analyst can gain a more thorough understanding of the situation. Only by being able to view the same data elements in both

the map view and the information visualization view, can the analyst see the whole picture and thus be able to make informed decisions in each situation.

There are also other visualizations that can be used. A table view can be used to get more details of the data value distribution of each input layer. A textual description of each cluster is included in the prototype, and the analyst can use it to see the details of each cluster. More visualizations, such as histograms of the input data layers, or table view of the output, could also be used if required. These views allow the user more ways to examine the data sets, and allow them to see specific details on demand. Such options would allow for the system to be used in varied situations, and for the analyst to use their individual preferences in selecting how to view the data.

The important thing about the visualizations is that they are selected to convey a given message to the analyst. All these messages are such that they are assumed to help the analyst in solving the given problem. In this case, it is assumed that in order to solve a suitability problem, the analyst needs to view the input layers and their meta-data. The analyst also needs to normalize the input for the computational analysis, and then interpret the results of the computation.

*How Can Exploratory Methods Be Used To Solve the Cross-Country Mobility Problem?*

The analysis process described in Chapter 8 and the prototype implementation described in Chapter 9 are an example of how to use exploratory methods for solving suitability problems. The analysis process, described in Figure 8.1 has four main phases, where the analyst familiarizes themselves with the data, modifies it to a format usable in the process, uses computational methods to gain additional information, and finally constructs a solution. The process is user driven, interactive, and iterative, and thus clearly an example of exploratory analysis.

The system has been demonstrated to be capable of solving the cross-country movement problem, which is one example of suitability problems. A number of short, ad-hoc tests indicate that it could also be used to solve other suitability problems. The input data for this method needs to be expressed using raster data layers. However, since vector data can be rasterized, this means that any data layers should be usable inputs for the process, as long as the layer represents data with no spatial dependencies.

The process outlined here is not the only way to solve the problem. The

process does, however, contain all steps required in order to create a solution. First, input data is gathered, evaluated, and selected. Then, the data is modified to be comparable, and each input data layer normalized to represent its effect on the problem at hand. Then, the input data is combined, and the combined data is turned into a solution. Each of these phases is required in order to create a mobility map.

*How Can the Exploratory Suitability Analysis Process Be Modelled?*
The analysis process is described in Chapter 8, and an overview of the process is shown in Figure 8.1. As described in the previous subsection, the process consists of four main phases, which are all required in order to find the solution. A number of short tests indicate that the same process could be used to solve other suitability problems, and therefore it has at least the potential to be general process for solving suitability problems.

While it is likely that some suitability problems can be solved without using all the subphases of the process, the four main phases are always required. The third phase of the process, the execution of a computational method, can, however, be replaced by a user-controlled method for combining the input layers together. Indeed, suitability problems can be solved by using sieve mapping, where each input layer is divided into suitable and non-suitable areas. Areas suitable for the whole activity are then found by overlaying all input layers together, and declaring areas that have no disqualifying criteria (non-suitable in any input layer) as suitable, and other areas unsuitable.

Nevertheless, input gathering, input modification, combination, and production of output are phases that will always be required in suitability analysis.

### 10.2.2 How Exploratory Methods Enhance Cross-Country Mobility Analysis Compared To the Existing Model-Based Approach?

This research question was further divided into three smaller questions: *"How does the output gained from an exploratory process compare to the output of the model-based approach"*, *"Is there new (meta)data or information that can be found using the new process"*, and *"Does the process based on exploratory methods offer improvements over the model-based approach"*. These subquestions discuss the similarities and differences between the two analysis methods, and possible improvements of the use

of exploratory methods.

*How Does the Output Gained from an Exploratory Process Compare To the Output of the Model-Based Approach*

The model-based approach used in this work was the analysis method used by the Finnish Defense Forces. The maps created by using this method originally had eight categories: seven classes of mobility and one for water. For comparison with the cluster-based mobility maps produced in this research, the FDF maps were generalized to three mobility categories. The first category included areas of good mobility, where the vehicle was capable of moving at least half of its maximum speed. The second category included areas where mobility was possible, but using less than half of the maximum speed. The third category included areas where mobility was not possible.

The misclassification analysis first reported in Publication VI and described in Section 9.2 indicates that the generalized FDF model results and exploratory analysis results are comparable. There are, of course, some minor differences. Some of the results, such as the mobility map gained using the DBSCAN clustering algorithm, give results that are not consistent with the FDF model. Detailed analysis of the DBSCAN result indicate that in this case the DBSCAN result is likely to be less accurate with the reality than the FDF model. The reason for this can be found in the DBSCAN algorithm itself.

DBSCAN is a density-based clustering algorithm, which means that it will put into the same cluster all data elements that form a point cloud of sufficient density. Thus, if there is a cloud of sufficiently densely packed data elements that includes both the data elements that depict good mobility and data elements that depict bad mobility, these elements are put into the same cluster. This is what, apparently, happened with the DBSCAN clustering algorithm. The only cluster that contained data elements representing good mobility also contained data elements representing fair, or bad mobility.

Perhaps, since DBSCAN is density-based, it would be better to use the algorithm with the whole data set rather than subset with unique data elements. That way the point density would correspond to the real density of points in the data set instead of the density of the set of unique data elements, which might in turn give better results. This was not, however, tested in this work.

In the second experiment, a mobility map with ten distinct mobility val-

ues was created. This map was not, however, tested against the FDF model, since the map was constructed on a location for which we did not have the corresponding FDF model. Expert analysis of the mobility map deemed it sufficient for the purposes of the experiment. The map was not, however, verified using any external data or on-site experimentation.

*Is There New (Meta)data or Information that Can Be Found Using the New Process?*

The exploratory method preserves a lot of information that is not preserved by the FDF method. The old, model–based approach does not preserve any metadata about why certain position or area is given certain mobility value. The only output of the process is the mobility map, and no information about how the different mobility values came about is preserved. In contrast, using exploratory, user-controlled methods the analyst can record the reasons why each area was given a certain mobility value. For example, when using clustering, the normalizations, the clustering result, and the mobility value decisions made for each cluster can be stored and later reviewed.

This metadata can be useful later on, if the analysis is reviewed, or if new analysis is being done based on the same data. For example, if an area is considered intractable due to the amount of vegetation, it means it is possible to clear roads through it if necessary. Similarly, if the mobility map is shown to be inaccurate in some places, the reasons for this can be investigated. The use of metadata and the information gained during the process was not further investigated in this work.

*Does the Process Based on Exploratory Methods Offer Improvements Over the Model-Based Approach?*

Compared to the existing model-based method, exploratory methods enable the analyst and other users to review the analysis process and thus, for example, make certain that the result is valid. The metadata could also be used in further analysis, if a similar problem is later encountered. The model-based approach, on the other hand, is more of a black box, where data is put in and from which a result is gained. No data about the procedure itself is preserved, and all further analysis of the conclusions must be done from scratch.

The ability to review the analysis process has other potential benefits besides assessing the validity and correctness of an analysis. One of the initial motivations for this research was the fact that in international cri-

sis management exercises the different actors working together did not trust each others' analyses. The main reason for the distrust was the inability to ascertain how a given analysis had been produced. When exploratory methods are used instead of a model-based approach, the metadata of the work can be stored for further review. Therefore, it is possible to give this metadata to other actors, who can then ascertain how a specific analysis had been conducted and decide, whether they can use the analysis result.

The method is also flexible and capable of handling many different types of data. A traditional model typically requires certain types of input data, and does not work if the data is in a different format or missing. The bigger role of the analyst in the exploratory method enables them to use whatever data happens to be available when an analysis needs to be done. Thus, the new method is more flexible than existing model-based approaches. Of course, the quality of the input, especially if some important data layers are missing, will be reflected in the output and the results gained may have inconsistencies or errors. However, quality issues are out of the scope of this work.

## 10.3 Comparison of the two processes

Two processes have been discussed in this thesis: the process of solving visual algorithm simulation exercises, and the process of analyzing data using exploratory methods. In the following discussion these will be referred to as the "simulation process", and the "analysis process".

Both processes are interactive, and often iterative. In both cases the process is controlled by the user through a graphical user interface, and the user's interaction with the various visualizations is an important part of the process. In the simulation process, the user interacts with algorithm visualizations that depict various aspects of the data structure or algorithm being studied. In the analysis process, they interact with information visualizations that depict the data being analyzed. Both processes can be divided into clearly defined phases. Since the process is iterative, the user may go through some of the phases several times.

### 10.3.1 Process Structure

Both the simulation process and the analysis process can be divided into roughly four phases, which are here called *preparation*, *visual interaction*, *algorithm execution*, and *interpretation*. In the analysis process, depicted in Figure 8.1, these four phases are *Preparation for the Analysis*, *Visual Data Exploration*, *Computational Data Analysis*, and *Interpretation of the Output*. A similar division of the simulation process is depicted in Figure 10.1.

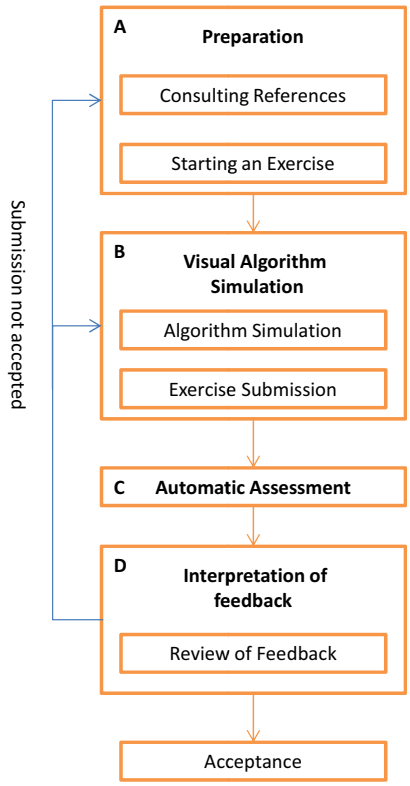


**Figure 10.1.** Overview of the visual algorithm simulation process.

As can be seen in the figure, the simulation process can be depicted using the same division to four phases as the analysis process. In the *preparation* phase the learner consults learning material on the exercise topic and then starts the TRAKLA2 exercise. The learning material can be, for example, a course book, lecture slides, or Internet material. In the *Visual Algorithm Simulation* phase the learner solves and submits a TRAKLA2 exercise. After submission, the exercise is assessed by the TRAKLA2 sys-

tem in the *Automatic Assessment* phase. Finally, in the *Interpretation of Feedback* phase the learner reviews the results of their submission and views the feedback they have received. Depending on the results of the submission, the learner may either accept the results or go back to a previous phase in the process and try to do the exercise anew.

In the simulation process there are fewer sub-phases than in the analysis process, and there are fewer links back to the previous phases of the process. This is due to the fact that the simulation process is intrinsically simpler and more constrained than the analysis process. The aim of a visual algorithm simulation exercise is to teach the learner how a specific data structure, or an algorithm works. Thus, the interaction the learner has with an exercise is typically restricted to specific user interface actions. This is in contrast with the analysis process, where the analyst typically has the functionality of a full desktop GIS system at their disposal and can select the best way to proceed according to their expert knowledge and the problem at hand.

### 10.3.2 Generalization of the Processes

Both the simulation process and the analysis process can be seen as examples of a general problem solving process, as depicted in Figure 10.2. In the figure, each process phase containing a face, is an interactive phase, where the user guides the process. The third phase, which has no face associated with it, is not user-controlled. The figure also explains the data or visualizations the user interacts with in each phase of the process.

The figure highlights the similarities between the two processes. Both can be depicted using four phases, three of which are characterized by high level of interaction between the user and the computer. The exception to this is the algorithm execution phase, which typically has minimal user input. It should be noted, however, that the algorithm execution phase could also be replaced with an interactive phase, where the user is in control. Of the three interactive phases, the preparation phase is characterized by interaction between the user and various information sources. These sources could range from discussions with other people to existing literature and databases.

The two remaining phases of the process, visual interaction and interpretation, are characterized by interaction between the user and the visualizations. In both phases the user manipulates the visualizations in order to advance the process. In the visual interaction phase the user is

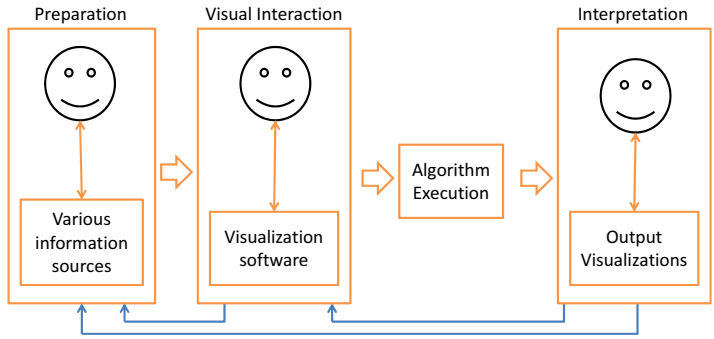


**Figure 10.2.** Depiction of a general problem solving process. Face inside a phase depicts interaction with the user, and the box inside a phase describes what the user interacts with

interacting with visualizations that depict the input data for the process and the modifications done to them via user interaction. In the interpretation phase the user is interacting with the output of the algorithm execution phase.

For example, when using TRAKLA2, the learner views data structure visualizations and modifies the visualized data structures. The goal is to simulate the execution of an algorithm. In the interpretation phase the learner views the feedback given by the system. The feedback consists of textual and visual components. Textual feedback tells the learner how many simulation steps they did correctly, and the amount of points they gained. Visual feedback consists of a model answer which shows how a real algorithm would modify the data structures. Interaction consists of viewing different phases of the model answer and comparing it to the learner's submission.

### 10.3.3 Sharing and Transferring Knowledge

Two notable differences between the processes are the knowledge distribution between the user and the software, and the knowledge transfer between the user and the software. Knowledge, in this context, is defined

to be the means required to solve the current problem in a manner that creates an acceptable solution. For TRAKLA2 exercises, the acceptable solution is a correct algorithm simulation sequence; for cross–country mobility the acceptable solution is a mobility map the analyst considers to be sufficiently accurate.

For the simulation process, the learning environment contains knowledge of how the given exercise should be solved. When a learner starts an exercise, the learning environment selects the appropriate visualizations and shows them to the learner, who is then required to interact with the visualizations in order to solve the exercise. The feedback the learner gains when submitting an exercise is designed to enable the learner to find out where they did mistakes. These facilities have been designed in order to help the learner create a viable mental model of the topic of the exercise. Thus, the system helps the learner to gain new knowledge, and the knowledge is transferred from the system to the learner.

For the analysis process, the user holds the knowledge. The prototype application described in this work has been implemented to contain as little knowledge as possible in order to make it flexible and acceptable in situations where multiple actors need to cooperate. It is possible to create an exploratory tool that contains knowledge of the analysis process. However, the review of the available tools, described in Chapter 7 would imply that the more general the tool is, the less knowledge concerning a particular problem it contains.

During each phase the analyst inserts their knowledge to the analysis process. Since the process itself is values-free in its general form, the analyst needs to do this in order to be able to advance the process. Thus, here the analyst holds the knowledge, and uses it to guide the process towards the desired goal. The analyst also gains new knowledge during the process, as he or she manipulates the data set. During this process, the data is rearranged and shown using various new visualizations, which in turn give the analyst a chance to see the data in new ways and thus find new insight and construct new knowledge.

In addition to the system and the user, there is also a third party involved in the distribution of knowledge: the system developer. Especially in the case of TRAKLA2, the role of the developer in the knowledge management is significant. In TRAKLA2, the developer of the system implements the exercises that the learners must solve. Thus, the developer originally holds the knowledge, and tries to implement the exercises in a

way that is consistent with their mental model of the data structure or algorithm at hand. In the case of a data analysis system, the developer does not have direct knowledge about any specific problem at hand. Instead, the developer creates a system reflecting the developer's knowledge about general problem solving techniques, instead of concentrating on a specific problem. The only exception is when the system is designed to solve a specific problem and therefore needs to contain a lot of problem-specific knowledge. Thus, in the case of the analysis process, the knowledge transfer is from the analyst to the system. The analyst, in turn, can gain new knowledge by interacting with the system.

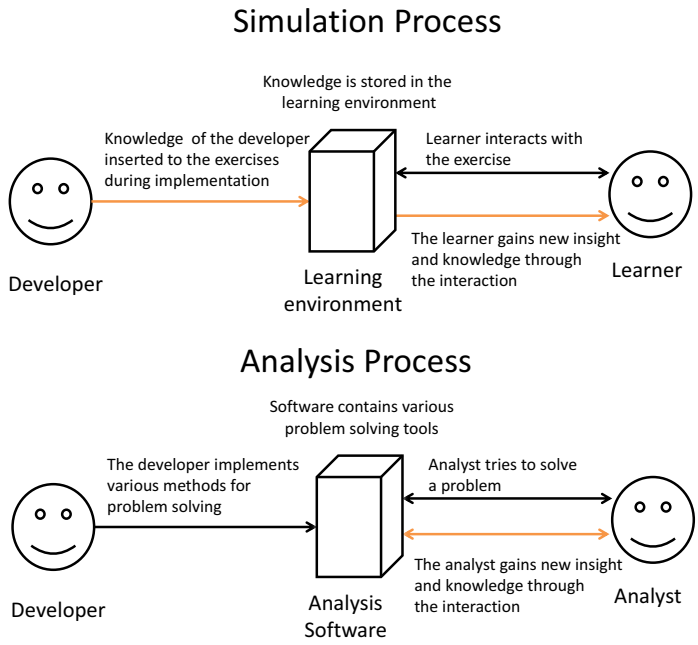


**Figure 10.3.** Knowledge transfer between the system and the user, as well as the system and the developer. Orange arrows show knowledge transfer, while black arrows show interaction and other effects.

Figure 10.3 shows how knowledge is transferred between the system developer, the software, and the user. Orange arrows in the figure show transfer of knowledge, while black arrows stand for other types of interaction. The difference between the two processes can easily be seen in the figure. In the case of the simulation process, the knowledge is transferred from the developer to the learner via the learning environment; in the case of the analysis process, the knowledge transfer is between the analyst and the system.

### 10.3.4 Improving the Processes

The general outline of the process depicted in Figure 10.2 and the knowledge transfer model depicted in Figure 10.3 show how both the simulation and the analysis process can be structured. The processes have much in common, but are also in some ways the mirror images of each other. This structure can potentially be used to help, for example, in the design of an analysis, in the design of an analysis software, or in the design of an educational system.

In the simulation process the developer creates an exercise, into which they store their knowledge of the exercise topic. The exercise is designed so that when a learner tries to solve the exercise, the knowledge stored in the exercise should be transferred to the learner. This is done by the learner testing their mental model of an algorithm against the implementation originally created by the developer, and visualized using the learning environment. It is via this visualization, that the knowledge of the developer can be transferred to the learner.

In the analysis process, on the other hand, the analyst uses their knowledge to guide the analysis process. They also insert their knowledge to the process and to the analysis software as required. Typically, this knowledge is not completely documented or stored in the system for future use. However, if the metadata and the knowledge about the analysis process were to be stored, it would make it possible to improve the analysis process in several ways.

First, a stored the analysis process can be reviewed. If the data is stored, it is later possible to see how a particular result was reached, and verify whether the result is valid. In order to do this review thoroughly, a lot of information about the analysis process needs to be saved. For example, in order to later review a cross-country mobility analysis, at least the following need to be stored: the original input data, all data manipulations done for it, the preprocessing done in order to make the data compatible with the algorithmic method used, the algorithm parameters, algorithm output, and interpretation of the output. In the optimal case, also the analyst's reasoning for each action would be stored, but that would likely lead to a situation, where the analysis process itself becomes too complex and cumbersome compared to the amount of benefit gained from the increased effort required.

Second, if the decision making processes done by the analyst could be

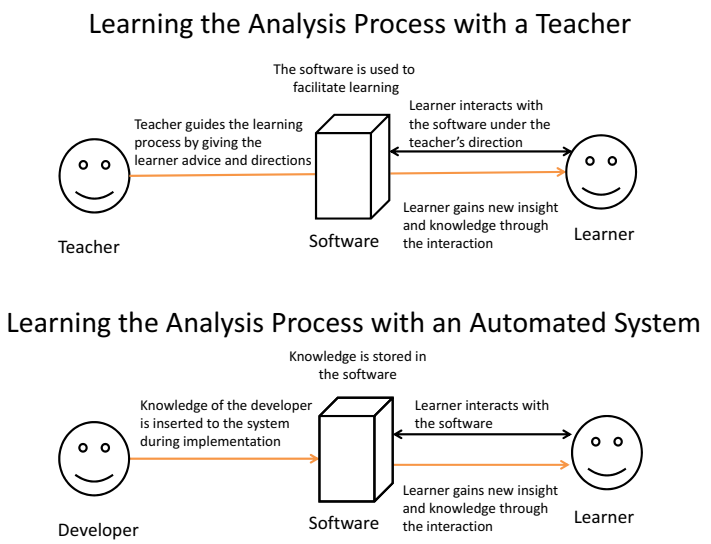


**Figure 10.4.** Learning under teacher supervision, and with an automated system.

stored in addition to the other metadata of the process, it might also be possible to use the stored information for teaching purposes. Currently, classroom exercises are used in the teaching of spatial analysis at Aalto University. In the exercise session the teacher gives students analysis tasks they need to complete. The teacher may also show an example of how the analysis task can be solved. If the students have questions or problems, the teacher is there to help them. At Aalto, this works well on most courses on spatial analysis, since the class sizes tend to be small. However, as the number of students increases, the amount of teaching resources required for this type of teaching also increases. Thus, when the class size become sufficiently large, alternative teaching methods, such as web-based learning environments, may become an attractive alternative. If automatically assessed, web-based exercises could be used to help teaching spatial analysis, the amount of teaching resources required per student would decrease dramatically. Such exercises would also have other advantages over class-based ones, such as support for distance learning.

Of course, implementing such a web-based system would be a major undertaking requiring considerable resources. In such a system, the knowledge transfer would be similar to the simulation process. The developer needs to insert their knowledge into the system, where it is stored. Then,

the learner interacts with the system in order to gain new knowledge from it. In the current instruction model at Aalto, the software used does not contain knowledge about the problem. Instead, it is used merely as a medium of knowledge transfer from the teacher to the learner.

The difference between the two learning approaches is shown in Figure 10.4. The top of the figure represents how knowledge is transferred in the teacher-led approach. The teacher gives the students exercises they need to solve, and instructs them in the problem solving process. The software is used to facilitate learning, but does not contain any knowledge about the problem solving process in itself. When using a learning environment, the developer's knowledge is stored in the system and the learner gains knowledge by interacting with the system without any teacher intervention.

The actual implementation of the learning environment could have similarities to the Alice visual programming environment [24]. Alice has tutorials for teaching how the framework can be used. These tutorials strongly limit the amount of user interface actions available for the user, and thus force them to use particular actions. The goal of the tutorials is to help to learn how the system can be used. Similar methods could be used to teach the learner how to use a system for learning spatial analysis without teacher intervention, and guide them towards a specific way of solving an analysis problem. After the learner has cleared the tutorials, they can be given exercises where all user interface functionality is available.

There are, however, several other factors that need to be taken into consideration when building a learning environment for spatial analysis. For example, for many spatial analysis problems there is no unambiguous correct solution. This may complicate the creation of an automatic assessment system for the environment, since the system would have to be able to assess whether a given solution is acceptable or not. When the correct solution is not unambiguous, designing an automatic assessment procedure is likely to be rather challenging.

# 11. Conclusion

This work has concentrated on two processes that are iterative, interactive, and characterized by a high degree of interaction between the user and various visualizations. The visualizations used in the simulation process, in which the user tries to solve spatial data algorithm exercises, are quite different from the visualizations used in the analysis process, in which the user tries to analyze a spatial problem. The simulation process employs algorithm visualizations, which depict the status of a data structure or a number of data structures using various views. The focus is on showing how data is arranged in a data structure. The analysis process, on the other hand, uses map and information visualizations in various ways to depict the data being analyzed. The focus is on showing the data and the relationships between data items.

Despite these differences, the two processes have much in common. Both are user-driven, often highly iterative, and enable interaction with the computer through the manipulation of visualizations. Furthermore, both processes can be depicted using the same overall structure of preparation, visual interaction, algorithm execution, and output interpretation. The knowledge transfer between the system and the user, and between the system and the system developer, is also similar in the two processes. In many ways, the two processes mirror each other.

In the simulation process, the developer has inserted their knowledge into the system during system development, and as the user solves exercises, the knowledge is transferred from the system to the user. Thus, the direction of the knowledge transfer is from the system to the user. Or, from the point of view of mental models, the developer has created the system in such a way that it challenges users with non-viable mental models.

In the analysis process, the user holds the knowledge, and inserts their

knowledge to the system during the process. The developer has merely enabled this knowledge insertion by implementing a number of visualizations and tools in the system. The user reflects on the feedback the system gives to them during the process. This feedback enables the user to guide the analysis process and, if possible, gain new knowledge about the situation. Thus, the direction of knowledge transfer is from the user to the system. Or, from the point of view of mental models, the system enables the user to see whether their mental model of the situation is viable, and if not, adjust the model accordingly.

This research has also shown, that the simulation process can help to teach spatial data algorithms, and that the analysis process can be used to solve spatial analysis problems. Thus these processes offer an alternative to traditional class room learning for teaching SDA, and and alternative to spatial models in analyzing problems. In neither case can the work be said to be complete.

The TRAKLA2 spatial extension constructed in this research contains approximately a dozen exercises, which cover only a small part of what is actually discussed on the Aalto University spatial data algorithms course. Thus, there is much work to be done, before the system contains a sufficient range of exercises for the students to benefit from them throughout the course. Furthermore, all the exercises currently implemented cover algorithms where the data is in vector format, meaning that each data element is a distinct object, which has a location, an extent, and a number of attribute values. In addition to vectors, spatial data can also be stored as rasters, or fields, where the value of specific spatial attribute is stored separately for each location. This raster data format is used, for example, in the cross-country mobility analysis discussed in this work.

There were efforts to develop raster data exercises during the first part of the research. However, no worthwhile method for making visual algorithm simulation exercises for raster data was found. Vector data, and vector data structures typically work well with algorithm simulation. With different input, the user typically needs to do different manipulations to the visualizations in order to accomplish their goal. This reflects the fact that typically with vector data, the algorithm execution differs between different inputs. In other words, the input affects how the control structures of the algorithm behave.

With raster data and raster data structures, the input typically has little effect on the algorithm execution. The same lines of code are executed in

the same order, and the difference is typically only in the output of the calculations done on specific code lines. In other words, the input typically does not affect how the control structures of the algorithm behave. Thus, TRAKLA2-style algorithm simulation exercises would not work for raster data, since the simulation steps would not depend on the input data.

This thesis does not prove that the implemented TRAKLA2 exercises are particularly effective or helpful learning tools. The results gained from two courses, where the system was used, are promising and indicate the system might help students to learn. However, the small class sizes and the lack of a control group make the results very hard to interpret. Therefore, no definitive conclusions can be made.

Similarly, the prototype for exploratory analysis of suitability problems constructed in this research contains only the functionality required to solve the example problem, which was not directly supported by the ESRI ArcMap desktop GIS environment. It is a long way from being a complete software product that could be given to other people for use. There are many aspects in the prototype that would need to be changed before it could be used to comfortably solve suitability problems in general, since the initial version was meant only for mobility analysis. In the analysis window, for example, it is assumed that the area is split into three categories, which are explicitly identified as GO, GO SLOW, and NO GO; the colors used for depicting these areas in the visualization views are originally taken from the FDF mobility maps, etc. Thus, new functionality could be added to the prototype to make it more useful, and there are also several ways, in which the prototype user interface could be improved.

The testing that was done to ascertain the usability of the prototype was similarly rather limited. The experiments show that it is possible to get results similar to the FDF model using an exploratory process and k-means clustering, and that a mobility map for a given area can be created from scratch in a rather short amount of time. There were also some informal, short tests with other problems, the results of which indicate that the system could be used to solve them. However, no systematic larger testing with a larger number of problems, on a larger geographical scale, or with more users, were conducted. Therefore, no definitive conclusions can be drawn on the generality, usability, or suitability of the system.

## 11.1 Future Work

The results of the simulation process research could use further verification. The number of students in the classes where TRAKLA2 was used was small, and thus the results obtained are not definitive. Furthermore, no control group was arranged due to the small class size. Thus, a further experiment with a larger number of students would be useful. It might be possible to arrange for such an experiment in cooperation with another course at the Aalto university.

The spatial expansion to TRAKLA2 could also benefit from further development. The number of exercises implemented is rather small and covers only a part of the spatial data algorithms course. However, if further exercise development were to be done, part of the focus should be on the development of the user interface semantics. The current TRAKLA2 semantics, which are based on clicking or dragging data structure elements, seem to be insufficient for many spatial data algorithm exercises. The default click and drag semantics (selection and movement/copying) need to be modified in many exercises. This, in turn, makes it harder for the student to solve the exercises, since they first need to learn the user interface semantics before being able to concentrate on solving the exercise itself.

The more complex user interface semantics could be shown, for example, by using a toolbar. This way the learner would know the possible semantics beforehand, and know what kind of semantics are used in each exercise immediately upon starting the exercise. Some steps towards this have been taken in the system in the form of tool tips that are shown when putting the cursor over a visualization. The tooltip shows the semantics active for that particular visualization.

However, before such a user interface element could be implemented, the developers should have a clear vision of what types of semantics are required in spatial data algorithm exercises. This would require a thorough review of the existing exercises as well as design work on how the system could be further expanded before implementation.

In this work the experimental and visual spatial analysis process was applied to only one problem, off-road mobility. There were a few short, informal experiments, in which the process was applied to other problems, including artillery battery locations and wireless communication link locations. However, these experiments did not include any true verification of the results beyond brief expert evaluation. According to these eval-

uations, it seems that the process can be used to solve these problems. However, the results of these experiments are not on a level where they could be included in any scientific report. Thus, one possible direction of future work would be to verify the generality of the described process by applying it to a number of different problems and evaluating whether the results gained are useful.

The current prototype application used to test the process has rather a limited functionality. There are only a handful of computational analysis methods and just a few visualization views in the prototype. For future research, it would be interesting to take, for example, an existing visual analysis application and combine that with an existing data mining framework in order to increase the number of available visualizations and computational methods. This would also make it possible to evaluate the various analysis methods and the overall process in a much more robust way.

Another possible future direction is research on how to store and use the metadata of an analysis process. Stored metadata can be useful later on if the analysis is reviewed or if new analysis is being done on the same data. For example, if some area is considered intractable due to the amount of vegetation, it means it is possible to clear roads through it if necessary. If some parts of the mobility map are shown to be inaccurate, the reasons for this can be investigated. This possibility of storing metadata about the analysis process and the decisions made during the process thus enables further work and research.

The comparison of the two processes revealed many similarities and differences in the processes; how they can be arranged, and especially how the transfer of knowledge between the system and the user is arranged. This work could be used as a starting point for further knowledge management research. The goal of such work could be, for example, to identify, categorize, formalize, and store the knowledge gained during a problem solving process. Such knowledge could then be used to help in solving similar problems. The simplest use for such knowledge would be to have the system store normalizations for further use. A more complicated example would be to use stored analysis processes as a basis for an automated or semi-automated spatial analysis learning environment.

# Appendices

# A. Spatial Data Exercise Examples

## A.1 Line Segment Intersections

The line segment intersections problem is as follows: given a set of possibly intersecting line segments, a line segment intersection algorithm finds all intersections. Such problem is trivial for a human to answer, once the line segments have been drawn. For a computer, however, the problem is non-trivial. The conceptually simplest way of solving such problem by computer is to compare each line segment to all others. Such approach, however, would make $O(n^2)$ comparisons. When the number of line segments grows large, such an algorithm will become inefficient.

A more efficient solution is to use *line sweep* to solve the problem [13]. In the line sweep approach a conceptual sweep line will be moved across the plane, stopping at points where interesting events happen. In this problem, the endpoints of line segments and their intersections are such events. The line sweep algorithm is capable of finding all line segment intersections in $O(nlogn+klogn)$ time, where $k$ is the number intersections in the data set.

Figure 1.1 contains a screenshot of the TRAKLA2 exercise applet for the problem. The applet contains four data structure views. In the top row there is a tree view of a binary heap, which is used as a priority queue, and an area view showing the line segments, their intersections and the current position of the sweep line. In the bottom row there is the adjacency structure used by the algorithm, visualized as an array, and a linked list for storing the output. In the figure, the student has started solving the exercise. As can be seen from the area view, three line segment intersections have been discovered in the simulation and are explicitly shown. Intersections are explicitly drawn to the area view when

the corresponding line segments are next to each other in the adjacency structure, simulating how a real algorithm finds the intersections.

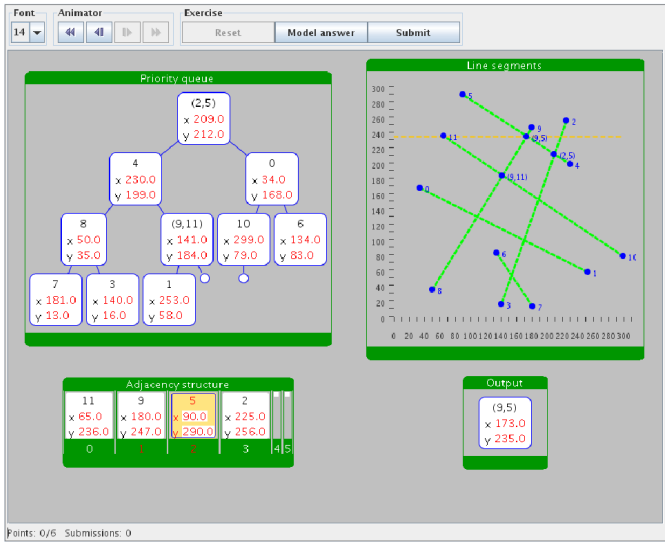


**Figure 1.1.** Line segment intersections with line sweep.

With the exception of the output visualization, all visualizations in the exercise are actually representation level views. The priority queue used in the exercise is a binary heap, which is actually implemented using an array. In the exercise the heap is visualized as a binary tree. Similarly, the adjacency structure in the algorithm is typically a balanced binary tree, such as red-black tree. However, since the purpose of the structure is to order the line segments according to where they intersect the sweep line, we decided to visualize the adjacency structure as an array. In an array the order of the segments is easy for a human to see.

The area view is also a representation level visualization, although significantly different from the two other representation level views. Furthermore, the area view shows a conceptual element that is actually not explicitly included in a real algorithm implementation: the sweep line. Typically, the sweep line is implicitly stored as the y-coordinate of the latest element taken from the priority queue. In the exercise, it needs to be explicitly implemented in order to be visualized.

To solve the exercise, the student simulates the algorithm by dequeueing elements from the priority queue, maintaining the adjacency list, and adding intersections to the priority queue and the output. Snapshots of the solution process can be seen in Figure 1.2. the top left snapshot depicts the situation at the beginning of the simulation, when no elements

have been handled. The second snapshot, at the top right, shows the visualizations after four elements from the priority queue have been handled. The bottom left snapshot depicts the situation after more elements have been handled, and bottom right snapshot shows the situation at the end of the simulation process.

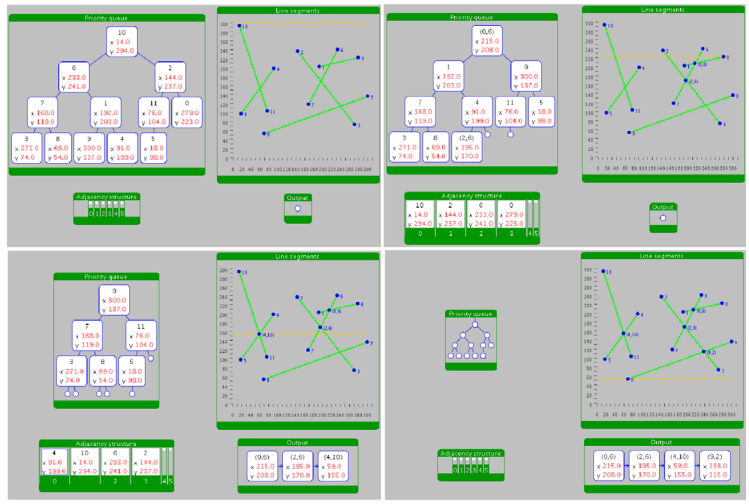

**Figure 1.2.** Four snapshots of solving the line segment exercise. Snapshots start at the top left, and end at the bottom right picture.

The line segment intersection exercise has been used on the SDA course for three years. It was completely re-designed after the first year, and Figure 1.1 illustrates the current version. In the original design the sweep line swept from left to right and was not visualized, and the line segment endpoints and intersections were visualized using much larger elements. The original version of the exercise was much harder for the students to solve than the current version, and most gained approximately 50% of the points. Using the new version students typically gain 80-100% of the points from the exercise.

The exercise combines views on very high level of abstraction (the area view), with views that are closer to the actual implementation (the other views) in order to show the student both the problem instance, and the data structures the algorithm uses to solve the problem.

## A.2 Several Exercises On the Same Topic: Delaunay Triangulation

Delaunay triangulation is a triangulation of point set $P$ such that no point in the set $P$ is inside the circumcircle of any triangle. The Delaunay tri-

angulation is a very important structure in many spatial problems, since each point in the triangulation is connected to its closest neighbouring points, the triangles have as equal angles as possible, and the Delaunay is a dual of the Voronoi diagram for the point set [85]. Both Delaunay and Voronoi have numerous applications in geoinformatics.

Delaunay triangulation and Voronoi diagram are, on the conceptual level, more complicated than line segment intersections. Whereas finding intersections is trivial for a human, constructing a valid Delaunay or Voronoi is not. Therefore, before discussing algorithms that construct, modify, or use the structures, the students need to learn the concepts behind them. Therefore we made several TRAKLA2 exercises about Delaunay triangulation, two of which deal with the creation of a Delaunay Triangulation.

In the first exercise, the student needs to build a valid Delaunay triangulation for a set of points. No specific algorithm needs to be followed, and the exercise is assessed by comparing the student's solution to the actual Delaunay (which is unambiguous) for the point set. The applet for this exercise can be seen in Figure 1.3. In the Figure, the student has started solving the exercise. The exercise shows the Delaunay edges in white, corresponding Voronoi edges in dark grey, and the student has highlighted one triangle, for which the circumcircle is drawn using red dash line. The additional point shown in the Figure is the meeting point of three Voronoi edges (and coincidentally over one Delaunay edge), and the center of the circumcircle.

The student solves the exercise by clicking on pairs of points. If the points are not connected, a new Delaunay edge will be added between them, and if the points are connected, the existing edge will be removed. Snapshots of a solution process can be seen in Figure 1.4. The top left snapshot shows the beginning of the simulation when no Delaunay edges have been added. The top right snapshot shows that the student has started solving the exercise by connecting the points of the convex hull. In the bottom left picture, the Delaunay triangulation is approximately halfway done and the Voronoi diagram is starting to take shape. The bottom right picture shows the completed Delaunay and the corresponding Voronoi.

The goal of the Delaunay construction exercise is for the student to familiarize him- or herself to the concept of Delaunay triangulation, and to be able to create a valid Delaunay for a small point set. Since no actual algorithm is used in the exercise, there are not set data structures

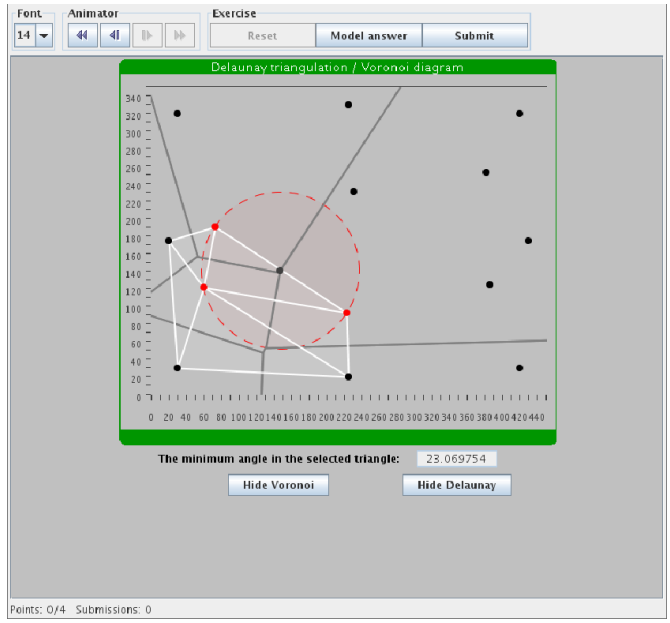


**Figure 1.3.** Delaunay construction exercise.

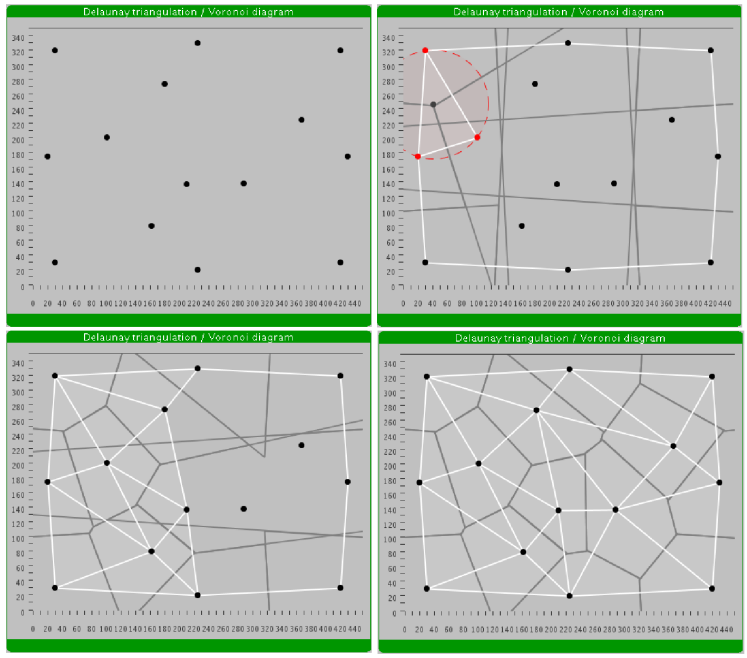


**Figure 1.4.** Snapshots of the Delaunay construction.

in it either. Therefore, the only view used in the exercise is an area view. The area view can be understood to be either representation or domain level view. The view includes some algorithmic information in the form of circumcircles and their minimum angles and therefore can be seen as

representation level. However, the data structures used are not visualized in any way, and therefore the structure can also be seen as domain level visualization. It thus demonstrates how differences between the levels of abstraction can be fuzzy.

In the second Delaunay construction exercise, the student needs to simulate the expanding wave -method for Delaunay construction [76]. With this method the Delaunay triangulation is created by sweeping around each point of the set in turn, connecting the current point to all of its Delaunay neighbours. Data structures the algorithm uses are a set of auxiliary points that form a border around the point set and a queue.

The expanding wave exercise applet can be seen in Figure 1.5. There are two views in the applet. On top there is an area view, which shows the point set and the Delaunay triangulation under construction. The current point to be handled is shown in red, and its latest found neighbouring point in yellow. The circle drawn in the picture is used to find new neighbour candidates, which are drawn in green. Next to the area view are two radio buttons which are used to modify the semantics of clicking on a point in the area. Below the radio buttons there is a "Get candidates" button, which will draw a circle to the area view. Both the current point and the current neighbour will be on the on the edge of the circle. Clicking on the button again will draw a new, larger circle. This feature can be used to find neighbour candidates. At the bottom there is a queue visualized as a linked list.

The student simulates the expanding wave algorithm by manipulating the area view and the queue. Removing an element from the queue will cause it to become the current element in the area view and be colored red. Then, the student can select neighbouring elements , or measure angles from the area visualization, in order to construct the Delaunay triangulation. New elements can be added to the queue by dragging and dropping the corresponding point from the area view to the queue. The gradual construction of the triangulation can be seen in Figure 1.6. The queue has been omitted from the figure.

The two visualizations used in the exercise are on different levels of abstraction. The area view, which shows the points, the Delaunay triangulation being constructed, and the circle for selecting neighbour candidates, is clearly on representation level. The actual data structure implementation is hidden, but several elements used in the algorithm and the triangulation being constructed are visualized in the view. The queue view, on

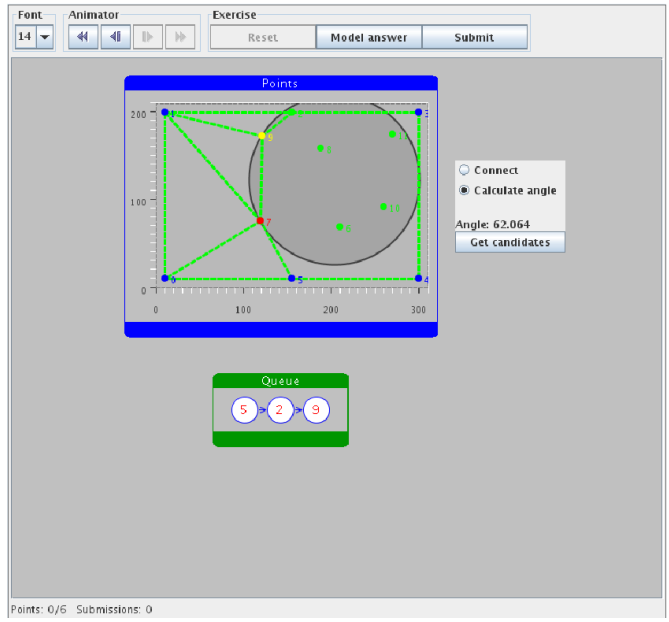


**Figure 1.5.** The expanding wave method.

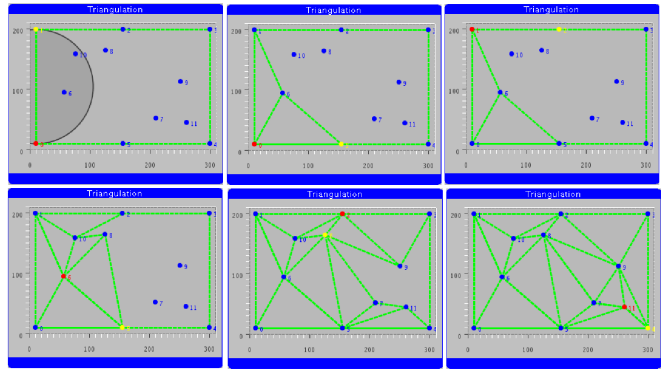


**Figure 1.6.** Snapshots of the expanding wave exercise.

the other hand, accurately depicts how the queue used in the exercise is implemented as a linked list and thus is on data structure level.

## A.3 An Unsuccessful Exercise: Closest Pair of Points

The closest pair of points problem asks what is the shortest distance between two points in a given point set. The problem can be solved in many ways. The conceptually simplest solution is to calculate the distance between all point pairs in the set, leading to a $O(n^2)$ runtime. More efficient solutions can be achieved, for example, as by-product of calcu-

lating the Delaunay triangulation of the set. After all, the closest point pair are neighbours in the triangulation. The problem can also be solved by a divide-and-conquer algorithm originally described by Preparata and Shamos [93]. These more efficient methods can solve the problem in $O(nlogn)$ time.

The divide-and-conquer algorithm works by splitting the point set into two smaller subsets, for which the closest pair problem is then solved recursively. When two subsets are merged, the closest pair is either the current known closest pair, or a pair where the first point is in one subset and the second point in the other subset. The algorithm keeps track of the closest known pair of points, and during each merge-operation, the points sufficiently close to the dividing line between the two point sets are examined for possible closest point pair. The merges in the algorithm are based on the merge operation used in the merge-sort algorithm.

A TRAKLA2 exercise of the algorithm was used on the 2007 course. After the course, the exercise was reviewed, using the students results and expert opinion. The students' results in the exercise were much worse than their results in most other exercises on the course. Furthermore, the user interface was deemed to be very complicated and the exercise gave an overall impression of being messy and hard to understand. Therefore, we decided to remove the exercise from the course.

The exercise applet can be seen in Figure 1.7. As can be seen from the figure, the applet contains a large number visualizations. At the very top, there is an array holding the point set. In this visualization, the coordinates of each point are shown, since those are required in the merge part of the algorithm. Below the point set, there is an auxiliary table used in the merge part of the algorithm, and next to it is the array for holding the current known closest pair. Below that is an area view of the point set, where the part of the area relevant to current merge shown in green. next to the area view is a call stack, used to keep track of where in the recursion tree the algorithm currently is. Below the call stack there is button for moving points from the auxiliary table to the table holding the point set ("move") and buttons for manipulating the call stack ("call" and "return").

As can be seen from Figure 1.7, the user interface to the exercise contains more views than the GUI of any other exercise described. Furthermore it is not initially clear what the function of each view is. For example, the area view highlights the subarea on which a new closest pair should

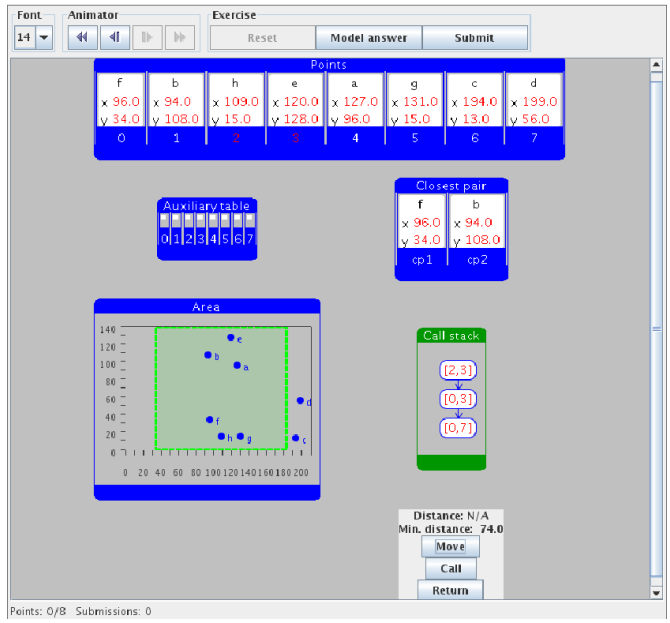


**Figure 1.7.** The expanding wave method.

be searched for. However, not all points in this area need to be considered in a given merge, since some of them may not be included in the currently handled subsets. Relevant points have not, however, been highlighted in any way, and need to be identified using the array visualization. Furthermore, in order to solve the exercise, the student needs to manipulate all the visualizations in the exercise, and in many operations needs to keep track of the state of several visualizations at the same time. Therefore, the GUI of the exercise is very complex, and it probably takes a student quite a lot of time to comprehend both what should be done, how it can be done, and how the different GUI actions correspond to different parts of the algorithm. Therefore we actually hypothesize that many students who were able to solve the exercise did not comprehend how the GUI operations translate to operations of the algorithm. They merely learned how to "win the game" in this exercise. However, since no students were interviewed in the 2007 course, we could not test this hypothesis.

The root of the problem in the closest pair exercise is probably including the whole algorithm in a single exercise, on a very high level of detail. There are several rather difficult concepts that all need to be understood in order to comprehend how the algorithm works. First, the algorithm divides the points into smaller sets using the same recursion principle as the merge-sort algorithm. Second, before the algorithm executes, the

points are already sorted according to their x-coordinate. The points are sorted again, according to their y-coordinate, during the algorithm's execution. Third, during each merge, the points that are in the merged sets are compared to see if a new closest point pair is found. The points are compared to each other in a bottom-up fashion according to their y-coordinate. All these aspects of the algorithm need to be understood in order to truly comprehend how the algorithm works and be able to simulate it. This makes the whole algorithm conceptually very hard. Combined with the complex user interface, this makes the cognitive load for this exercise extremely high [88]. This, in turn, makes the exercise hard for students to solve, and probably even harder to truly understand.

A better approach would probably have been to either split the exercise into several exercises, like was done with the Delaunay triangulation, or at least raise the abstraction level in order to hide some of the algorithmic details. However, many of these faults were truly understood only after the exercise had already been used on the course for the first time.

Currently, the exercise has not been modified after being taken from the course, and there are no plans of doing more development work on it. This is mostly because the closest pair problem, at least in the number of dimensions typically used in geoinformatics, can be efficiently solved using Delaunay triangulation. Therefore, from the pedagogical point of view, the only reason for including this algorithm is for teaching the divide-and-conquer problem solving method. There are, however, numerous other algorithms that use the method and therefore the closest pair of points algorithm is not essential for the course.

The closest pair algorithm was not the only algorithm for which we were unable to create a sensible TRAKLA2 exercise. There were two other cases during the project: Fortune's algorithm for Voronoi construction [36] and a polygon network traversal algorithm by de Berg et al. [28]. In the case of Fortune's algorithm we were not able to create satisfactory GUI interaction for manipulating the beach line used in the algorithm. For the polygon traversal we discovered that TRAKLA2 did not have a good visualization for the Doubly Connected Edge List data structure used in the algorithm. Unfortunately, we did not have sufficient resources for developing a new visualization view just for illustrating the DCEL.

# B. List of Exercises

**Table 2.1.** Spatial exercises in the TRAKLA2 system.

| Name | Description | Years used |
|---|---|---|
| Point in polygon | The learner is to find whether a point is inside a polygon by counting the number of intersections a half-line drawn from the point has with the polygon. | 2007, 2008 |
| Closest pair of points | The learner is to find the closest pair of points in a set of points by using a divide-and-conquer approach based on the mergesort algorithm. | 2007 |
| Douglas-Peucker line simplification | The learner is to simplify a polyline using the Douglas-Peucker line simplification algorithm. | 2007,2008 |
| Line sweep | The learner is to find line segment intersections in a set of line segments by using the line sweep algorithm. The exercise was completely redesigned between the two courses. | 2007, 2008 |
| Voronoi construction | The learner is to construct a valid Voronoi diagram from a set of points. There is no need to follow a specific algorithm, only the end result is assessed. | 2008 |
| Adding a point to TIN | The learner is to add three new points to a Delaunay triangulation and to modify the triangulation so that it still stays valid. There is no need to follow a specific algorithm, only the end result is assessed. | 2008 |
| Expanding wave-method | The learner is to construct a Delaunay triangulation using the expanding wave algorithm. | 2007, 2008 |
| Visibility with rotational sweep | The learner is to find polygon end points visible from a given point by using the rotational sweep algorithm. | 2007,2008 |
| R-tree insert | The learner is to insert a number of polygons into an R-tree. | 2007,2008 |
| Point-region quadtree insert | The learner is to insert a number of points into a point-region quadtree. | 2007,2008 |
| Point in polygon with R-tree | Point in polygon where the edges of the polygon are in an R-tree and the learner must search the R-tree for edges that may cross the half-line. | 2007,2008 |

# Bibliography


[1] Commongis homepage, http://www.commongis.com/, accessed september 4th 2012. Web Page.

[2] Geovista center web page, http://www.geovista.psu.edu/, accessed september 4th, 2012. Web page.

[3] N. Andienko and G. Andrienko. *Interoperating Geographic Information Systems*, chapter IRIS: a tool to support data analysis with maps, pages 221–234. Springer, 1999.

[4] D. F. Andrews. Plots of high-dimensional data. *Biometrics*, 29(1):125–136, 1972.

[5] G. Andrienko, N. Andrienko, U. Demsar, D. Dransch, J. Dykes, S. I. Fabrikant, M. Jern, M.-J. Kraak, H. Schumann, and C. Tominski. Space, time and visual analytics. *International Journal of Geographical Information Science*, 24(10):1577–1600, 2010.

[6] G. Andrienko, N. Andrienko, R. Fischer, V. Mues, and A. Schuck. Reactions to geovisualization: an experience from a european project. *International Journal of Geographical Information Science*, 20(10):1149–1171, 2006.

[7] G. Andrienko, N. Andrienko, D. Keim, A. M. MacEachren, and S. Wrobel. Challenging problems of geospatial visual analytics. *Journal of Visual Languages & Computing*, 22(4):251 – 256, 2011. Part Special Issue on Challenging Problems in Geovisual Analytics.

[8] G. Andrienko, N. Andrienko, and H. Voss. *Maps and the Internet*, chapter GIS for everyone: the CommonGIS project and beyond, pages 131–146. Elsevier, 2003.

[9] N. Andrienko and G. Andrienko. *Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach*. Springer-Verlag, 2005.

[10] R. M. Baecker. Sorting out sorting. Narrated colour videotape, 30 minutes, 1981.

[11] R. M. Baecker. Sorting out sorting: A case study of software visualization for teaching computer science. In M. Brown, J. Domingue, B. Price, and J. Stasko, editors, *Software Visualization: Programming as a Multimedia Experience*, chapter 24, pages 369–381. The MIT Press, Cambridge, MA, 1998.

[12] M. Ben-Ari. Constructivism in computer science education. *Journal of Computers in Mathematics and Science Teaching*, 20(1):45–73, 2001.

[13] J. Bentley and T. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on Computers*, 28(9):643–647, 1979.

[14] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.

[15] J. Bertin. *Graphics and Graphic Information Processing*. de Gruyter, 1981.

[16] M. Bertolotto, S. Di Martino, F. Ferrucci, and T. Kechadi. Towards a framework for mining and analysing spatio-temporal datasets. *International Journal of Geographical Information Science*, 21(8):895–906, 2007.

[17] C. A. Brewer. *Designing Better Maps: A Guide for GIS Users*. Esri Press, 2005.

[18] M. H. Brown. Exploring algorithms using Balsa-II. *IEEE Computer*, 21(5):14–36, 1988.

[19] M. H. Brown. Zeus: a system for algorithm animation and multi-view editing. In *Proceedings of IEEE Workshop on Visual Languages*, pages 4–9, Kobe, Japan, 1991.

[20] S. Caquard, G. Brauen, B. Wright, and P. Jasen. Designing sound in cybercartography: from structured cinematic narratives to unpredictable sound/image interactions. *International Journal of Geographical Information Science*, 22(11-12):1219–1245, 2008.

[21] J. Chambers, W. Cleveland, B. Kleiner, and P. Tukey. *Graphical Methods for Data Analysis*. Wadsworth, 1983.

[22] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68(342):361–368, 1973.

[23] A. Cöltekin, S. Fabrikant, and M. Lacayo. Exploring the efficiency of users' visual analytics strategies based on sequence analysis of eye movement recordings. *International Journal of Geographical Information Science*, 24(10):1559–1575, 2010.

[24] S. Cooper, W. Dann, and R. Pausch. Teaching objects-first in introductory computer science. *SIGCSE Bulletin*, 35(1):191–195, 2003.

[25] T. H. Cormen, C. L. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2009.

[26] J. R. Crouch, Y. Shen, J. A. Austin, and M. S. Dinniman. An educational interactive numerical model of the chesapeake bay. *Computers & Geosciences*, 34(3):247–258, 2008.

[27] M. de Berg, M. V. Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer, 2000.

[28] M. de Berg, R. van Oostrum, and M. Overmars. Simple traversal of a subdivision without extra storage. In *SCG '96: Proceedings of the twelfth annual symposium on Computational geometry*, pages 405–406, New York, NY, USA, 1996. ACM.

[29] G. Degrande and K. Geraedts. An electronic learning environment for the study of seismic wave propagation. *Computers & Geosciences*, 24(6):569–591, 2008.

[30] U. Demšar. *Data mining of geospatial data: combining visual and automatic methods*. PhD thesis, Royal Institute of Technology, 2006.

[31] U. Demšar and K. Virrantaus. Space-time density of trajectories: exploring spatio-temporal patterns in movement data. *International Journal of Geographic Information Science*, 24(10):1527–1542, 2010.

[32] B. Dent, J. Torguson, and T. Hodler. *Cartography: Thematic Map Design*. McGraw-Hill, 2008.

[33] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density–based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[34] R. Finkel and J. Bentley. Quad trees - a data structure for retrieval on composite keys. *Acta Informatica*, 4(1):1–9, 1974.

[35] J. Fisher. Visualizing the connection among convex hull, voronoi diagram and delaunay triangulation. In *37th Midwest Instruction and Computing Symposium*, 2004.

[36] S. Fortune. A sweepline algorithm for voronoi diagrams. In *SCG '86: Proceedings of the second annual symposium on Computational geometry*, pages 313–322, New York, NY, USA, 1986. ACM.

[37] M. Gahegan. Four barriers to the development of effective exploratory visualisation tools for the geosciences. *International Journal of Geographical Information Science*, 13(4):289–309, 1999.

[38] M. Gahegan, R. Agrawal, A. Jaiswal, J. Luo, and K.-H. Soon. A platform for visualizing and experimenting with measures of semantic similarity in ontologies and concept maps. *Transactions in GIS*, 12(6):713–732, 2008.

[39] P. Gerdt and J. Sajaniemi. A web-based service for the automatic detection of roles of variables. In *Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education*, pages 178–182. ACM, New York, NY, USA, 2006.

[40] M. Gerke and C. Heipke. Image-based quality assessment of road databases. *International Journal of Geographical Information Science*, 22(8):871–894, 2008.

[41] P. Gorsevski, P. Jankowski, and P. E. Gessler. Spatial prediction of landslide hazard using fuzzy k-means and dempster-shafer theory. *Transactions in GIS*, 9(4):455–474, 2005.

[42] A. Guttman. R-trees: a dynamic index structure for spatial searching. In *SIGMOD '84: Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, pages 47–57, New York, NY, USA, 1984. ACM Press.

[43] D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. MIT Press, 2001.

[44] F. Hardisty and A. Robinson. The geoviz toolkit: Using component-oriented coordination methods to aid geovisualization application construction. *International Journal of Geographic Information Science*, 25(2):191–210, 2011.

[45] A. Hausner and D. P. Dobkin. GAWAIN: visualizing geometric algorithms with web-based animation. In *SCG '98: Proceedings of the fourteenth annual symposium on Computational geometry*, pages 411–412, New York, NY, USA, 1998. ACM.

[46] C. A. Hipke and S. Schuierer. Vega – a user-centered approach to the distributed visualization of geometric algorithms. Technical report, University of Freiburg, 1998.

[47] B. Huang and M. F. Worboys. Dynamic modelling and visualization on the internet. *Transactions in GIS*, 5(2):131–139, 2001.

[48] C. D. Hundhausen and S. A. Douglas. Low-fidelity algorithm visualization. *Journal of Visual Languages and Computing*, 13(5):449–470, Oct. 2002.

[49] C. D. Hundhausen, S. A. Douglas, and J. T. Stasko. A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages and Computing*, 13(3):259–290, June 2002.

[50] J. Hyvönen and L. Malmi. TRAKLA – a system for teaching algorithms using email and a graphical editor. In *Proceedings of HYPERMEDIA in Vaasa*, pages 141–147, 1993.

[51] P. Ihantola, V. Karavirta, A. Korhonen, and J. Nikander. Taxonomy of effortless creation of algorithm visualizations. In *Proceedings of the 2005 international workshop on Computing education research*, pages 123–133, New York, NY, USA, 2005. ACM Press.

[52] A. Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(2):69–97, 1985.

[53] M. Janlöv. Prediktering av rörelsen för ett objekt i arméns lägesbild med hjälp av terrängdata. Master's thesis, Teknillinen korkeakoulu, 2005.

[54] V. Karavirta. *Facilitating Algorithm Visualization Creation and Adoption in Education*. Doctoral dissertation (research rep. no. tkk-cse-a3/09), Helsinki University of Technology, 2009.

[55] D. Karssenberg, P. A. Burrough, R. Sluiter, and K. de Jong. The pcraster software and course materials for teaching numerical modelling in the environmental sciences. *Transactions in GIS*, 5(2):99–110, 2001.

[56] D. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1 –8, 2002.

[57] D. A. Keim and H.-P. Kriegel. Visualization techniques for mining large databases: A comparison. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):923–938, 1996.

[58] I. Koifman, I. Shimshoni, and A. Tal. Mavis: A multi-level algorithm visualization system within a collaborative distance learning environment. *Journal of Visual Languages & Computing*, 19(2):182–202, 2008.

[59] A. Korhonen. World wide web (www) tietorakenteiden ja algoritmien tietokoneavusteisessa opetuksessa. Master's thesis, Helsinki University of Technology, 1997.

[60] A. Korhonen. *Visual Algorithm Simulation*. Doctoral dissertation (tech rep. no. tko-a40/03), Helsinki University of Technology, 2003.

[61] A. Korhonen and L. Malmi. Matrix — Concept animation and algorithm simulation system. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, pages 109–114, Trento, Italy, May 2002. ACM Press, New York.

[62] J. Korpi. Designing pictorial symbols for situation pictures of international crisis management. In *Cartography and Art -symposium*, Vienna, Austria, January 2008.

[63] E. L. Koua, A. Maceachren, and M. J. Kraak. Evaluating the usability of visualization methods in an exploratory geovisualization environment. *International Journal of Geographical Information Science*, 20(4):425–448, 2006.

[64] K. Krippendorff. *Content analysis : an introduction to its methodology*. Thousand Oaks, CA, 2004.

[65] J. Krisp. *Geovisualization and Knowledge Discovery for Decision-making in Ecological Network Planning*. PhD thesis, Helsinki University of Technology, 2006.

[66] J. Krygier and D. Wood. *Making Maps: A Visual Guide to Map Design for GIS*. The Guilford Press, 2005.

[67] M.-J. Laakso, T. Salakoski, L. Grandell, X. Qiu, A. Korhonen, and L. Malmi. Multi-perspective study of novice learners adopting the visual algorithm simulation exercise system TRAKLA2. *Informatics in Education*, 4(1):49–68, 2005.

[68] R. Laurini and D. Thompson. *Fundamentals of spatial information systems*. Academic Press, 1992.

[69] K. Liao and D. Guo. A clustering-based approach to the capacitated facility location problem. *Transactions in GIS*, 12(3):323–339, 2008.

[70] A. Lucieer and M.-J. Kraak. Interactive and visual fuzzy classification of remotely sensed imagery for exploration of uncertainty. *International Journal of Geographical Information Science*, 18(5):491–512, 2004.

[71] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[72] P. C. Mahalanobis. On generalized distance in statistics. *Proceedings of the National Institue of Sciences in India*, 2(1):49–55, 1936.

[73] L. Malmi, V. Karavirta, A. Korhonen, J. Nikander, O. Seppälä, and P. Silvasti. Visual algorithm simulation exercise system with automatic assessment: TRAKLA2. *Informatics in Education*, 3(2):267–288, 2004.

[74] L. Malmi and A. Korhonen. Automatic feedback and resubmissions as learning aid. In *IEEE International Conference on Advanced Learning Technologies*, pages 186 – 190, 2004.

[75] L. Malmi, A. Korhonen, and R. Saikkonen. Experiences in automatic assessment on mass courses and issues for designing virtual courses. In *Proceedings of The 7th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education, ITiCSE'02*, pages 55–59. ACM Press, New York, 2002.

[76] M. J. McCullagh and C. G. Ross. Delaunay triangulation of a random data set for isarithmic mapping. *The Cartographic Journal*, 17(2):93–99, 1980.

[77] M. Monmonier and H. J. de Blij. *How to Lie with Maps*. University of Chicago Press, 1996.

[78] G. Mountrakis and K. Gunson. Multi-scale spatiotemporal analyses of moose-vehicle collisions: a case study in northern vermont. *International Journal of Geographical Information Science*, 23(11):1389–1412, 2009.

[79] A. T. MURRAY and V. ESTIVILL-CASTRO. Cluster discovery techniques for exploratory spatial data analysis. *International Journal of Geographical Information Science*, 12(5):431–443, 1998.

[80] N. Myller, R. Bednarik, E. Sutinen, and M. Ben-Ari. Extending the engagement taxonomy: Software visualization and collaborative learning. *ACM Transactions on Computing Education*, 9(1):1–27, 2009.

[81] T. Naps, S. Cooper, B. Koldehofe, C. Leska, G. Rößling, W. Dann, A. Korhonen, L. Malmi, J. Rantakokko, R. J. Ross, J. Anderson, R. Fleischer, M. Kuittinen, and M. McNally. Evaluating the educational impact of visualization. *SIGCSE Bulletin*, 35(4):124–136, 2003.

[82] T. L. Naps. JHAVÉ: Supporting Algorithm Visualization. *Computer Graphics and Applications, IEEE*, 25(5):49–55, 2005.

[83] T. L. Naps, G. Rößling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger, and J. Ángel Velázquez-Iturbide. Exploring the role of visualization and engagement in computer science education. *SIGCSE Bulletin*, 35(2):131–152, 2003.

[84] J. Nikander. *Software Visualization in Teaching Spatial Data Algorithms*. Licentiate's thesis, Helsinki University of Technology, 2009.

[85] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, 2000.

[86] E. Orava. Terrain analysis for military use. Master's thesis, Helsinki University of Technology, 1997.

[87] D. O'Sullivan and D. J. Unwin. *Geographic Information Analysis*. Wiley, 2003.

[88] F. Paas, A. Renkl, and J. Sweller. Cognitive load theory and instructional design: Recent developments. *Educational Psychologist*, 38(1):1–4, 2003.

[89] C. Paniconi, S. Kleinfeldt, J. Deckmyn, and A. Giacomelli. Integrating gis and data visualization tools for distributed hydrologic modeling. *Transactions in GIS*, 3(2):97–118, 1999.

[90] M. Patton. *Qualitative Research and Evaluation Methods*. Sage Publications, 2002.

[91] T. Pei, C. Zhou, A.-X. ZHu, B. Li, and C. Qin. Windowed nearest neighbour method for mining spatio-temporal clusters in the presence of noise. *International Journal of Geographical Information Science*, 24(6):925–948, 2010.

[92] W. Pierson and S. Rodger. Web-based animation of data structures using JAWAA. In *Proceedings of the 29th SIGCSE Technical Symposium on Computer Science Education*, pages 267–271. ACM Press, New York, 1998.

[93] F. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 1985.

[94] B. A. Price, R. M. Baecker, and I. S. Small. A principled taxonomy of software visualization. *Journal of Visual Languages and Computing*, 4(3):211–266, 1993.

[95] R. Purves, D. Medyckyj-Scott, and W. Mackaness. The e-mapscholar project–an example of interoperability in giscience education. *Computers & Geosciences*, 31(2):189–198, 2005.

[96] F. Ren and M.-P. Kwan. Geovisualization of human hybrid activity-travel patterns. *Transactions in GIS*, 11(5):721–744, 2007.

[97] C. Rinner and J. P. Taranu. Map-based exploratory evaluation of non-medical determinants of population health. *Transactions in GIS*, 10(4):633–649, 2006.

[98] A. H. Robinson, J. L. Morrison, P. C. Muehrcke, A. J. Kimerling, and S. C. Guptil. *Elements of Cartography*. Wiley, 1995.

[99] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, 1989.

[100] H. Samet. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*. Addison-Wesley, 1990.

[101] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann, 2006.

[102] J. Sander, M. Ester, H. Kriegel, and X. Xu. Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.

[103] B. Schneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *IEEE Symposium on Visual Languages*, pages 336–343, 1996.

[104] H. Seppänen. Maavoimien tilannekuvan muodostamisen automatisointi ja nopeuttaminen paikkatiedon avulla. Master's thesis, Teknillinen korkeakoulu, 2005.

[105] C. A. Shaffer, M. L. Cooper, A. J. D. Alon, M. Akbar, M. Stewart, S. Ponce, and S. H. Edwards. Algorithm visualization: The state of the field. *Trans. Comput. Educ.*, 10(3):9:1–9:22, 2010.

[106] S.-L. Shaw, H. Yu, and L. S. Bombom. A space-time gis approach to exploring large individual-based spatiotemporal datasets. *Transactions in GIS*, 12(4):425–441, 2008.

[107] M. Shneerson and A. Tal. Interactive collaborative visualization environment for geometric computing. *Journal of Visual Languages & Computing*, 11(6):615–637, 2000.

[108] T. A. Slocum, R. B. McMaster, F. C. Kessler, and H. H. Howard. *Thematic Cartography and Geographic Visualization*. Prentice-Hall, 2004.

[109] J. T. Stasko. TANGO: A framework and system for algorithm animation. *IEEE Computer*, 23(9):27–39, 1990.

[110] J. T. Stasko, J. B. Domingue, M. H. Brown, and B. A. Price. *Software Visualization: Programming as a Multimedia Experience*. MIT Press, Cambridge, MA, 1998.

[111] E. Stefanakis. Net-dbscan: clustering the nodes of a dynamic linear network. *International Journal of Geographical Information Science*, 21(4):427–442, 2007.

[112] M. Takatsuka and M. Gahegan. Geovista studio: a codeless visual programming environment for geoscientific data analysis and visualization. *Computers & Geosciences*, 28(10):1131–1144, 2002.

[113] J. Thomas and K. Cook, editors. *Illuminating the Path: The Research And Development Agenda for Visual Analytics*. IEEE, 2005.

[114] J. J. Thomas and K. A. Cook. A visual analytics agenda. *IEEE Computer Graphics and Applications*, 26(1):10–13, 2006.

[115] D. Tomlin. *Geographic Information Systems and Cartographic Modeling*. Prentice Hal, 1990.

[116] J. W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.

[117] E. Valanto. Paikkatietoon perustuvan taistelumallin simulaattori. Master's thesis, Teknillinen korkeakoulu, 2005.

[118] P. van Oosterom and T. Vijlbrief. The spatial location code. In *the Seventh International Symposium on Spatial Data Handling*, 1996.

[119] O. Špatenkova and A. Stein. Identifying factors of influence in the spatial distribution of domestic fires. *International Journal of Geographic Information Science*, 24(6):841–858, 2010.

[120] J. P. Wilson and A. S. Fotheringham. *The Handbook of Geographic Information Science*, chapter Geographic Information Science: An Introduction, pages 1–8. Blackwell Publishing Ltd, 2008.

[121] M. Worboys and M. Duckham. *GIS: A Computing Perspective*. Taylor & Francis, 2004.

[122] N. Xiao, C. A. Calder, and M. P. Armstrong. Assessing the effect of attribute uncertainty on the robustness of choropleth map classification. *International Journal of Geographical Information Science*, 21(2):121–144, 2007.

Aalto-DD 134/2012

ISBN 978-952-60-4826-0
ISBN 978-952-60-4827-7 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934
ISSN 1799-4942 (pdf)

**Aalto University**
**School of Science**
**Department of Computer Science and Engineering**
**www.aalto.fi**

BUSINESS +
ECONOMY

ART +
DESIGN +
ARCHITECTURE

SCIENCE +
TECHNOLOGY

CROSSOVER

**DOCTORAL
DISSERTATIONS**