Cartographic knowledge-based generalisation of spatial features

Author:
Kazemi, Sharon

Publication Date:
2008

DOI:
https://doi.org/10.26190/unsworks/23688

CARTOGRAPHIC KNOWLEDGE-BASED GENERALISATION OF SPATIAL FEATURES

by

Sharon Kazemi

A dissertation submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Surveying and Spatial Information Systems

4th December 2008

THE UNIVERSITY OF

NEW SOUTH WALES

School of Surveying and Spatial Information Systems

The University of New South Wales

Sydney 2052 NSW Australia

THE UNIVERSITY OF NEW SOUTH WALES
Thesis/Dissertation Sheet

Surname or Family name: **Kazemi**

First name: **Sharon** Other name/s:

Abbreviation for degree as given in the University calendar: **PhD**

School: **Surveying & Spatial Information Systems** Faculty: **Engineering**

Title: **CARTOGRAPHIC KNOWLEDGE-BASED GENERALISATION OF SPATIAL FEATURES**

ABSTRACT

This study has developed a framework for generalisation of geographical features using a knowledge-based solution. The proposed method consists of three major components:

- *Generalisation Framework* - a detailed generalisation framework for deriving multi-scale spatial data has been developed based on an assessment of existing generalisation systems. Generalisation techniques were applied over a test area in the Australian Capital Territory of Australia in order to generalise 1:250,000 national topographic data to produce small scale maps through derivative mapping. Also, a framework for segmentation and generalisation of raster data was developed. Test results shows that all objects derived from the generalisation of land use data over Canberra, Australia, were well classified and mapped with a classification accuracy of 85.5%.

- *Cartographic Knowledge-based Generalisation* – this was achieved through an international cartographic generalisation survey that collected inputs of several national mapping agencies, state mapping agencies and a number of software vendors. The findings from the survey are formulated as a series of cartographic rules to propose and implement a knowledge-based generalisation solution.

- *Generalisation Expert System* - acquired knowledge is utilised to build a knowledge-based solution: a 'Generalisation Expert System' (GES) developed in the Java-Python-C programming environments for the delivery of generalised geographical features. Its capabilities are demonstrated in a case study through generalising several line and polyline databases over the study area. The tests demonstrated that the algorithms implemented in GES are able to extract characteristic vertices on the original entity lines and polylines (e.g. for roads) while excluding non-characteristic ones so as to reduce insignificant computation. This study has demonstrated reasonable improvements in Vertex Reduction, Classification and Merge, Enhanced Douglas-Peucker and Douglas-Peucker-Peschier algorithms.

The test results also demonstrated that the developed methodology improves the efficiency of line and polyline generalisation. This thesis aims to contribute to generalisation system design and the production of a clear framework for users. Experiments described in the study can be applied to real world problems such as the generalisation of road networks and area features using GES. Future research should be directed towards developing web mapping platforms with generalisation functionality at varying scales.

FOR OFFICE USE ONLY Date of completion of requirements for Award:

Author's e-mail: sharon_kazemi@hotmail.com

ORIGINALITY STATEMENT

'I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.'

Signed: 

Date: 26th May 2011

COPYRIGHT STATEMENT

Signed: 

Date: 26th May 2011

AUTHENTICITY STATEMENT

'I certify that the Library deposit digital copy is a direct equivalent of the final officially approved version of my thesis. No emendation of content has occurred and if there are any minor variations in formatting, they are the result of the conversion to digital format.

Signed: 

Date: 26th May 2011

ACKNOWLEDGEMENTS

This research would not have been possible without the support, guidance, and patience of many people. I would like to acknowledge the following:

This research was sponsored by the Faculty of Engineering's Research PhD Scholarship and Supplementary Engineering Award (SEA) at the University of New South Wales (UNSW) during 2005-2008.

I would like to thank my supervisor Associate Professor Samsung Lim for his support and the opportunity to carry out this research project.

I wish to also to thank Professor Chris Rizos, Head of the School of Surveying and Spatial Information Systems for his supports in the form of feedback, critical comments, constructive suggestions and careful amendments which have contributed to the improvement of the thesis.

I would like to sincerely thank the anonymous examiners and the independent reviewer (Professor Bill Wilson of School of Computer Science and Engineering, UNSW) for their insightful comments, constructive suggestions and thorough reviews of the thesis.

I am grateful to several reviewers including Dr Alan Forghani of Murray-Darling Basin Authority, Professor Gary Hunter of University of Melbourne, Professor Graeme Wright of Curtin University of Technology, Mr Graham Baker of Geoscience Australia and Dan Lee of ESRI Redlands. They participated in various discussions on the topic of generalisation and provided constructive suggestions toward presented/published works stemmed from this study.

GIS software applications including Intergraph DynaGen software, ESRI ArcGIS and Laser-Scan Clarity were made available to this study as student licence. Geoscience Australia provided the Geographic Information Systems (1:250,000 national topographic data, 1:1,000,000 Global Map), Global Positioning System (Differential GPS survey points), Remote Sensing (Landsat 7 and SPOT 4 imagery).

Special thanks are also extended to the Geographic Information Science Center (GISC) of the University of California Berkeley for hosting me as Visiting Scholar during 2006-2007.

I would like to extend my deep appreciation to my parents for their continuous support and my beautiful boys (Pedram and Payam) because this work owes much to their love, patience, support and encouragement. Last but not least my husband (Alan) provided unfailing moral support. This thesis is dedicated to them.

ABSTRACT

Custodians of Geographic Information Systems (GIS) databases currently provide timely and high quality spatial products to users, maintaining multiple databases at different scales for different uses. These custodians are also committed to maintaining and updating the currency of cartographic data sets. Maintaining multiple databases is resource-intensive, time consuming and cumbersome. Cartographic generalisation has primarily been used to derive smaller scale map products from these databases. This practice is based on a cartographer's skill and incurs a high cost. Thus, new approaches for automating the generalisation of spatial data to produce highly varied sets of versatile, multipurpose map products need to be developed.

This thesis develops a framework for the generalisation of geographical features using a knowledge-based solution. The proposed method consists of three major components:

Generalisation Framework - a detailed generalisation framework for deriving multi-scale spatial data has been developed based on an assessment of existing generalisation systems. Generalisation techniques were applied over a test area in the Australian Capital Territory in order to generalise 1:250,000 scale national topographic data to produce smaller scale maps through derivative mapping. The study paid particular attention to the integration and utilisation of generalisation operators in order to generalise a road network database and produce small scale maps at 1:500,000 and 1:1,000,000 from 1:250,000 scale national topographic data.

There are a number of parameters that may be used for generalisation of roads to maintain their legibility, the visual identity of each road segment and the pattern of the roads. It is well understood that generalisation of geographical features (for example, roads, rivers, or any other linear features) requires at least six key operations or processes: *classification*, *selection, elimination, simplification, typification* and *symbolisation*. These are used in the generalisation framework in this study through an assessment of a set of commercial software. The resultant maps match flawlessly with existing small-scale road maps, such as Global Map, at a scale of 1:1,000,000. The generalisation operations/algorithms, and the parameters embedded in a modern map generalisation system, deliver coherent capabilities to automate the generalisation process for more practical production

applications. Nevertheless, the technology is still emergent and requires integration of generalisation algorithms with a cartographer's intuition and skills to operate effectively within a GIS. This paves the way to develop a powerful, flexible and robust expert system capable of composing, editing, exhibiting and demonstrating an innovative method for semi-automated generalisation of geographical features.

In addition, a framework for segmentation and generalisation of raster data, '*Interactive Automated Segmentation and Raster Generalisation Framework*' (IASRGF) was developed. Test results of the IASGRF show that all objects derived from the generalisation of land use data over Canberra, Australia, were well classified and mapped. The error assessment indicates that the percentile classification accuracy is 85.5%, whereas the commission error is relatively high (38.5%). More importantly, the maximum likelihood classifier using training sites and associated ground truth data suggest that the Kappa index is 0.798, which can be interpreted as a reliable and satisfactory classification result. In order to further enhance supervised classification, a post-classification was carried out. As a result, this extra process slightly improved the overall classification accuracy, however its commission error also increased by 6%.

Cartographic Knowledge-based Generalisation - this is achieved through an International Cartographic Generalisation Survey that collected inputs from several national mapping agencies, state mapping agencies and a number of software vendors. This included cognitive cartographers' knowledge about the principles of cartographic generalisation and experience with existing generalisation platforms. The findings from the survey were used to create a series of cartographic rules to propose and implement a knowledge-based generalisation solution.

Generalisation Expert System – the automation of map generalisation requires an expert system approach that consists of four main components including knowledge acquisition, an inference engine, knowledge representation and a user interface. The acquired knowledge was then utilised to build a knowledge-based solution: a '*Generalisation Expert System*' (GES) developed in Java, Python and C programming environments for the delivery of generalised geographical features. Its capabilities are demonstrated in a case study through generalising several line and polyline databases over the study area in

Canberra, Australia. The cartographic and GIS software communities will benefit from this study through access to a set of tools, guidelines and protocols that incorporate a standardised cartographic generalisation methodology.

The results of the trials utilising GES were analysed: a series of generalisation routines were performed to assess the quality of simplification results for different spatial layers. Cartometric measures such as the total length and number of line or polyline segments were used as indices of generalisation to quantify generalisation performance for the target small scale. For example, there are 101,228 segments in 1:250,000 scale and 9,491 segments in 1:500,000 scale contours over the study area. This requires a reduction in the complexity and the density of elevation data. Changes in the representation of contour features at 1:250,000 and 1:500,000 scales as a result of generalisation were quantified. Outputs from map derivation have been analysed applying the *Radical Law*, this determines the retained number of objects for a given scale change and the number of objects of the original source map.

Testing demonstrated that the implemented algorithms in GES are able to extract characteristic vertices on the original entity lines and polylines (e.g. for roads) while excluding non-characteristic lines and polylines to reduce irrelevant computation. This study has demonstrated reasonable improvements in Vertex Reduction, Classification and Merge, Enhanced Douglas-Peucker and Douglas-Peucker-Peschier algorithms. The test results show that GES generalises line features accurately while still maintaining their geometric relations. Existing generalisation software requires advanced technical skills from users; GES however, has a basic and user friendly Graphical User Interface (GUI) which is an advantage to users with basic technical skills and understanding of spatial data management.

Changes to geographic parameters should be updated in multi-scale maps and spatial databases in near real-time. GES can be developed as a potential tool for generalising large-scale maps into smaller scales, and creating maps of different themes across a variety of scales. Test results have also demonstrated that the methodology developed improves the efficiency of line and polyline generalisation. This study aims to contribute to generalisation system design and the production of a clear framework for users.

Experiments presented in this thesis can be applied to real world problems such as the generalisation of road networks and area features using GES. Future research should be directed towards developing web mapping platforms with generalisation functionality at varying scales.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY OF TERMS

AAA	Australian Automobile Association
ADE	Application Development Environment
AGENT	Automatic Generalisation New Technology
ANN	Artificial neural networks
CCRS	Canada Centre for Remote Sensing
CLIPS	C Language Integrated Production System
COM	Component Object Model
CPG	Classification by Progressive Generalisation
DC	Dual Carriageway
DP	Douglas–Peucker
ESRI	Environmental Systems Research Institute
ETM	Enhanced Thematic Mapper
FT	Foot Track
GA	Geoscience Australia
GCP	Ground Control Points
GES	Generalisation Expert System
GIS	Geographic Information Systems
GPS	Global Positioning Systems
GUI	Graphical User Interface
IASRGF	Interactive Automated Segmentation and Raster Generalisation Framework
ICA	International Cartographic Association
IDE	Initialisation Data Editor
ISCGM	International Steering Committee for Global Mapping
KBS	Knowledge-Based Systems
LINZ	Land Information New Zealand
MR	Minor Road
MS	Multispectral
NLS	National Land Survey
NMAs	National Mapping Agencies
OS	Ordnance Survey

PR	Principal Road
SMAs	State Mapping Agencies
SR	Secondary Road
UML	Universal Modelling Language
USGS	United States Geological Survey
VT	Vehicle Track

PUBLICATIONS

Some of the work presented in this thesis appears in peer-reviewed journal and conference publications. Considerable parts of these publications are transformed into the thesis.

Kazemi, S., Lim, S. and Rizos, C. (2009a). ***Interactive and automated segmentation and generalisation of raster data***. International Journal of Geoinformatics, Vol. 5, No. 3, pp. 65-73.

Kazemi, S., and Lim, S., (2007). ***Deriving multi-scale GEODATA from TOPO-250K road network data.*** Journal of Spatial Science, 52(1), pp. 165-176.

Kazemi, S., Lim, S., and Paik H., (2009b). ***Generalisation expert system (GES): a knowledge-based approach for generalisation of line and polyline spatial datasets***. Proceedings of the Surveying & Spatial Sciences Institute Biennial International Conference, 28 September –2 October 2009, Adelaide , Australia, pp 717-729.

Kazemi, S., Lim, S., and Ge, L., (2007). ***An international survey research: cartographic generalisation practices in mapping agencies***. Proceedings of the SSC 2007 Spatial Intelligence, Innovation and Praxis: The National Biennial Conference of the Spatial Sciences Institute, 4-8 May 2007, Hobart, Tasmania, pp. 537-556.

Kazemi, S., Lim, S. and Ge. L., (2005). ***Integration of cartographic knowledge with generalisation algorithms.*** Proceedings of the 2005 IEEE Geoscience and Remote Sensing Symposium, Volume 5, 25-29 July 2005, Seoul, Korea, pp. 3502 – 3505.

Kazemi, S., and Lim, S., (2005). ***Generalisation of road network using Intergraph DynaGen™ system.*** Proceedings of the SSC 2005 Spatial Intelligence, Innovation and Praxis: The National Biennial Conference of the Spatial Sciences Institute, 12-16 September 2005, Melbourne, Australia, pp. 1285-1296.

Kazemi, S., Lim, S., and Rizos, C., (2004). ***A review of map and spatial database generalisation for developing a generalisation framework***. Proceedings of the XXth Congress of the International Society for Photogrammetry and Remote Sensing, Istanbul, Turkey, 12-23 July 2004, pp. 1221-1226.

CHAPTER 1: INTRODUCTION

1.1 OVERVIEW

National mapping agencies (NMAs) and commercial producers of digital spatial databases are currently providing timely and high quality spatial products to users. They maintain multiple databases at different scales for different uses. They are also committed to maintaining a set of cartographic data with synchronised updates. Maintaining multiple databases is resource-intensive, time consuming and cumbersome (Arnold and Wright, 2005). Around the globe, the NMAs are responsible for maintaining the primary topographic databases, covering their respective territories. Cartographic generalisation has been used to derive smaller scale map products from these databases. This practice is based on a cartographer's skill and incurs a high cost. Thus, more efficient methods for generalisation are essential for production of highly varied sets of versatile, multipurpose map products derived from geospatial databases (Sarjakoski *et al.*, 2002).

When a mapping agency updates the information in its most detailed database (e.g. seamless master), usually more than one map is affected by any given change. This then requires the changes to be transferred to the relevant smaller scale databases (Dutton, 1999). This is a major challenge for NMAs and other map producers (Sarjakoski *et al.*, 2002). A seamless master database should offer capabilities to derive various types of maps at different scales, e.g. at scales ranging from 1:250,000 to 1:10,000,000.

Derivative mapping has been identified as an active research and development area for NMAs, industry and academia in order to address requirements for evaluation and validation of generalisation systems. Several researchers have highlighted a need to evaluate and validate existing generalisation tools rather than developing new generalisation algorithms and systems (Muller *et al.*, 1995; Visvalingam and Herbert, 1999; Kazemi *et al.*, 2004b; Mackaness *et al.*, 2007; Kazemi and Lim, 2007a). In addition, many researchers (Shea and McMaster, 1989; and Meng, 1997) have highlighted the questions of 'how', 'when' and 'why' to generalise. Kazemi *et al.*, (2005a) noted that, for automation of the map generalisation process, it is necessary to integrate generalisation algorithms with the cartographer's intuition and skills within a

Geographic Information System (GIS) that facilitates the selection of appropriate techniques for map and database generalisation tasks, such as feature displacement. With the increased integration of human knowledge; computer scientists, mapping specialists and cartographers have developed generalisation operators and algorithms for automated generalisation (e.g. Kreveld and Peschier, 1998; Iwaniak, *et al.*, 2004).

The majority of existing standard GIS software applications support generalisation of both line and area features. A comprehensive review of line and area generalisation is provided by Kazemi *et al.*, (2004a). Many generalisation processes exist but the key operators are selection, simplification, classification and symbolisation of features. Line feature generalisation often involves the use of geometric operators such as selection, elimination, merge, displacement, aggregation and symbolisation. However, there is no standard definition of generalisation, since it depends on *ad hoc* applications (AGENT, 1999 and Haire, 2001).

Generalisation of linear features is very important because lines represent the majority of map features. Therefore, linear feature generalisation plays an important role in GIS (Barrault, 1995; Skopeliti and Tsoulos, 2001). A review of the application of generalisation with a special emphasis on road network generalisation from 1:250,000 scale to 1:500,000 and 1:1,000,000 scales, is provided in Kazemi and Lim (2007b). Commercial generalisation systems for this type of mapping are limited to four major vendors: 1) ArcGIS™ Generalise™ (ESRI), 2) DynaGen™ (Intergraph), 3) Clarity™ (Laser-Scan), and 4) CHANGE (Institute for Cartography at Hanover University).

The author's experience has concluded that Laser-Scan Clarity™ is a very complex system which only experts, and perhaps only the designers of the system, are able to adapt and use for generalisation purposes. This was also reported by Lecordix *et al.*, (2005). It is not a generic system for automatic generalisation. Individual users would not be able to adapt it easily for their own needs, using their own specifications. The provision of technical support and training is a key factor in successfully overcoming these weaknesses in future versions of such systems.

The Intergraph generalisation system has been tested for various generalisation tasks in several countries, but not in Australia. Iwaniak and Paluszynski (2001) investigated the application of cartographic skills, together with Intergraph MGE Map Generaliser™ software tools, to automate the generalisation of topographic maps of urban areas from 1:10,000 to 1:50,000 scale. The system uses the MGE Map Generalise™ and a rule-based system implemented in the C Language Integrated Production System (CLIPS). This system was developed by Purdue University for controlling the generalisation process through development of a knowledge-based expert system that generates results similar to those obtained with the manual procedure.

One of the key features of the MGE Map Generalise™ system is its efficient handling of conflict resolution among objects. The expert system enables the integration of generalisation operations, generalisation rules and manual intervention. The authors suggested that this approach warranted further research. Iwaniak *et al.*, (2004) applied the same approach to the generalisation of a Polish topographic database at a scale of 1:10,000 in order to derive 1:50,000 databases. A key feature of the system is its powerful ability for handling conflict resolution among objects. The authors recommended that the integration of generalisation operations, rules, and manual intervention should be pursued. However, undertaking this approach with a large database at the state or country level is not cost effective. Therefore, an automatic/semi-automatic method was called for (Paluszynski and Iwaniak, 2001).

The above GIS software packages support the generalisation of line features that motivated this research for a generalisation framework to derive multiscale mapping products. It focuses on integration and utilisation of generalisation operators to generalise a road network database from 1:250,000 scale national topographic data to produce small scale maps at 1:500,000 and 1:1,000,000. Assessments of the derived maps are based on the *Radical Law* approach (Pfer & Pillewizer, 1966; Muller, 1995; Kazemi and Lim, 2007b). This research demonstrates the principle for roads, applying a conceptual generalisation framework and then expanding the application to other features.

To date, many frameworks/workflows have been developed for the generalisation of cartographic data. It seems that many of these frameworks are generic and do not offer a total solution for the operational environment. The aim of this research is to develop a workflow specifically for derivation of multiple scale maps from a master road network database. A large portion of this framework may be considered generically applicable. However, in this study all the processes are considered specifically for road generalisation.

There are a number of parameters that could be used for generalisation of roads to maintain the legibility, the visual identity of each road segment and its pattern. Out of the six generalisation parameters (reducing complexity, maintaining spatial accuracy, maintaining attribute accuracy, maintaining aesthetic quality, maintaining a logical hierarchy, and consistently applying generalisation rules) of Shea and McMaster (1989), the following four are taken into consideration in this study due to their relevancy: *Congestion/Legibility, Coalescence*, *Imperceptibility* and *Length/Distance*. Some of the Shea and McMaster parameters have also been used by Kreveld and Peschier (1998). Upon a review of the literature and consulting with key scientists and commercial developers of generalisation tools (e.g. Lee, 2003), it was concluded that generalisation of a network (for example, roads, rivers, or other linear features) requires at least six key operations/processes: *Classification, Selection, Elimination, Simplification, Typification, and Symbolisation.* These six processes are used in the generalisation framework of this research, as discussed in the next section.

This thesis presents a generalisation framework to derive multiscale spatial data. It focuses on the integration and utilisation of generalisation operators with cartographic knowledge. The original idea stemmed from reviewing worldwide research in generalisation by academia and industry. It includes a review of the concepts of cartographic generalisation, model and map generalisation, generalisation operators, generalisation software packages and data structures with special reference to derivative mapping from a seamless database as well as the benefits of generalisation in relation to spatial and temporal data access. Primary research and development in the field of generalisation was conducted and is presented.

1.2 BACKGROUND OF GENERALISATION

In this section, some recent studies relevant to this research are highlighted. Research initiatives and contributions by generalisation experts from Australia, Canada, China Finland, Germany, The Netherlands, Switzerland, USA and the United Kingdom are discussed to provide an overview of the global research agenda and findings. Several typical examples of automatic cartographic generalisation are discussed.

In the last decade there has been a rapid increase in the development and application of semi-automatic techniques for map and database generalisation. Computer scientists, mapping specialists and cartographers are involved, with increased integration of human knowledge with generalisation operators (algorithms) for automation of generalisation (Paluszynski and Iwaniak, 2001; Thomson and Brooks, 2002). Several generalisation processes exist but the key operators include feature selection, feature simplification, feature classification and feature symbolisation. The process of generalisation of linear features often involves the use of geometric operators: selection, elimination, merging, displacement, aggregation and symbolisation. Detailed descriptions of these operators are presented in McMaster and Shea (1992), Muller *et al.,* (1995), Harrie (2001), Lee (2003) and Kereld (2001). Some recent examples relevant to this research are:

- Kreveld and Peschier (1998) developed a new approach to decide which roads should be selected when generalising road network maps. The approach considered not allowing roads to be too close to each other, avoiding detours between important points, and giving priority to larger roads.
- McKeown *et al.,* (1999) improved the line simplification operation of the Douglas and Peucker algorithm to prevent the most common topological and terrain related anomalies (artefacts). This resulted in reduction of the amount of manual intervention required by incorporating topology checks for self-intersection and terrain checks to ensure that the relationship between the object and related terrain were preserved in a virtual world production environment, while these checks can be applied in other application environments.
- Skopeliti and Tsoulos (2001) developed a methodology for the parametric description of line shapes and the segmentation of lines into homogeneous segments along with measures for the quantification of shape change due to generalisation. They stated that measures that describe positional accuracy are

computed for manually generalised data or cartographically acceptable generalisation results. Skopeliti and Tsoulos's study uses a knowledge-based approach for the generalisation of linear features with special emphasis on coastlines through the use of assessment tools, namely; global routines (Douglas Peucker), local processing routines, routines taking into account structure (ESRI bend simplify by Wang and Muller, 1998). In their study four conditions were established, including very smooth, smooth, sinuous and very sinuous. Skopeliti and Tsoulos's assessment was based on human interpretation but failed to quantify the assessment. This certainly impacts on quality of generalisation by affecting areas such as positional accuracy (displacement), attribute accuracy (classification and aggregation) and completeness.

- Harrie (2001) applied a graphic generalisation process that used the generalisation operators (simplification, smoothing, exaggeration and displacement) on the scale of 1:10,000 to derive 1:50,000 scale map of roads and buildings using the LAMPS2 software. The method was focused on readability and clarity of maps while preserving the characteristics of data that are of use in real time navigation systems (e.g. transportation, mobile mapping). Also, Bakker (1997) developed a framework for the re-engineering of Dutch topographic map production at the scale of 1:10,000 scale databases to derive 1:25,000 scale maps. This provides a more structured database for GIS applications in the Netherlands. Furthermore, Bengtson (2001) implemented a new automated generalisation approach for topographic maps of Denmark at a scale of 1:10,000 to generate 1:50,000 map products within the LAMPS2 environment. The process developed for generalisation of roads supports the collapse and re-establishment of the connections between roads associated with the original centreline and the derived centreline.
- Thomson and Brooks (2002) employed perceptual grouping principles for generalisation of road and river networks to automatically produce the *National Atlas of Canada*. Perceptual grouping techniques use a fundamental set of tools for understanding images and maps based on identification of basic 'natural' elements in images, particularly images designed for human perception such as maps and diagrams. This intermediate level of structure more closely corresponds to real world elements and allows the formulation of map features.

Their findings indicate that this technique is potentially applicable to other network types such as powerlines and pipelines.

- Oosterom (2005), however, criticised these types of approaches as time consuming, hence he introduced the reactive-tree data structure for line simplification that is applicable to seamless and scaleless geographic databases. There is still, however a need for the cartographer's input in generalising lines/curves to make them fit for use. The reactive-tree approach was used to decide which feature to be removed when performing the aggregation and merging operations.
- Generalisation of roads in a holistic manner requires a number of parameters including generalisation operations, cartographic knowledge, the contextual information of features, topology, thematic properties (e.g. length, width and connectivity). Lee (2003) and Lee and Hardy (2006) have designed and implemented a set of generalisation framework and tools. According to these authors as well as Kazemi and Lim (2007), generalisation of a network (e.g. roads and rivers) requires at least six key operations/processes: *Classification, Selection, Elimination, Simplification, Typification, and Symbolisation.* These are used in the development of a conceptual framework for derivation of multiple scale maps from a master road network database in this study as discussed in **Chapters 4, 5** and **7**.

To build an automatic generalisation tool, Lee (2003) highlighted three major areas that needed development:

- *A well-designed database,* which provides a platform to support data derivation, generalisation, symbolisation, and updating. The idea is to associate geographic features to multiple scales and maintain cartographic quality of spatial data products. If necessary, pre-generalised geometry of multiple representations can be stored in a computer for fast retrieval. Designing such a database for generalisation is not an easy task. For instance, ArcGIS™ supports the idea by providing database capabilities within object-oriented map production software, by allowing scale tags in the attributes, and allowing pre-generalised geometry as additional layers (Lee, 2003). However, Jones *et al.,* (2001) are concerned that a major challenge in pre-generalising map data is ensuring topological integrity among map features.

Powerful tools are required to extract data based on given criteria in order to generate multiple scale products from a master database.

- *The data extraction task* is relatively straightforward, but requires a powerful technique to retrieve and transform data quickly from a very large seamless database. For example, the user can define an area of interest, then the generalisation system will take that area of interest and extract it from the master database.
- *A set of automatic generalisation tools* and a set of efficient post-editing and cartographic editing tools from simplification, aggregation, to displacement in a logical sequence in order to produce high quality maps.

Some researchers (e.g. Kilpelainen, 1994 and 2000; Ruas and Plazanet, 1996; Sarjakoski *et al.,* 2002; Kazemi *et al.,* 2004a) believe that NMAs and other spatial information providers/receivers should work with GIS software developers to build a powerful universal generalisation tool. The work being completed by a number of GIS software companies is significantly focused on research and development on generalisation, and these developments will be reviewed in **Chapter 2**.

Some national mapping and research organisations have evaluated many of these systems, and have found that drawbacks of existing generalisation techniques include the need for high levels of interactivity, inconsistent performance in different parts of the same feature, and a distinct lack of flexibility. For example, ESRI ArcGIS™ and Intergraph DynaGen™ software generalisation capabilities are evaluated in the course of this thesis. A number of major bottlenecks in relation to generalisation are identified and suggestions are made to provide feasible strategies for their solution. Practical strategies for road generalisation are introduced and discussed in detail in **Chapters 2** to **7**. In relation to this, it is important to note that knowledge development of guidelines plays an integral role in the development of semi-automatic generalisation systems and contributes to the efficiency of this operation in a map production environment.

1.3 STUDY QUESTIONS

In order to address the research challenges a literature review was conducted. The need for evaluation and validation of existing generalisation tools was identified. Based on an

assessment of popular generalisation tools it also became apparent that the technology is still developing, and that it requires an integration of generalisation algorithms with the cartographer's intuition and skills within a GIS. The study questions therefore are:

- What are the capabilities of existing generalisation systems?
- What does a map/database generalisation framework offer the GIS community?
- Why does map/database generalisation require a systematic framework for simplifying line and polyline databases?
- How does heuristic natural knowledge transfer from cartographers take place when constructing a generalisation expert system?

These four questions can be examined from different perspectives.

1.4 METHODOLOGY

This thesis describes a methodology to derive multiple scale maps from a master database in a manner that highlights a framework to be used in a map production application. One of the main objectives of this study is to develop a knowledge-based solution known as 'Generalisation Expert System (GES)'. A brief description of key steps undertaken in building a GES and its components are presented in **Chapter 7**. GES uses operations, algorithms and knowledge-based rules. Key components and methods of this research are shown in **Figure 1.1**.

The research methodology consists of three major components:

- Cartographic knowledge acquisition, which was completed through a survey, resulting in production of a series of cartographic rules to build a GES (**Figure 1.1a)**;
- An assessment of existing generalisation systems as a test bed (**Figure 1.1b)**; and
- Collection and processing of various spatial datasets (**Figure 1.1c**).

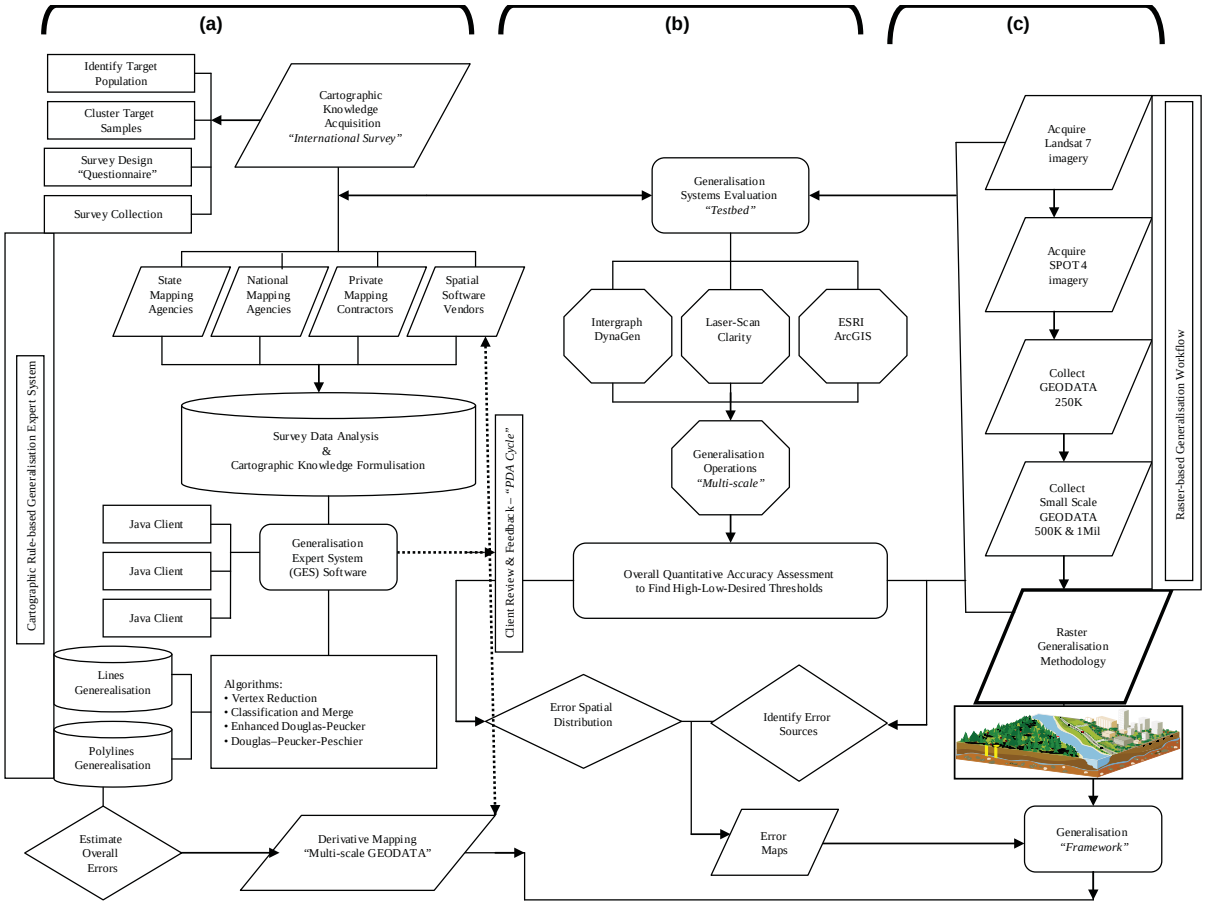


Figure 1.1 Overview of research methodology

1.4.1 Activities

Having defined the study questions, the following steps define the methodology that attempts to demonstrate the efficacy of a road network generalisation approach with 1: 250,000 roads data in this study. Generalisation techniques were applied in a test in order to generalise 1:250,000 national topographic data to produce small scale maps through derivative mapping. The activities undertaken were:

- Definition of the objectives, which deals with the development of the methodology and its implementation for a range of spatial features.
- Selection of suitable hardware and software to perform the processing.
- Selection of an appropriate study area.
- Selection and acquisition of the datasets and georeferencing of the datasets.
- Spatial data processing and construction of a database.

- A detailed review and evaluation of current generalisation methods and solutions.
- Developing a conceptual methodology to generalise 1:250,000 national topographic data to 1:500,000 and 1:1,000,000 scales.
- Qualitative and quantitative analysis and accuracy assessment of the results.
- Cartographic knowledge acquisition by undertaking an international cartographic generalisation survey in order to build a rule-based expert system.
- Cartographic knowledge encoding to construct a knowledge-based solution: the GES for the delivery of simplified spatial data.
- Demonstration of the GES application for spatial database generalisation for various spatial features.
- Assessment of the overall generalisation performance using referenced maps and experienced cartographers' feedback.

1.4.2 Study Site and Resources

A part of Australia's capital city Canberra, covering 23,630 km², was chosen as the study area. The study area was chosen because it offers a mixture of different roads types, and because of ease of access to the data (requiring minimum resources for data collection purposes). The approach and methods used in this thesis are discussed further in **Chapter 3**.

The main reference spatial datasets used in this research includes Geoscience Australia's 1:250,000 national topographic data, the International Steering Committee for Global Mapping's (ISCGM) 1:1,000,000 Global Map, a set of Global Positioning System (GPS) survey points and Landsat 7 imagery.

ArcGIS™ and Intergraph DynaGen™ were used as key the GIS software. The GES was interfaced with ArcGIS™. Other software, including ERDAS IMAGINE™ and Laser-Scan Clarity™, were also used. GES implementation was carried out in the Java-Python-C programming environment; selected for its user-friendliness, flexibility, interface capabilities, high level programming language, and the familiarity of the author with these languages.

1.5 CONTRIBUTIONS OF THE THESIS

The thesis contributions are essentially in three main areas, including *theoretical*, *methodological* and *empirical findings* resulting from the design, implementation and evaluation of a generalisation system (**Figure 1.2**).

To support novel and original contributions of this study, the author's work appeared in peer-reviewed conference and journal papers (**Publications**). Considerable parts of the publications are incorporated into the thesis. The contributions offered from the relevant chapters are outlined below:

- Development of a detailed generalisation framework to produce small scale maps through derivative mapping, applying an assessment of existing generalisation systems - *theoretical and methodological* (**Chapters 2**, **4** and **5**),
- Building a framework for a heuristic natural knowledge transfer from cartographers using an international Cartographic Generalisation Survey - *theoretical and methodological* (**Chapter 6**), and
- Construction of the GES for delivering automated generalisation of lines and polylines. Results demonstrated that the implemented algorithms led to an improvement of efficiency of generalisation - *empirical findings* (**Chapter 7**).

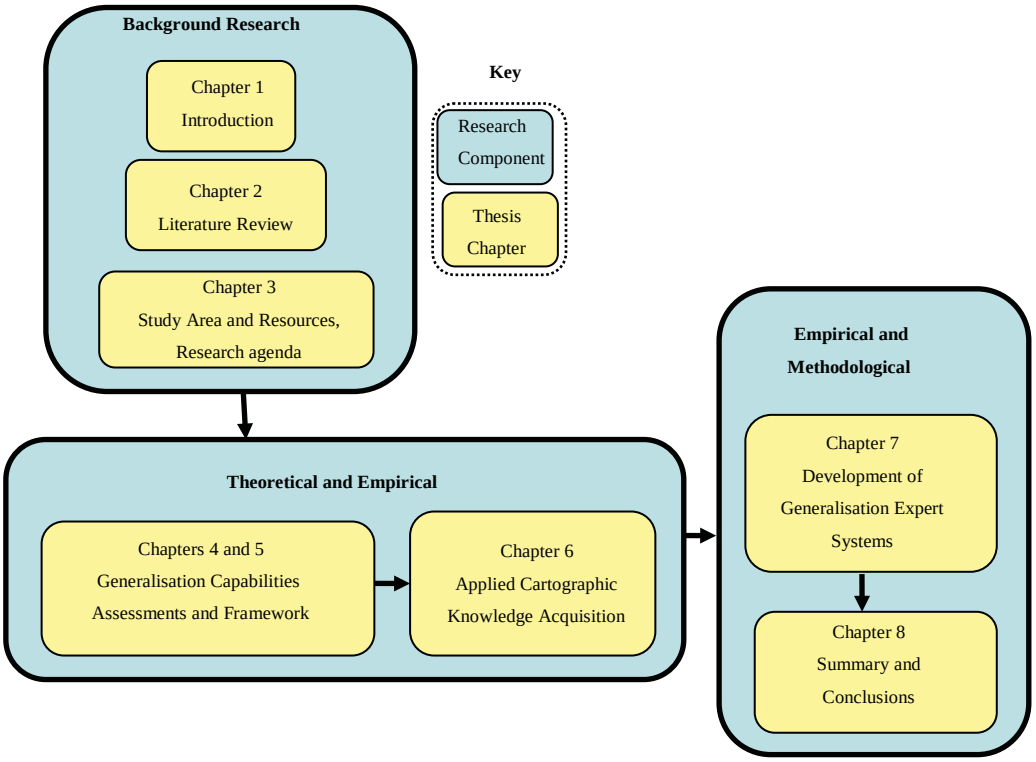


Figure 1.2 Structure of thesis and research components

1.6 OUTLINE OF THE THESIS

The thesis is organised into eight chapters. In this first chapter the study has been placed in a general context, the basic questions, study objectives and proposed methodology have been introduced, and a brief description of the study area, datasets and computing software is presented.

Chapter 2 outlines the major concepts which form the basis for this study. It reviews generalisation methods in the literature on GIS, and covers perspectives of cartographic generalisation, model and map generalisation, generalisation operators, and generalisation software packages and data structures with special reference to derivative mapping from a national seamless database. It also details the benefits of generalisation in relation to spatial/temporal data access, and highlights future challenges in the field of generalisation. It then describes some of the relevant work being done on map generalisation by various authors in relation to derivation of multiple scale spatial databases and maps from a seamless database.

Chapter 3 describes the study area, and presents the available data including GIS data, together with the hardware and software, and describes the development of a database.

Chapter 4 presents a generalisation methodology to derive multiscale spatial data through an evaluation of mapping software (ArcGIS™) that was used as a testbed based on the principles of generalisation. It focuses on integration and utilisation of generalisation operators in order to generalise a road network database, in order to produce small scale maps at 1:500,000 and 1:1,000,000 from 1:250,000 national topographic data that led to the development of a framework for derivative mapping concepts.

Chapter 5 focuses on applying a systematic approach to the generalisation of a road network database using a dynamic generalisation approach through an evaluation of DynaGen™ generalisation capabilities. It is used as a testbed based on the principles of generalisation. It then compares the results derived from DynaGen™ with an earlier work with ArcGIS™ road generalisation results. It also includes details of the functionality of each process which, combined with human interaction, involves resetting the rules and numerical parameters to achieve the desired results.

Chapter 6 provides a brief discussion on expert systems and their application in GIS with a particular emphasis on cartographic knowledge capture. An expert system consists of four main components, including: (a) knowledge acquisition, (b) inference engine, (c) knowledge representation, and (d) user interface. These are briefly discussed. It then discusses the process of a heuristic natural knowledge transfer from cartographers for building an artificial intelligence system using an international Cartographic Generalisation Survey. The survey results are utilised for constructing a knowledge-based expert system. Key findings then are formulated into a series of rules as part of the conceptual spatial databases framework. This practical generalisation method delivers coherent capabilities and automates the generalisation of features as much as possible for 'derivative mapping' applications.

Chapter 7 discusses the development of a GES which is built using Java-Python-C, delivering automated generalisation of lines and polylines. A brief description of key steps undertaken in constructing the GES and its components is presented. Its capabilities are demonstrated in a case study through simplifying roads, native vegetation and elevation datasets.

Chapter 8 summarises the aims and methodology of the research, and presents key findings from the previous chapters, outlines the conclusions reached in the study, and provides recommendations for future research on this topic.

CHAPTER 2: AN OVERVIEW OF SPATIAL DATA GENERALISATION

2.1 INTRODUCTION

There is a need for dynamic generalisation capabilities in order to generate multiscale spatial databases from the single database using an automated generalisation process. The goal is to collect data once, and maintain or use it at different levels based on requirements.

For a basic understating of generalisation it is useful to consider some of the relevant definitions and specific examples related to this research:

- Generalisation '*refers to selection and simplified representation of detail appropriate to scale and/or purpose of a map*' (International Cartographic Association, 1973).
- Generalisation '*is the simplification of observable spatial variation to allow its representation on a map*' (Goodchild, 1996).
- Map generalisation is '*an information-oriented process intended to universalise the content of a spatial database for what is of interest*' (Muller *et al.,* 1995).
- Generalisation '*means a process which realises transition between different models representing a portion of the real world at decreasing detail, while maximising information content with respect to a given application*' (Weibel and Dutton, 1999).

In view of these definitions, one can clearly distinguish two focuses on cartographic and model generalisation. The first two authors' perspectives refer to improving the aesthetic and graphic visualisation of the geographic data through symbolisation of the features and map legibility, whereas the latter two definitions are concerned with reduction of the content of spatial data from a database perspective through removing or modifying feature information.

Considering all these approaches, generalisation from an automated perspective is a systematic process of removing the details from the master database with the aim to derive small scale maps and spatial data from a single database. It transforms spatial representation of the real world to an abstracted representation, resulting in a product metrically and aesthetically correct at reduced cost (Kazemi and Lim, 2007a).

Automatic generalisation refers to the generation of abstract features from a rich database through computer algorithms rather than through human judgment. It is commonly used for individual objects such as lines or polygons. Researchers (e.g. Ruas and Plazanet, 1996; Lee, 2003) believe that National Mapping Agencies (NMAs) and other spatial information providers/users should work with GIS software developers to build a universal generalisation tool.

In practice, generalisation aims to reduce complexity while maintaining accuracy levels in terms of the attribute and spatial characteristics of the features. These depend on the map's purpose and scale, adjusted to provide adequate information or facilitate efficient communication. It often leads to more desirable outputs than those of the original dataset (Joao, 1998) through the integration of automated generalisation algorithms with a cartographer's intuition and skills.

Since NMAs are moving towards building and maintaining an infrastructure database, maintenance of multiple databases at different scales or themes is generally no longer acceptable (Brooks, 2001). In order to pave the way for automated generalisation, the first step must be a close cooperation between universities, map producers, GIS software vendors and NMAs to work towards a holistic approach in concert. This includes: (1) building a national seamless 'object-oriented' database to support all map production needs, (2) deriving multiple map products in real time with minimal interactive work, (3) updating features only in the seamless database, (4) synchronising changes to all map products with no duplication of data costs, and (5) making cartographic edit capabilities only on map products not the seamless database. This allows the users to make adjustments to the output of generalisation to add, remove, or otherwise change the appearance of objects as needed, and to support web based delivery products such as Wireless Application Protocol (WAP) technology (Sarjakoski *et al.*, 2002).

Many data models for multipurpose seamless databases have been developed. For example, Land Information New Zealand (LINZ) and Geoscience Australia (GA) have built their national topographic seamless databases in such a way that they enable cartographers to apply cartographic rules on database maintenance and revision. This

results in more cost effective, and more up-to-date maps and spatial digital topographic data (Holland *et al.,* 2003).

NMAs are committed to maintaining a set of cartographic data with different scales and to synchronise the updates with other multiple scale data (Harrie, 2001). This is a major challenge for NMAs and other map/spatial data producers (e.g. Kilpelainen, 1997; Lemarie, 2003). A multipurpose spatial corporate database should offer capabilities to derive different maps at different scales from objects (e.g. topographic objects), at scales ranging from, say, 1:250,000 to 1:1,000,000. This capability is referred to as a 'derivative mapping' capability. This is composed of several stages that include indentifying user requirements, building case tools, developing a corporate spatial data model, developing Universal Modelling Language (UML), populating data into the UML, data loading into the generalisation software package, quality assurance, developing generalisation rules and semi-automatic / automatic generalisation. In particular, the user needs identification stage should focus on highlighting constraints for each generalisation.

Development of numerical methods to generate multiscale maps by utilising advanced GIS-based technologies have been attempted (e.g. McMaster and Shea, 1992; Joao, 1998; Lee 2003). Also, the release of commercial GIS generalisation tools has been well received by major NMAs (Kilpelainen, 1997; Lee, 2003). Better qualification of generalisation tools in finding reasonable solutions for deriving multiple scale data from a master database (e.g. McKeown *et al.,* 1999; Thomson and Richardson, 1999; Jiang and Claramunt, 2004) and full integration of the generalisation capability for deriving new datasets and compiling cartographic products has become inevitable (Lee, 2003).

The remainder of this chapter is organised as follows. In **Section 2.2**, differences between the database (model) generalisation and the cartographic generalisation in a GIS environment are described. In **Section 2.3**, the relevant literature on generalisation operations with special emphasis on linear features (e.g. roads) is highlighted. In **Section 2.4**, generalisation frameworks are reviewed and a conceptual generalisation model is proposed for derivative mapping from a master database with particular reference to road networks. Also this section provides an overview of past and current

developments in generalisation of raster data and describes an effort in developing a framework for segmentation and generalisation of raster data known as *'Interactive Automated Segmentation and Raster Generalisation Framework'* (IASRGF). Details of the IASRGF are discussed by Kazemi *et al.,* (2009b). This is followed by an overview of object-oriented technology that is embedded in many current GIS systems (**Section 2.5**). **Section 2.6** summarises key generalisation systems and their applications in various countries. **Section 2.7** provides an overview of generalisation frameworks in the context of measures and conditions. Finally, **Section 2.8** concludes the discussion and indicates research directions for future work.

2.2 GENERALISATION THEMES

Weibel and Jones (1998) classified generalisation into two main approaches, known as the *cartographer driven* (cartographic generalisation) approach and the *feature reactive* (database or model generalisation or conceptual generalisation) approach. This perspective is revisited here by considering other researchers' points of view.

Database generalisation filters the data through a scale reduction process, whereas the cartographic generalisation deals with representation or visualisation of the data at a required scale (Weibel and Jones, 1998). A geographic database is usually richer than cartographic information. The database should offer multiple map generations in a continually varying range of scales. The latter method uses an object-oriented data model utilising data modelling formalisms to capture the map structure of applications at a given point in time (Yang and Gold, 1997), since this requires a highly structured dataset (Brooks, 2000). The object-oriented technology enables feature definitions and storage as objects with intelligence to represent natural behaviour of the objects and the spatial relationships of features. This is based on varying scale in one representation by displaying certain objects dynamically (Zhou *et al.,* 2002; Lee 2002). This type of database can be also referred to as scale-less or scale-free with a single maintenance procedure (Muller *et al.,* 1995) that is appropriate for multipurpose applications. In this regard derivative mapping is considered the most cost-effective and efficient method for deriving multiple scale maps and spatial data from a detailed master database to satisfy the map content requirements of a specific application.

A major advantage of database generalisation is reduction in the cost and workload of the manual process once the database is highly structured and attributed. It selects the set of features or attributes and chooses an approximate level of generalisation, then employs generalisation operators/algorithms, and finally post-processes the dataset. This limits the degree of human intervention (Meng, 1997). Furthermore, reducing the number of features in database generalisation is a key task that can be accomplished by six major operations based on the geometric and semantic relationships and database constraints that are well documented in the literature (e.g. McMaster and Shea, 1989 and 1992). These comprise simplification (line generalisation), aggregation (geometric and thematic combination), symbolisation (for line, polyline and point), feature selection (elimination and deletion), exaggeration (enlargement) and displacement or moving objects (Oosterom, 1995).

In cartographic generalisation a cartographer chooses features from a larger scale map to be shown on a smaller scale map through modifications to filter out overly detailed information, while maintaining a constant density of information by considering the purposes of the map (Davis and Laender, 1999). A drawback of this approach is that the generalisation is based on a cartographer's knowledge, skills and intuition including his/her visual/aesthetic sense (e.g. clarity, readability, ease of interpretation), and the lack of extensibility for multiple representations in GIS.

The way forward is incorporation of a data modelling process as this provides a detailed description of the database structure, or the so-called *schema*. A major advantage of this approach is a reduction in spatial and semantic resolutions, which permits both spatial analysis and map production. For example, Jiang and Claramunt (2004) proposed a generalisation model of an urban street network that aims to retain the central structure of a street network by relying on a structural representation of this data employing graph modelling principles (e.g. Gross and Yellen, 1999). The proposed method provides a flexible interactive solution to a street network because it incorporates the concept of a hierarchy-based generalisation in terms of connectivity to an average path, length and measures.

Peter and Weibel (1999) identified several measures, such as size, distance and proximity, shape, topological, density, distribution, pattern and alignment, as a set of generic constraints that need to be applied to database generalisation. Therefore it is suggested that selection of appropriate algorithms and prioritising of constraints needs to be studied. A series of selection rules emerged for road networks, such as if the average segment length of a street is less than a given threshold, then it is kept in a database, otherwise it is deleted. The graph theory approach has some distinct shortcomings, for example, the geometric aspects of coalescence as well as imperceptibility, and semantic perspectives (e.g. avoiding large detours that are not clearly explained). Kreveld and Peschier (1998) used this method for road network generalisation that significantly improved the manual interventions.

2.3 GENERALISATION OPERATIONS

Automatic generalisation has been researched and implemented by different investigators (McMaster and Shea, 1989 and 1992; Kerveld, 2001; Limeng and Lixin, 2001; Buttenfield, 2002) using various generalisation operators that lead to less human involvement, time saving, and providing a practical, more manageable operation. One of the major challenges is to choose the most effective generalisation operation, and that depends mostly on the feature types to be generalised (Ruas 2001). For example, when dealing with linear features (e.g. roads), relying only on automatic processes is not an acceptable solution and calls for cartographic editing in case of two overlapping roads due to lack of distance between them or in the case of congestion and closeness. In connection to automated generalisation, Chen and Du *(*2000) presented a graphics matching technique to automate generalisation of Japanese cadastral maps. Their method is based on feature point detection, feature point matching, conflicted line detection and automated line generalisation. The method requires four points to perform line feature detection, matching and boundary line processing. Ibid (2000) pointed out that achieving quality results for automatic generalisation is impossible, so it needs human cartographer interaction to develop editing tools for modification of the errors caused by automated processing.

Bader (2001) made a clear distinction between generalisation algorithms and operations; their relationship is hierarchical. An operator defines the transformation

through a generalisation algorithm to implement the required transformation. It means that operators are independent of a particular data model, whereas algorithms are related to a given representation, a certain change of scale and a given data structure.

There are no standard definitions for generalisation operations since these depend on *ad hoc* application perspectives (Cecconi *et al.,* 2002; Kazemi *et al.,* 2004a). A comprehensive review of generalisation algorithms and operations is presented in AGENT (1999). This provides a common basis for algorithm selection for building potential prototypes and for use in a generalisation workflow. The applications of some of these operations define generic rules, whereas some are just used by cartographers in a subjective manner (Davis and Laender, 1999) due to different feature type geometry. Thus, a definition of each of the operators could have different meanings in terms of the feature type, e.g. area elimination of vegetation features and elimination of hydrographical features. Area generalisation involves the simplification of independent polygons, the aggregation of polygons through clustering and, potentially, displacement of polygons, e.g. buildings (Anders and Sester, 2000). However, other researchers' points of view on those operations are revisited here:

1. *Selection:* This is the first step for generalisation and refers to what objects are retained, selectively based on requirements such as target scale, purpose, visual clarity and constraints (Yaolin *et al.,* 2001). It also involves decision-making regarding the geographic space to be mapped, map scale, map coordinates/projections, and data variables to be mapped.
2. *Elimination:* Small features that can cause conflict in the final map and are not significant for presentation on the map need to be eliminated, e.g. small buildings, short roads, and small villages when generalised from 1:250,000 scale to 1:1,000,000 scale spatial data (GA, 2001). They will be gradually eliminated by shrinking to a point, and buildings will vanish or can fade into the background. It is notable that selection and elimination based on size or attributes are well advanced (Lee, 2003), and require a well-designed database and attributes.
3. *Simplification:* This reduces the amount of coordinate data used to represent an object (Harrie, 2001) by reducing the number of vertices. Incidentally, it also

removes unnecessary details such as extraneous bends and fluctuations from a line or an area boundary, without destroying the essential properties of the shape. For example, a straight line between two cities could indicate the connectivity between cities rather than the exact positional location of a road.

4. *Aggregation:* This merges two connected or disconnected features to create a larger object (Doihara *et al.,* 2002) within a specified tolerance of each other. It will join point and polygon elements which are close to each other, e.g. forming a water body area from a cluster of lakes or joining patches of differing vegetation into a main class of vegetation. It is suggested that the operation should treat natural features and man-made features (buildings) differently to preserve their natural look or orthogonal shapes (Lee, 2002).
5. *Collapse:* Reduces the size of the representation of an object (Davis and Laender, 1999) due to lack of space, such as collapsing areas into lines or points and parallel road lines (i.e. road polygon) into road centreline (Harrie, 2001). For instance, DynaGen™ software handles several types of feature collapse including area to line, area to point, line to point, and dual lines to single line using different algorithms to create an appropriate representation of these features (Intergraph, 2004).
6. *Typification*: In this operation the density and details of a feature will be reduced (GA, 2001) while keeping the correct impression of the original features. This is a difficult operation for conversion of lines or points or area to a smooth version.
7. *Exaggeration:* This operation increases the graphic characteristics of an object's representation (Meng, 1997) when features are too small to be perceived visually (Davis and Laender, 1999).
8. *Conflict Resolution (Displacement):* This decreases the presentation scale of data or maps since features usually conflict with each other and need displacing so that spatial conflicts are solved by moving or by distorting objects (Harrie, 2001). For successful displacement certain rules need to be developed. Road displacement operation involves the use of a number of constraints, including minimum distance to ensure the symbols are sufficiently separated to distinguish the roads as individual segments, legibility of intersections, junctions, shape to limit the admissible modifications to road geometry, and topology to preserve

the connectivity in a road network, positional accuracy that is indirectly related to preservation of shape and to avoid unnecessary shifting.

9. *Smoothing:* This operation alters and adjusts a feature's geometry or appearance to improve its aesthetic (visual) impression and to ensure its conformity with reality. For example, area boundaries are simplified locally to present the features with the minimum amount of data (Lee, 1992).
10. *Refinement:* This process mainly deals with the visualisation point of view as Davis and Laender (1999) stated:*'…refinement discards less significant objects which are close to more important objects in order to preserve the visual characteristics of the overall representation but with less information density'*. Also r*efinement* modifies the characteristics of a symbol in order to make it more adequate for visualisation at smaller scales.
11. *Classification:* This is the systematic process of creating graphic marks, which represent the objects and phenomena to be mapped. It involves placing objects in groups according to similar properties (i.e. measures of proximity and common attributes) through sharing similar geographic characteristics into a new, higher-level feature class and representing them with a new symbol (Meng, 1997). It reduces the complexity and improves the organisation of a map through the database model by taking into consideration the typographical parameters, e.g. assigning the same attribute to all features on the map.
12. *Symbolisation:* This adopts the visual characteristics of the appearance of objects (Davis and Laender, 1999) by using a set of marks, symbols, line weights, colours, etc, to present real world phenomena on a map (Harrie, 2001).

Lee (1992) examined operational consequences and developed criteria through formalising workflow using the MG Intergraph software product for generalisation of areal, linear and point features. Results are presented at the 1:100,000 scale, and the amount of information retained in the final map was comparable to the real work. Again, there is no holistic or even ideal sequence for the utilisation of these operations. However, Monmonier and McMaster (1991) claimed there are sequential effects of the operations in cartographic line generalisation, but have not received much support from others as each of the operations may serve a specific generalisation problem. Typically the approach is to break down the generalisation process into sub-processes, and later

combine several operators to build a more robust generalisation workflow. Cecconi *et al.,* (2002) evaluated and integrated generalisation operations to improve automated generalisation for on-demand web mapping from multiscale databases. This is an excellent example of recent work on combining existing generalisation algorithms for an operational environment for dynamic map generalisation.

Many commercial GIS tools have incorporated a number of these operations. However, some of these operations (e.g. displacement, exaggeration) are still in an experimental form since they are strongly based on a cartographer's intuition. For example, ESRI's recent object-oriented ArcGIS™ software (version 9.2) provides a spatial framework to support generalisation needs by introducing geo-processing concepts and map generalisation tools that have been enhanced and implemented in a geo-processing framework (Lee, 2003). Detailed description of ArcGIS™ Generalise tool and its comparison to other available tools (e.g. DynaGen™) are provided in **Chapters 4** and **5.**

Typically current GIS software applications offer both line generalisation and area generalisation algorithms. Since the focus of this research is on road network generalisation, this chapter only highlights some of relevant literature on linear features (Skopeliti and Tsoulos, 2001). Linear feature generalisation plays an important role in GIS (Barrault, 1995; Skopeliti and Tsoulos, 2001). Several algorithms have been developed to simplify lines. McMaster (1989) classified the processing of linear features into five major algorithmic categories: (a) independent point algorithms of map generalisation where a mathematical relationship between neighbouring pairs of points is not established; (b) local processing routines that apply the characteristics of immediate neighbouring points to determine selection; (c) extended local processing routines that apply distance, angle, or number of points to search beyond neighbouring points; (d) extended local processing routines that use morphologic complexity of the line to search beyond neighbouring points; and (e) global routines that take into account the entire line or specified segment. However, none of these methods leads to an automated generalisation mechanism.

One of the revolutions in generalisation was the development of an algorithm by Douglas and Peucker (1973) and Duda and Hart (1973) (iterative endpoint fit method).

This algorithm is regarded by many as the best of the line generalisation algorithms incorporated into GIS tools (e.g. Visvalingham and Whyatt, 1993). It should be noted that the underlying concept of the Douglas and Peucker algorithm is derived from Attneave's (1954; cited by Visvalingham, 1999) theory that curvature conveys informative points on lines. Many other pieces of research have subsequently enhanced Douglas and Peucker's algorithm (e.g. Wang and Muller, 1993 and 1998; Visvalingham and Whyatt, 1993; Ruas and Plazanet, 1996) in the area of curvature approximation applying various thresholds. Oosterom (1995) criticised these types of algorithms as time-consuming, hence he introduced the reactive-tree data structure for line simplification that is applicable to seamless and scale-less geographic databases. There is still, however, a need for the cartographer's input in generalising lines/curves to make them 'fit-for-use'.

A majority of map features are represented as lines or polygons that are bounded by lines. Skopeliti and Tsoulos (2001) developed a methodology for the parametric description of line shapes and the segmentation of lines into homogeneous segments, along with measures for the quantification of shape change due to generalisation. They stated that measures for describing positional accuracy are computed for manually generalised data or cartographically acceptable generalisation results. Muller *et al.*, (1995) imply that ongoing research into line generalisation is not being managed properly. Most of the research in generalisation has focused on single cartographic line generalisation instead of working on data modelling in an object-oriented environment to satisfy database generalisation requirements. In contrast, other researchers (e.g. Visvalingham and Whyatt, 1993) have highlighted a need to evaluate and validate existing generalisation tools rather than developing new generalisation algorithms and systems. So far, standard GIS software applications do not fully support automatic generalisation of line features.

Automated generalisation is an important operation for producing smaller scale GIS database and digital maps from larger scale existing GIS database and digital map. This is done through using the aforementioned operations and a number of generalisation algorithms which have been built into the GIS software packages. This research focuses on the integration and utilisation of generalisation operators in order to generalise a road

network database from 1:250,000 national topographic data to produce smaller scale maps at 1:500,000 and 1:1,000,000.

2.3.1 Relevant Generalisation Algorithms

The most important task of the cartographer is to portray information appropriate to the scale and purpose of a map in a comprehensible, succinct and neat form. To do this, a number of ‘generalisation processes’ are typically in use in relation to simplification of linear spatial information that eliminates unwanted detail from the master database. A line in vector graphics is stored digitally as a series of representative points. Line simplifications select the subset of points that best preserves the general characteristics of the line at the scale at which it is being displayed. Each scale requires a level of simplification resulting in the data being depicted in neither too detailed nor too general a form, the smaller the map scale, the greater the level of simplification required.

2.3.1.1 Douglas–Peucker Algorithm

The Douglas–Peucker algorithm was developed by David Douglas and Thomas Peucker in 1973. The Douglas–Peucker (DP) remains one of the most popular and extensively used line simplification algorithms by both computer graphics and geographic information systems for automated spatial information. This algorithm keeps global optimisation; which is an excellent attribute. It uses the distance of a vertex from an edge as the criterion to establish whether a vertex is a characteristic one or not.

A review of literature reveals that there is an error in the calculation of the distance between baselines and intermediate data points when it was implemented by a number of commercial software vendors (Hershberger and Snoeyink, 1992). Thus, the majority of written DP code exhibits a frequent fault in operation of the DP algorithm. Ebisch (2002) presented a means to code the correct distance calculation. It should be appreciated that not every active implementation of the DP algorithm is in error, e.g. the code by Wessel and Smith (1996) contains an accurate method, and its function counts are only slightly higher than those of the code developed by Wessel and Robinson (1999; cited by Ebisch, 2002).

The DP algorithm iteratively selects new points for insertion in the thinned output polyline based on their departure from a baseline connecting two neighbouring points

already chosen for inclusion. Ebisch (2002) highlighted that this problem arises from the original chapter written by Douglas–Peucker in 1973 that is somewhat ambiguous in its definition of the distance criterion for selection of a point. Definitions from that paper are: (a) '*The perpendicular distance of C from the segment A–B…,' (b) '…distance from the straight line segment…*', (c) '*…the furthest point from the straight segment…', (d) '…maximum perpendicular distance from the segment…*', (e) *'…the greatest perpendicular distance between it and the straight line defined by the anchor and the floater'*. Of these definitions, (a) and (c) are mostly strictly correct, but (e) seems to be the most widely used by computer scientists. The distance from the segment is important, rather than the distance from the line or the perpendicular distance from the segment.

The start and end points of a line are connected by a straight line segment (**Figure 2.1**). While vertex reduction algorithms use the relative closeness of vertices as a rejection criterion, DP uses the closeness of a vertex to an edge segment. It works from the top down by starting with a crude initial guess at a simplified polyline, namely the single edge joining the first and last vertices of the polyline. Then the remaining vertices are tested for closeness to that edge. If there are vertices further away than a specified tolerance from the edge, then the vertex furthest from it is added to the simplification. Tolerance is a numerical value representing the acceptable error range a feature (e.g. road segment) will have from its actual point found on the ground when processing or editing a geographic feature's coordinates. The above process creates a new deduction for the simplified polyline. Using recursion, this process continues for each edge of the current presumption until all vertices of the original polyline are within the tolerance of the simplification (**Figure 2.2**).

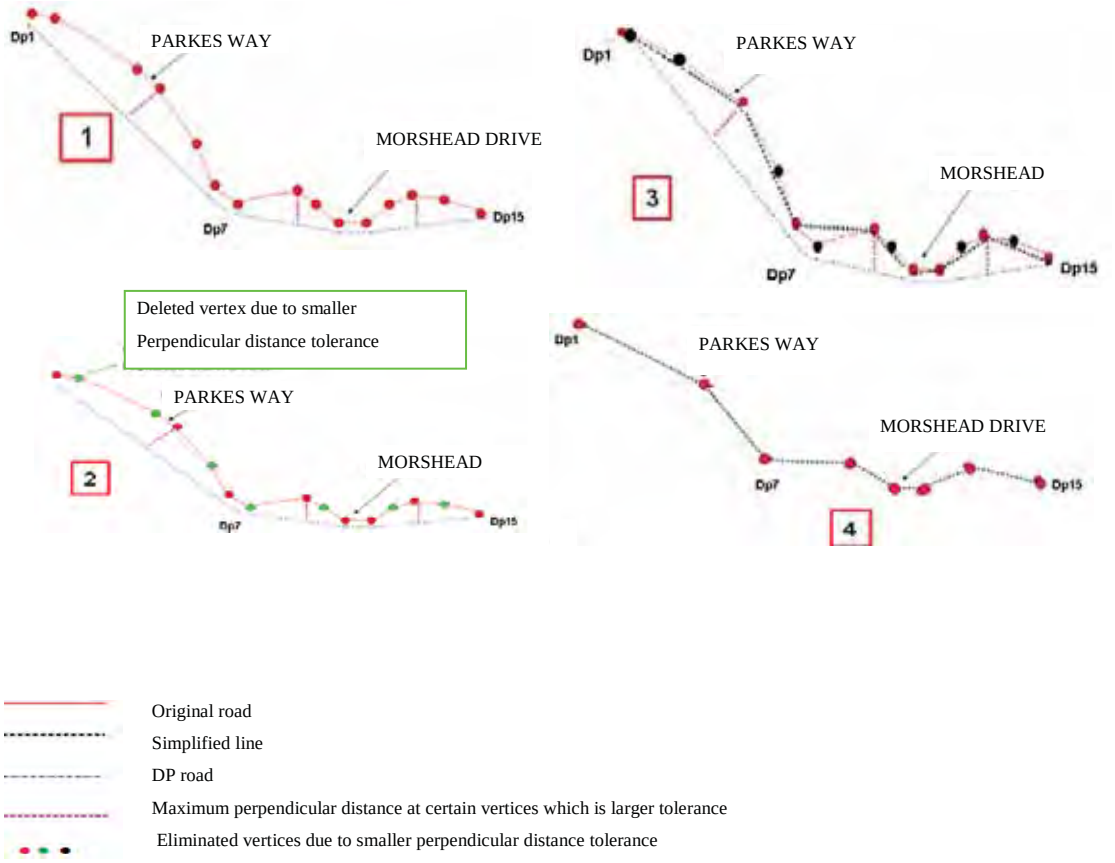


Figure 2.1 The Douglas–Peucker (DP) simplification algorithm overall over a segment of Morshead Drive, Canberra, Australia

In **Figure 2.1**, the red line is the original road segment, the black dotted line is simplified road using a conservative threshold level, the blue dotted line is DP simplified road, the magenta line is the maximum perpendicular distance for vertices with value greater than the specified tolerance, and the solid black circles are eliminated vertices due to smaller perpendicular distance tolerance.

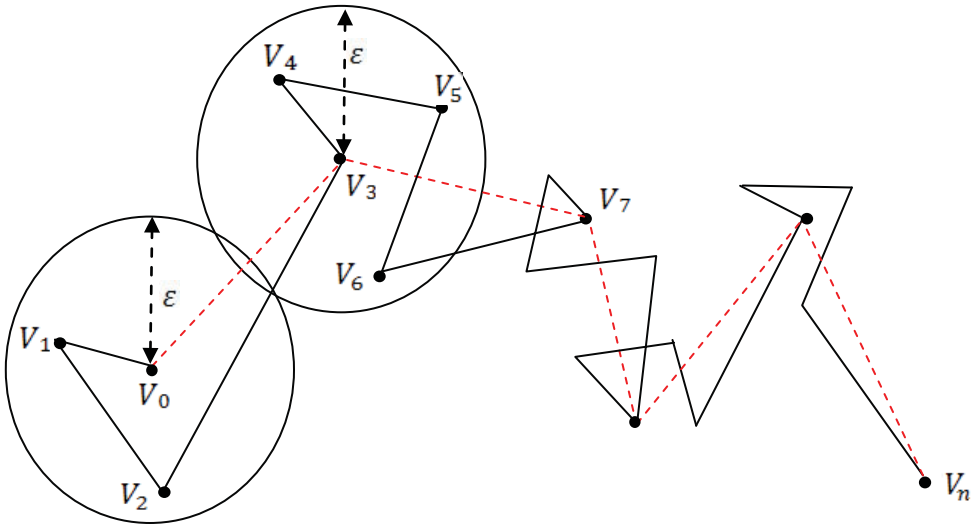


Figure 2.2 Vertex reduction of a polyline (Sunday, 2003). The solid line segment (•——•) represents the original polyline, the dotted line segment (•– – –•) represents the reduced polyline and this solid symbol (•) represents vertex removal.

The above figure **(Figure 2.2)** portrays a polygonal chain that consists of a series of vertices $v = \{v_0, v_1 ..., v_n\}$. Finding a subset of vertices to approximate the chain v can be handled with a line simplification algorithm (such as Douglas-Peucker recursive simplification). This can be done when assuming the input chain v has no self-intersections (Brown and Hershberger, 1994):

The algorithm for the recursive line simplification given in the following example starts with an array of points such as a polygonal chain $v = \{v_0, v_1 ..., v_n\}$. The recursive line simplification method functions as follows:

a) v is approximated by the line segment $\overleftrightarrow{v_0 v_n}$.

b) The farthest vertex v_f is from the line $v_0 v_n$. Its distance $d(v_f, v_0 v_n)$ is determined at a given tolerance $\varepsilon \geq 0$ in order to accept the segment $v_0 v_n$ as an appropriate approximation to v.

c) If its distance $d(v_f, v_0 v_n)$ is not determined at a given tolerance $\varepsilon \geq 0$ then, break v at v_f and recursively approximate the sub-segments:

$v = \{v_0 v_1 ..., v_f\}$ and $\{v_f ..., v_n\}$

d) Continue until all segments are either within the tolerance, or consist of one line.

The vertex reduction algorithm is a fast $O(n)$ algorithm and a less complicated one, but it gives a much cruder result. The outputs can be used as a pre-processing stage before applying other algorithms. This results in a faster combined algorithm since vertex reduction can significantly decrease the number of vertices that remain for input to other simplification algorithms. The DP algorithm contains two variants, including the original $O(n^2)$ method (Douglas and Peucker, 1973) and another published method such as $O(n \log n)$ (Vaughan and Brookes, 1989; Vaughan *et al.*, 1991; Hershberger and Snoeyink, 1992). The faster algorithm is more complicated to implement and only works for simple 2D planar polylines, and not in higher dimensions. Finally, these algorithms are merged with vertex reduction functions. These are then followed by DP approximation in the GES implementation, which uses a fast practical high-quality polyline simplification algorithm (**Chapter 7**). Four basic algorithms (Vertex Reduction, Classification, Merge and Enhanced Douglas-Peucker) and four generalisation operations are being used in relation to simplification and transformations of lines and polylines in an expert system in this research.

Regarding the vertex reduction algorithm, where successive vertices are clustered too closely they are then reduced to a single vertex, e.g. if a polyline is being drawn on a computer display, successive vertices may be drawn at the same pixel if they are closer than some fixed application tolerance. In a large range geographic map display, two vertices of a boundary polyline may be separated up to thousands of metres depending on the data set being used, and still be displayed at the same pixel; and the edge segments joining them are also being drawn at this pixel. To remove the redundant vertices successive vertices are detached. It is a brute-force algorithm for polyline simplification; a polyline vertex is discarded when its distance from a prior initial vertex is less than some minimum tolerance. In particular, after fixing an initial vertex v_0, successive vertices v_i are tested and rejected if they are less than ε away from v_0. But, when a vertex is found that is further away than ε, then it is accepted as part of the new simplified polyline, and it also becomes the new initial vertex for further simplification of the polyline. Thus the resulting edge segments between accepted vertices are larger than the ε tolerance.

Hershberger and Snoeyink's (1992) paper offers an Appendix with a complete implementation of DP that improves on the worst case behaviour, the original DP algorithm is usually $O(n \log m)$ and performs better in cases where the value of m is relatively small. The best algorithm should be easier to code, and generally the DP algorithm is recommended by this study and other authors. The implemented DP algorithm does the following in GES, as discussed in **Chapter 7**.

a) Find a vertex (*P*) which is farthest away from the two end points (called A and B, actually both are the same vertex).
b) Divide the polygon into two line segments, *A*-*P* and *P*-*B*.
c) Apply DP to both line segments.

There are still problems with polygon simplification, e.g. for the boundaries of states or countries as they share part of these boundaries. Simply applying the above algorithm will cause the shared part to be inconsistent. Further improvements would include:

a) Finding the first and the last vertices (*C*, *D*) of the shared part of two polygons (boundaries) and marking them as key points. Key points must be preserved when applying any line simplification/generalisation algorithms.
b) Apply DP to line segments A_1-C_1, C_1-D_1, D_1-B_1 for polygon 1 and A_2-C_2, C_2-D_2, D_2-B_2 for polygon 2.
c) When applying DP to line segment C_1-D_1 and C_2-D_2 (both are key points), ensure that the order of processing vertices is the same, e.g. from east to west and north to south, otherwise the shared part may be different after simplification.

Perpendicular offsets for all overriding points are then calculated from this segment, and the point with the furthermost offset is identified. If the offset of this point is less than some pre-defined tolerance, then the line segment is simplified. Otherwise the point is selected, and the line is subdivided at this point of maximum offset. The process is then recursively repeated for the two parts of the line until the tolerance criterion is fulfilled. Selected points are finally chained to produce a simplified line (Whyatt and Wade, 1988).

More specifically, the two extreme endpoints of a polyline are connected with a straight line as the initial rough approximation of the polyline. Then, how well it approximates the whole polyline is determined by computing the distances from all intermediate polyline vertices to that (finite) line segment. If all these distances are less than the specified tolerance ε, then the approximation is good, the endpoints are retained, and the other vertices are eliminated. However, if any of these distances exceeds the desired tolerance, then the approximation is not adequate. In this case the point is chosen that is furthest away as a new vertex subdividing the original polyline into two (shorter) polylines, as illustrated in **Figure 2.1**. This procedure is repeated recursively on those shorter polylines. If at any time all of the intermediate distances are less than the desired threshold, then all the intermediate points are eliminated. The routine continues until all possible points have been eliminated. Successive stages of this process are shown in **Figure 2.3**. The DP algorithm tackles line simplification through the specification of a single tolerance parameter so that a lower tolerance results in many points being selected, whereas a higher tolerance results in fewer points being selected.

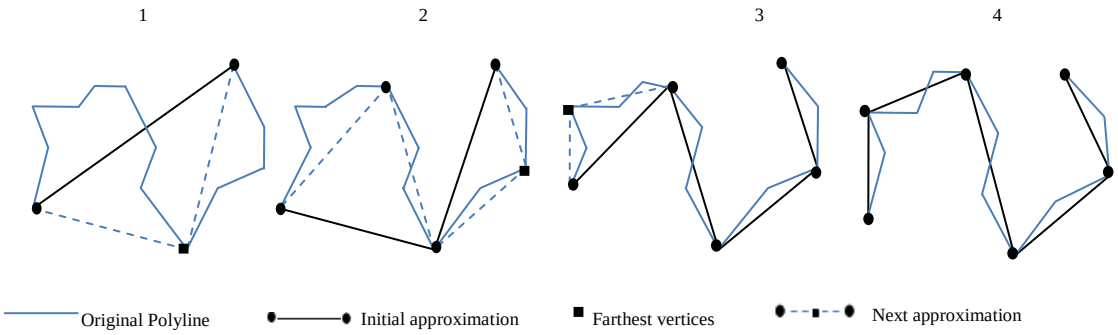


Figure 2.3 Successive stages of DP algorithm (Adapted from Sunday, 2003)

Despite the fact that the DP has been well regarded due to its performance in both perceptual and mathematical terms, it has also been criticised for the speed at which it operates. Vaughan *et al.*, (1991) and Hershberger and Snoeyink (1992) proposed a method of speeding up the operation of the DP algorithm using multitasking techniques. This was performed on a Sequent Symmetry computer in which three parallel implementations of the algorithm, and comparisons of the performance of each with the original sequential code were presented. The DP algorithm in its original form returns only the points that exceed a specified tolerance value derived in a non-recursive manner, written in FORTRAN language that does not support recursion. Vaughan and

Brookes (1989) modified the original algorithm to process all points on a line and store the suitable offset principles. Their tests were performed on a number of data sets of differing sizes and resulted in improvements in DP performance.

Detailed descriptions of the aforementioned algorithms and their applications can be found in Nakos (1997 and 1999); Barrault (1995); Skopeliti and Tsoulos, (2001) and published papers by the author.

2.3.1.2 Other Generalisation Algorithms

Kass *et al.,* (1987) initially introduced the Snakes model for contour detection in the field of image processing. Snakes '*…are lines defined with their own energy from their geometrical features. A snake is at an equilibrium position when its global energy is minimal. In order to minimise its energy and reach this position, the snake can deform itself …*' (Guilbert and Saux, 2008). The method consists of an internal energy and an external energy. The internal energy controls the shape of the snake and the external energy considers application constraints because it is not related to the geometry of the curve. Guilbert and Saux (2008) presented a B-spline snake model for simplification of maritime lines. A combination of generalisation operations (smoothing, displacement, aggregation and deletion) were used to correct visual conflicts in a set of lines by observing defined constraints. To manage local conflicts (e.g. intersections or self-intersections), the consistency of the lines was checked and discrete operations (e.g. segment removal) were performed during the generalisation of marine chart data.

Brophy (1973) smoothing algorithm involves user direct control for carrying out both feature elimination and systematic point elimination. It computes every point of the initial line by building a triangle with the current point (p) and the points $p+k$ and $p-k$ (k = look-ahead to build a triangle), inscribing each triangle with a circle, the current point is moved a specific distance towards the centre of the circle.

Lang's (1969) algorithm incorporates a Euclidean distance measure for point elimination applying tolerance values. It uses the coordinate pairs of a constructed line that connects the first coordinate pair to each successive coordinate pair. Every time the line is connected to a new coordinate pair, the perpendicular distance is computed from that line to all intermediary points.

The *N-th* Point simplification algorithm was introduced by Tobler (1966) and was one of the first sequential point elimination algorithms designed to select every *N-th* (determined by user) coordinate pair to be retained on the generalised line segment (Rhind 1973). According to de Koning (2010), the *N-th point* routine is a naive O(n) algorithm polyline simplification. It keeps only the *first*, *last*, and each *nth* point. All other points are removed. For example, if a polyline consists of 8 vertices: *{v1, v2 ... v8}*. This polyline can be simplified using n = 3. The resulting simplification consists of vertices: *{v1, v4, v7, v8}*. The algorithm is extremely fast, but has some significant drawbacks as it is not very good at preserving the geometric features of a line, as well as over representing the starting straight lines.

The Reumann-Witkam simplification algorithm uses two parallel lines to describe an area of interest (AOI) after calculating the original slope of the AOI, the line is processed successively until one of the edges of the search corridor intersects the line.

2.4 GEOGRAPHICAL FEATURES GENERALISATION

Technological development in the field of generalisation is moving very fast, following the trend from manual cartography to computer-based cartography. Map generalisation has become a desirable component of today's GIS. It is an integral part of spatial data collection, representation and access. Most generalisation algorithms developed and employed by the GIS and computer science communities have been tailored for map production. Typically these algorithms exhibit raster cell generalisation, vector line, polyline and area simplification.

2.4.1 Generalisation of Linear Features

The majority of features represented on maps are linear, such as rivers, boundary lines, coastlines and roads. Generalisation of linear features is known as one of the most important themes in the generalisation process (Skopeliti and Tsoulos, 2001). Therefore linear feature generalisation plays an important role in GIS (Barrault, 1995; Skopeliti and Tsoulos, 2001). Several algorithms have been developed to simplify these lines. There is, of course, no shortage of literature dealing with the generalisation algorithms. Numerous authors (e.g. McMaster, 1989; McKeown *et al.*, 1999; Oosterom, 1995;

Visvalingam and Williamson, 1995; Thomson and Brooks, 2002) have surveyed and applied simplification algorithms and operations.

McMaster (1989) classified the processing of linear features into five major algorithmic categories: (a) independent point algorithms of map generalisation where a mathematical relationship between neighbouring pairs of points is not established; (b) local processing routines that apply the characteristics of immediate neighbouring points to determine selection; (c) constrained extended local processing routines that apply distance, angle, or number of points to search beyond neighbour points; (d) unconstrained extended local processing routines that use morphologic complexity of the line to search beyond neighbour points; and (e) global routines that take into account the entire line or specified segment. However, none of these methods has led to an automated generalisation mechanism.

McKeown *et al.,* (1999) improved the line simplification operation of the Douglas and Peucker algorithm to prevent the most common topological and terrain related anomalies (artefacts). This resulted in reduction of the amount of manual intervention required by incorporating topology checks for self-intersection, and terrain checks to insure that the relationship of the object and related terrain were preserved in a virtual world production environment; while these checks can be applied in other application environments.

Oosterom (1995), however, criticised these types of algorithms as time consuming, hence he introduced the reactive-tree data structure for line simplification that is applicable to seamless and scale-less geographic databases. There is still, however, a need for the cartographer's input in generalising lines/curves to make them fit for use. Skopeliti and Tsoulos (2001) developed a methodology for the parametric description of line shapes and the segmentation of lines into homogeneous segments along with measures for the quantification of shape change due to generalisation. They stated that measures that describe positional accuracy are computed for manually generalised data or cartographically acceptable generalisation results.

Muller *et al.*, (1995) imply that the ongoing research into line generalisation is not being managed properly. Most of the research in generalisation has focused on single cartographic line generalisation instead of working on data modelling in an object-oriented environment to satisfy database generalisation requirements. In contrast, other researchers (e.g. Visvalingham and Whyatt, 1993) have highlighted a need to evaluate and validate existing generalisation tools rather than developing new generalisation algorithms and systems. Current standard GIS software applications support the generalisation of line features which is the focus of this research for developing a generalisation framework to derive multiscale spatial data.

Skopeliti and Tsoulos (2001) also stated '*…generalisation of linear features is considered to be among the most important generalisation operation. This is due to the fact that a majority of map features are represented as lines or polygons, which are bounded by lines*.' Linear feature generalisation constitutes an important research area for both the cartographic and model generalisation fields, since generalisation of these features to different scales by keeping their shape and size relative is an important task. A cartographer often generalises a road network based on geometric, topological and thematic properties such as road classes and road symbols (Thomson and Brooks, 2002). The relative importance of road segments can be inferred by thematic information. It should be noted that undertaking this exercise over a large database of a state or country is not cost effective. Therefore an automatic/semi-automatic method is called for (Paluszynski and Iwaniak, 2001).

Similarly, Skopeliti and Tsoulos (2001) developed a knowledge-based approach for the generalisation of linear features with special emphasis on coastlines through the use of assessment tools, namely; global routines (Douglas Peucker), local processing routines, routines taking into account structure (ESRI bend simplify by Wang and Muller, 1998). In their study four conditions were established, including very smooth, smooth, sinuous and very sinuous. Skopeliti and Tsoulos's assessment was based on human interpretation but failed to quantify the assessment. This certainly impacts on quality of generalisation by affecting areas such as positional accuracy (displacement), attribute accuracy (classification and aggregation) and completeness.

Lonergan and Jones (2001) described a simple measure of map quality and iterative strategy for object displacement. For measuring map quality it was considered that features must be separated from each other by at least a minimum distance in the final map, and each one can be displaced by up to a given maximum displacement from its original reference position. This quality measure considers legibility because of the separation distances and accuracy of locations.

Sester (2000) discussed a map generalisation solution using least squares adjustment theory (LSA) which is a well known general framework used to identify unknown factors based on given observations. Its use is particularly well documented in mathematical science and geology. This technique is applied to the simplification and displacement of buildings and streets that were applied over a topographic dataset at 1:25,000 scale. The results of simplification and displacement generally provide a much clearer view. Furthermore, this research linked all objects with each other in order to displace neighbouring objects. Ibid (2000) claims that the above method offers a very promising solution because it incorporates local and global object recognition algorithms. In later research (Sester, 2005); she presented the idea of integrating different generalisation approaches that include interpretation methods to detect and extract aerial objects (e.g. buildings) or linear features (e.g. roads) with common parameters. Also it calls for the availability of suitable control parameters and evaluation measures of the operations to define what operation should be executed with the relevant parameter.

2.4.2 Road Network Generalisation

In the last decade there has been a rapid increase in the development and application of semi-automatic techniques to map and database generalisation. Computer scientists, mapping specialists and cartographers are involved, with increased integration of human knowledge with generalisation operators (algorithms) for automation of generalisation (e.g. Kreveld and Peschier, 1998; Iwaniak and Paluszynski, 2001). Several generalisation processes exist however the key operators include feature selection, feature simplification, feature classification and feature symbolisation. The process of generalisation of linear features often involves the use of geometric operations: selection, elimination, merging, displacement, aggregation and symbolisation. Detailed

descriptions of these operations can be found in McMaster and Shea (1992), Muller *et al.*, (1995), Harrie (2001), and Kreveld (2001).

Most generalisation algorithms are applied to lines without taking into consideration that a linear feature may be a part of an area feature. Muller and Zeshen (1992) proposed an automated approach for generalisation of area features based on data display objectives for different scales. This technique comprises data pre-processing, area expansion and contraction, elimination, reselection, aggregation, displacement, topological integrity checks, smoothing and reduction. A description of some of the more important polygon/area generalisation algorithms is provided in Burrough (1986), Ruas (1995) and Hohl (1998).

A focus on road network generalisation from 1:250,000 national topographic data to 1:500,000 and 1:1,000,000 with some recent examples relevant to this research are provided in this section. Kreveld and Peschier (1998) developed a new approach to decide which roads should be selected in generalising road network maps. The approach considered not allowing roads to be too close to each other, avoiding detours between important points, and giving priority to larger roads. In addition, Iwaniak and Paluszynski (2001) presented a methodology to generalise topographic TOPO 1:10,000 roads of urban areas into 1:50,000 scale utilising the MGE Map Generaliser™ software in batch mode to perform map transformations and a rule-based system for controlling the generalisation process.

Similarly, Harrie (2001) applied a graphic generalisation process that used the generalisation operators (simplification, smoothing, exaggeration and displacement) on the scale of 1:10,000 to derive 1:50,000 scale map of roads and buildings using the LAMPS2 software. The method focused on readability and clarity of maps while preserving the characteristics of data that are of use in real time navigation systems (e.g. transportation, mobile mapping). Also, Bakker (1997) developed a framework for the re-engineering of the Dutch topographic map production at the scale of 1:10,000 scale databases to derive 1:25,000 scale maps. This provides a more structured database for GIS applications in The Netherlands. Furthermore, Bengtson (2001) implemented a new automated generalisation approach for topographic maps of Denmark at a scale of

1:10,000 to generate 1:50,000 map products within the LAMPS2 environment. The process developed for generalisation of roads supports the collapse and re-establishment of the connections between roads associated with the original centreline and the derived centreline.

Moreover, Geoscience Australia (GA) (2001) generated the *Global Map* at the 1:1,000,000 scale spatial data product across Australia by generalisation of the 1:250,000 national topographic data. The notion of the *Global Map* was initiated in response to the United Nations' Agenda 21 to create a 1:1,000,000 map of the world in digital on-line format to respond to the needs of improving environment related decision-making on the global scale (Taylor, 2007). Point features were automatically selected by attributes that were included in 1:1,000,000 scale data and appropriate reference databases and settlements were also added to the roads database. It involved manual editing and checking of the automation results including correction of attribute's errors and removal of extraneous point features. They utilised a number of generalisation operations for deriving the *Global Map* layers including; selection by attribute, selection by location, elimination based upon a minimum length criterion, and line smoothing. It has been stated that '*…line generalisation proved to be one of the most complicated and difficult processes to automate*' (GA, 2001).

Thomson and Brooks (2002) employed perceptual grouping principles for generalisation of road and river networks to automatically produce the *National Atlas of Canada*. Their findings indicated that this technique is potentially applicable to other network types such as powerlines and pipelines. In addition, National Land Survey (NLS) of Finland has focused on generalisation techniques used in 1:100,000 map database production. NLS Finland produces data for five different small scale map databases: 1:100,000, 1:250,000, 1:500,000, 1:1 million and 1:4.5 million. The million scale databases were collected mainly by manual editing methods in which 1:250,000 and 1:500,000 map databases were produced by automatic generalisation of larger scale map elements, manually digitising small scale map elements and matching them together. NLS Finland has developed some new automatic methods for the 1:100,000 map generalisation and for updating within the Arc/Info environment. The production line is totally based on the Arc/Info environment, where interfaces are built to start

different generalisation processes (Kilpelainen, 1997). NLS has achieved good results in automatic generalising; however significant manual work was still required.

Perceptual grouping, considered an important aspect of understanding maps and their relative importance in map generalisation, emerged from Gestalt laws. This technique has been used for analysing road network elements (Thomson and Richardson, 1999) and drainage networks (Thomson and Brooks, 2002). This method also reflects good continuation of linear elements as perceived from the perspective of human cognition (Elias, 2002). In relation to this Topfer (1996) identified the link between network attenuation and the required map scale using the principle of selection theory.

The ATKIS data model (Elias, 2002) incorporated good continuation of the grouping principle relying on geometric characteristics of roads based on relative importance to simplify the whole network. However, a deficiency of this approach is that long roads close to each other retained and formed many parallel structures. It is notable that Edwards and Mackaness (2000) tackled road network simplification by analysing and classifying the density distribution, and then integrating the results with the simplification algorithm. This was achieved by a scale dependent thresholding process.

Additional criteria can be used for road network attributes (e.g. road classification, names) and river networks (flow direction). Principles such as proximity, similarity, symmetry, closure, co-linearity, co-termination, continuity and familiarity that were originally introduced by Gestalt laws to computer vision can be used. This contextual information has been also used in identification and classification of road networks from remote sensing imagery (Gerke and Heipke, 2008). Good continuation is combined with the Radical Law of feature selection in order to determine how many features to remove in line with reduction in the map scale (Richardson and Muller, 1991). Utilising thematic attributes makes this more effective.

Comparative evaluation of the above technique, employed via GenSystem™ software developed by the Canada Centre for Remote Sensing (CCRS), versus cartographic generalisation shows that analysing a road network into linear elements (e.g. strokes) and evaluating their relative importance is easily performed to assist map derivation.

Removal of the strokes is based on the relative importance of each road; this can be an effective method for generalisation of a road network to the desired scale level (Elias, 2002). For example, Thomson and Brooks (2002) developed three simple principles for a road network generalisation using the concept of perceptual grouping: (1) a longer stroke is more important than a shorter one; (2) a stroke comprising an important road class is more desirable compared to less important road classes such as tracks versus principal roads; and (3) no connected portions of the original network should be split in two disconnected segments. These rules were applied for generalisation of a road network in Ottawa, Canada. However, while the generalised road network seems appropriate visually, there is a lack of quantification of the generalisation results.

2.4.3 Raster Generalisation

Generalisation of GIS data is one of the most challenging tasks for cartographers. It is particularly difficult to automatically generalise thematic raster maps derived from remotely sensed data. Over the past two decades many generalisation techniques have been developed. Generalisation of vector data and a generalisation framework for road networks was discussed previously (Kazemi and Lim, 2005). On the other hand, generalisation of raster data such as satellite imagery has been studied by, for example, Petit & Lambin (2001), Daley *et al.,* (1997), and He *et al.,* (2002).

Kazemi *et al.*, (2009a) also applied three generalisation techniques (supervised classification, thematic generalisation and spatial aggregation) in order to build a raster generalisation framework known as the Interactive Automated Segmentation and Raster Generalisation Framework (IASRGF) for segmentation and generalisation of satellite imagery. Test results of the IASGRF show that all objects derived from the generalisation of land use data over Canberra, Australia, were well classified and mapped. The error assessment indicates that the percentile classification accuracy is 85.5%, whereas the commission error is relatively high (38.5%). More importantly, the maximum likelihood classifier using training sites and associated ground truth data suggests that the Kappa index is 0.798, which can be interpreted as a reliable and satisfactory classification result. In order to further enhance supervised classification, a post-classification was carried out. As a result, this extra process improved the overall classification accuracy slightly, however its commission error also increased by 6%.

Raster generalisation algorithms (e.g. aggregate, boundary clean, expand, majority filter, region group, shrink, thin) embedded in a typical GIS software package can be applied to either clean up small erroneous cells/pixels such as unclassified data derived from remotely sensed imagery, or for the generalisation of raster data obtained from a scanned paper map in order to remove/smooth out unnecessary details including lines and texts or data imported from some other raster format (ESRI, 1992). The majority of existing software packages lack workflow, procedures or straightforward practical guidelines (protocols). If a cartographer's expertise and knowledge are applied to the software, many raster generalisation problems could be solved.

Daley *et al.,* (1997) compared a raster method (MapGen) and an object-oriented method (ObjectGen) to automatically generalise forests from multiple image datasets ranging from 1m to 1km spatial resolutions by segmentation of remotely sensed images (MEIS 1m, AVIRIS 20m, TM 30m, and AVHRR 1km), at corresponding resolutions to the GIS files to constrain the generalisations. MapGen is an automated raster generalisation system that is based on a set of polygon and vector generalisation rules. Each polygon rule specifies how to merge neighbouring polygons if their size is smaller than a specified minimum tolerance. In this system, feature attributes are stored in a database for fast sorting and selection. The GIS dataset used was composed of topographic data and forest cover maps, both at the 1:20,000 scale. Generalisation was carried out on three forest cover maps to create broad classes for deriving data with a map scale ranging from 1:20,000 to 1:250,000. It was concluded that there were no significant differences in class areas between the two generalisation methods and the original areas. An approximate reduction of 72% of the original input data was achieved without significant errors in class areas. However, the authors used ObjectGen and MapGen for purely research purposes and did not demonstrate the operational use of their methodology.

Similarly, Cihlar *et al.,* (1998) developed a Classification by Progressive Generalisation (CPG) procedure using fused AVHRR and Landsat-5 (30m resolution) data. It was demonstrated that CPG gave superior accuracy to other existing classification methods. They also demonstrated that CPG is user-friendly and has potential applications for other merged imagery datasets. Jaakkola (1998) also presented a rule-based

generalisation methodology for generating land cover maps from raster data. It was shown that it is feasible to automatically derive small scale land cover maps from large scale data using raster generalisation techniques and map algebra. Forghani *et al.*, (1997) applied various combinations of morphological operations (e.g. dilation, erosion, skeletonisation) to extract and generalise roads from aerial photography. However one shortcoming of this approach was that it is computationally expensive. Also, significant testing is required to determine what is the optimal threshold when applying different morphological operations.

Furthermore, Gjertsen (1999; cited in AGENT, 1999) at the Norwegian Institute of Land Inventory developed a workflow for the generalisation of a Norwegian national land type database that relies on the use of two generalisation processes: attribute-based generalisation (e.g. reclassification) and geometry-based generalisation (e.g. line elimination, class integration, area aggregation and area elimination). A total of 13 classes were used in the generalised classification system, and all area features were reclassified based on their attributes. It seems that this approach has the potential for operational use.

In addition, Walter (2004) applied an object-oriented classification of multispectral remote sensing data using a supervised maximum likelihood classification. A GIS database was used to derive training areas to update topographic maps at the 1:25,000 scale. However, it was not clearly explained how generalisation was used to update the topographic database. In another study Wenxiu *et al.,* (2004) developed a knowledge-based generalisation of land use maps for scales 1:10,000 to 1:50,000 using Arc/Info's™ generalisation tools. Two generalisation knowledge sources were used, including general knowledge (e.g. geometric and topological, GIS analyst's knowledge and experience on generalisation operations and GIS data management), and thematic knowledge (e.g. terrain knowledge, application-based knowledge). A series of rules, including rules of feature selection, attribute transformation, and rules for merging features, were employed. Although this research focused on vector generalisation it provides some constructive ideas on the integration of expert human knowledge.

He *et al.,* (2002) investigated effects of rule-based spatial aggregation index and factual dimension techniques on classified Landsat-5 imagery in an attempt to compare the effects of random rule-based aggregations when examining the distortions introduced by data aggregation with regard to cover type quantities and landscape patterns. The findings indicate that these spatial aggregation methods over a broad range of spatial scales (30-990m) lead to different outcomes in terms of cover type proportions and spatial patterns. The rule-based aggregations resulted in distortions of cover type percentage areas reported in other studies (e.g. Moody & Woodcock, 1995). In contrast, using random rule-based aggregations did not distort the results significantly. This is superior to the majority rule-based aggregations; hence this technique is a promising tool for scaling data of fine resolution to coarse resolution while retaining cover type proportions.

Notwithstanding the extensive research that has been undertaken, there are still intractable problems regarding these approaches which often make them impractical in an operational environment. Fully automated generalisation of raster and vector data has still a long way to go. Meanwhile, to meet current map production requirements for generalisation of raster data, a common approach is to classify imagery with the application of a trained image analyst/cartographer's knowledge, to reclassify and recode that classified data, and to then apply statistical and spatial filtering methods.

2.5 OBJECT-ORIENTED MODELLING AND GENERALISATION

Object-oriented technology is embedded in most of the current GIS software such as ArcGIS™ and Laser-Scan™. However, the current data models and structures are not sufficiently calibrated (matured) to support comprehensive derivative mapping (Sester, 2000; Mustiere and Moulin, 2002). The NMAs are moving towards building and maintaining an infrastructure database. Keeping multiple databases is no longer generally acceptable (Brooks, 2001).

Since an effective data model stores special relationships among features, by designing a good data model relationships such as adjacency and connectivity can be established, so that generalisation operations (e.g. aggregation) will be more effectively defined through topological relationships between features. In this regard ESRI geodatabase

technology provides a framework for objects to maintain geometry attribute, spatial reference, relationships, domain and validation rules, topology and custom behaviour (Zeiler, 1999). For instance at a very large scale map roads appear with detailed multiple lanes, and at medium range scale lanes are formed as two-way traffic direction, and at small scales appear as a single road.

Furthermore, to maximise the effectiveness of the generalisation process, Yang and Gold (2004) described a system approach to automated map generalisation by combining database generalisation and dynamic object generalisation capabilities in the system, and to couple a map agent on top of a map object that constructs transportation maps. Other applications of object-oriented spatial database include wireless communication and real time generalisation, and on demand map generalisation (e.g. Sarjakoski *et al.*, 2002). Also the database must offer continuous zooming, enrichment, and capability to perform smooth generalisation over the web.

In this regard multi-resolution spatial databases provide the ability to represent objects in multiple representations tailored towards the requirements of different users, especially for web applications. A multi-resolution database should preserve spatial relations throughout scale changes (Tryfona and Egenhofer, 1996). Generally there is a linear direct relationship between scale changes and the amount of generalisation (Kerveld, 2001). Continuous scale change of maps is already operational on modern computers and so technological developments will soon provide this capability for web cartography (Karaak and Brown, 2001, cited by Kerveld, 2001). It should be noted that consistent representation of networks such as roads need to be considered through two major criteria, including: (a) when small changes take place from one level to the next, and (b) when long changes accrue (Tryfona and Egenhofer, 1996). Examples of such changes have been presented by Kazemi and Lim (2007b). **Figure 2.4** presents a conceptual model for an object oriented, multiscale and multipurpose master database that enables derivative mapping.

Current data models and structures may not be sufficiently calibrated (matured) to support comprehensive derivative mapping (Mustiere and Moulin, 2002). Significant research has been directed towards the development of spatial data models using object-

oriented technologies (e.g. Wright and Goodchild; 1997; Li, 2000). Wright (2000) pointed out the disadvantages of storing spatial data in ArcInfo coverage format. For example, in GIS coverages features are aggregated into homogenous collections of points, lines, and polygons with generic 1-and 2-dimensional behaviour. There is no way to distinguish between behaviours within a set of feature classes. For example, the behaviour of a line representing a road is identical to the behaviour of a line representing a dynamic shoreline.

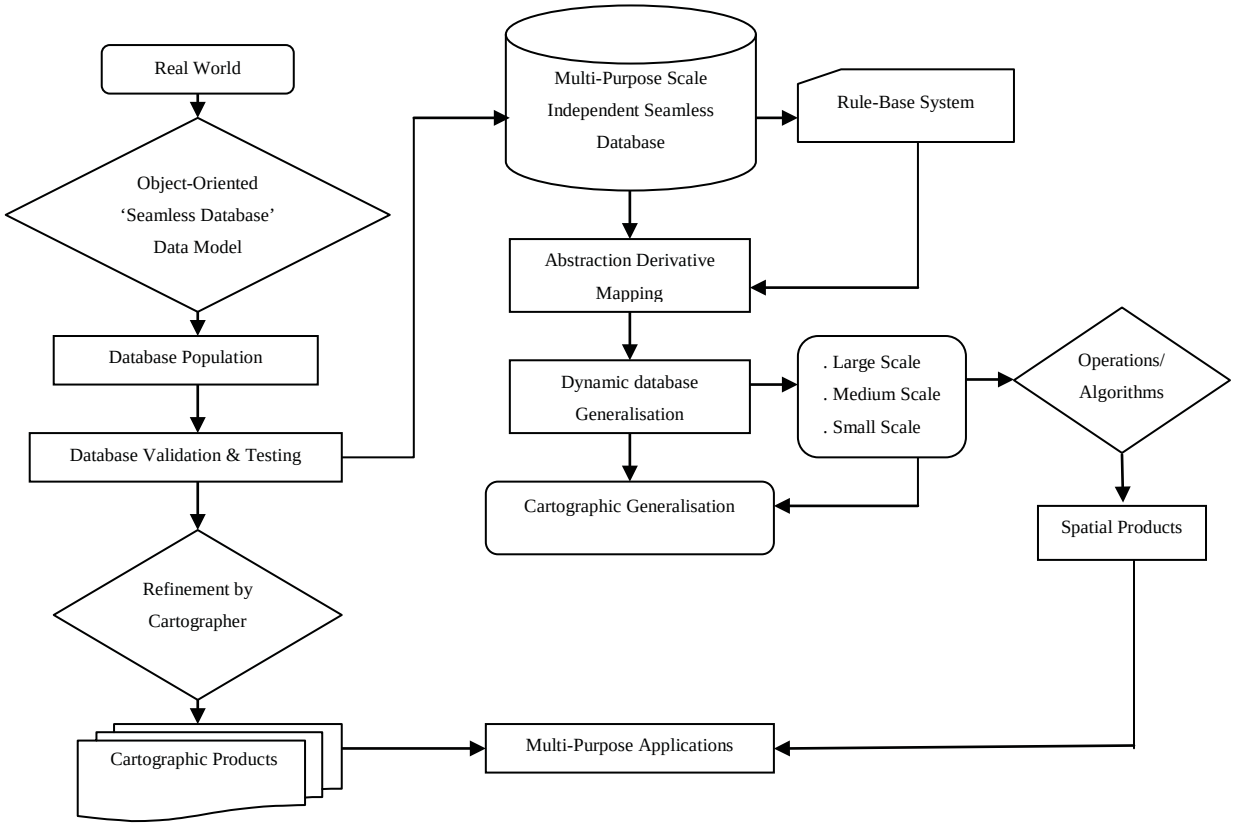


Figure 2.4 A conceptual model for an object oriented, multiscale and multipurpose master database that enables derivative mapping

The object-oriented data models handle feature behaviours for individual objects within the same categories. They can be modelled and their required relationships are defined appropriately (Zeiler, 1999 and 2001). Storing and managing spatial data in an object database allows the use of the most advanced manipulation and analysis capabilities within this environment. In particular it allows the ability to capture the behaviour of spatial objects and supports more complex rules that can be built into a geo-database.

Laurini and Thompson (1992) classified the data model design into four stages, increasing in abstraction from human-orientation to implementation in the computer:

a) External design - the real world is simplified according to application requirements, because information from the real world is too complex to be wholly included in a database. The first step in the data modelling process is to define the overall scope and content of the model. From an external design standpoint this involves the challenging task of identifying the common, essential objects that are modelled in most GIS projects within an application domain.
b) Conceptual design - this stage focuses on where a model is populated with defined spatial objects. This involves the creation of an analysis diagram, with the identification of major thematic groups and an initial set of object classes within these groups. In technical terms, starting with the core object classes and a set of named, real world objects to be modelled, an analysis diagram can be created. The creation of the conceptual data model often begins with a top-down approach, where the list of objects is conceptually divided into thematic layers.
c) Logical design - the entity-relation diagrams are converted to some kind of schema, such as tables.
d) Internal or physical design - this specifies functions of actual hardware and software for implementation of the model.

In relation to conceptual design for an object-based model when a basic grouping of objects is established, one can begin to identify more specific similarities between objects. In this process, new classes are identified and some classes are merged. The final result is a set of object classes, and an initial description of their attributes and key relationships defined in an analysis diagram such as Universal Modelling Language (UML). The objects and initial data model is built in UML and a schema is generated. The UML diagrams and schema are at the 'logical design' stage in the data modelling process using existing CASE (computer aided software engineering) tools to generate the required schemas.

In terms of types of classes, an abstract feature or object is a class that stores common attributes shared by the classes inheriting from it. It does not have objects of its own. An object class is a class represented as a table, in which each object has an ObjectID and Attributes, but no spatial coordinates. A feature class is an object class that has spatial coordinates. In terms of relationships, inheritance means that the classes below inherit the properties and methods of the classes above them within the class hierarchy.

There are advantages to moving from a coverage-based data model to an object relational data model. Prime examples and benefits are: modelling the behaviour of objects and the relationships between them in an object relational data model, allowing spatial data to be presented on the Internet, and facilitating efficient management of the database. The testing of the data model is an integral part of the modelling process. The development of the data model is an iterative process:

a) Features are broadly grouped thematically with a number of major feature datasets.
b) A balanced use of sub-types *vs.* feature classes should be developed.
c) Inclusion of desired data that is not currently held in an existing product.

2.6 CONVENTIONAL GENERALISATION SYSTEMS

Despite considerable research and development efforts directed toward the automation of cartographic generalisation by researchers and the GIS industry, existing software tools are not able to play a more significant role than graphic editing and statistical calculation (Meng, 1997). This is due to inadequate 'intelligence' (compared to cartographers) in determining ‘how’ and ‘when’ to generalise (McKeown *et al.,* 1999; Iwaniak and Paluszynski, 2001). To remedy this shortcoming, rule-based systems were introduced to incorporate topological, geographical and cartographical expert knowledge in order to build map generalisation expert systems. Examples of such expert systems (e.g. for generalisation of roads) are given in Kreveld and Peschier (1998) and Skopeliti and Tsoulos (2001). This implies a lack of fully automated generalisation tools. A number of commercial GIS vendors (e.g. Intergraph, ESRI and Laser-Scan™) have worked with various mapping agencies to use these generalisation tools for the production of maps at various scales, (e.g. Kilpelainen, 1997; Meng, 1997) while developing tools to automate generalisation. There are three major generalisation

systems that have been widely used in many countries. Relevant studies on the application of ArcGIS™, Intergraph DynaGen™ and Laser-Scan Clarity™ are highlighted below.

Generalisation modules have been used in a number of national mapping projects, for example: the *Global Map* at 1:1,000,000 spatial data product across Australia by generalisation of the Geoscience Australia's 1:250,000 national topographic data product (GA, 2000); production of five different small scale map databases: 1:100,000, 1:250,000, 1:500,000, 1:1 million and 1:4.5 million scale from the Finnish National Land Survey's 1:100,000 database; and production of 1:100,000 and 1:50,000 scale maps of the General Command of Mapping in Turkey. Recent mapping and cartographic products offer spatial frameworks to support GIS and mapping needs. Developing generalisation tools within a geoprocessing framework has opened opportunities to explore new technology and data models, and to make enhancements using better techniques (Lee, 2003). In principle, the generalisation systems embedded the Douglas and Peucker algorithm for line generalisation. However research shows that the Generalise tool does not provide a total solution for generalisation (Limeng and Lixin, 2001), because after the point, line and the feature are simplified, manual editing was still required. The reason is that topological errors are produced when applying the Generalise tools, such as line-crossing and line overlapping, and for polygon coverage errors such as no label or multiple labels were introduced. To deal with these problems manual editing is therefore necessary (Kazemi and Lim, 2007a). Detailed generalisation capabilities of this product are described by Lee (2002 and 2003).

Some examples and applications include generalisation of topographic 1:10,000 roads of urban areas into 1:50,000 scale (Iwaniak and Paluszynski, 2001), and generalisation of 1:100,000 scale maps from 1:20,000 of Quebec's reference database (Carignam and Dumoulin, 2002). As DynaGen™ performs model and cartographic generalisation tasks it can easily handle scale variations from double size up to five times the source scale in support of multiscale feature representations within a single database.

The Intergraph Corporation developed the MGE DynaMap™ Generaliser. The Intergraph generalisation module (DynaMap now known as DynaGen) has been tested

for various generalisation tasks in several countries, including the USA, United Kingdom, Canada, Germany, Spain, Sweden, Turkey, The Netherlands and China, but not Australia. In Spain, for example, DynaMap Generaliser was used to derive a topographic map at 1:100,000 from 1:50,000 scale data, and to produce an atlas composed of different maps at different scales (Baelia *et al.,* 1995). Therefore there is an opportunity for this research to test and apply this generalisation system. DynaMap deals with small scale derivation from large scale databases, theoretically without limitation of scale range.

A number of visualisation tools in DynaMap Generaliser are also available to assist the interactive generalisation processes (Lee, 1993). Iwaniak and Paluszynski (2001) combined the expertise of a cartographer with DynaMap Generaliser in batch mode to perform the actual map transformations, and to develop a rule-based system for controlling the process. They noted that this system does not have a mechanism for controlling topology. For making essential decisions in DynaMap Generaliser (such as tuning the generalisation sequence), it is necessary that system users select parameters for each algorithm and the number of iterations for each particular task.

Since 1990 Laser-Scan has been developing an Open Systems object-oriented Application Development Environment (ADE) named 'GOTHIC'. GOTHIC is an object-orientated spatial database (data model) that implements complicated topological structuring for use in quality assurance (QA) of data irregularities (e.g. overlaps and voids), and enables spatial search (e.g. adjacency). Operations are performed in two phases: compilation (database creation and maintenance from a range of sources), and product generation (extraction, symbolisation and generalisation).

Laser-Scan's generalisation module matured through the sponsorship of several mapping agencies and it has been operationally used by the South Africa Department of Survey of Land Information and the French National Mapping Agency for deriving the 1:100,000 database from the 1:50,000 database; and by the Ordnance Survey (OS) of Britain for generating 1:50,000 landranger maps from the OS MasterMap database. Other examples of the utilisation of Laser-Scan software include: production of 1:50,000 scale map of roads and buildings from 1:10,000 scale data (Harrie, 2001); and

automated generalisation for topographic maps of Denmark at a scale of 1:10,000 to generate 1:50,000 map products (Bengtson, 2001). It should be noted that recently a number of mapping agencies, including Institute Geographique National in Belgium, Kort and Matrikelstyrelsen in Denmark, Institute Geographique National in France as well as the OS, sponsored the MAGNET project in association with Laser-Scan to consolidate and extend current generalisation capabilities of Laser-Scan's recent Clarity™ product (www.laser-scan.com).

Clarity Laser-Scan™ software is created as an AGENT (Automatic Generalisation New Technology) cartographic suite. It consists of two systems at different levels of abstraction (micro, meso and macro). Micro is in charge of recognising the objects in risk of conflicts, and meso controls the topology and the relationship of objects with other map features. Some generalisation constraints (e.g. minimal size, and form preservation) happen at local level 'micro', whereas other operations such as displacement or selection are considered as contextual parameters that need to be applied to a set of geographical objects at the 'meso level'. The overall homogeneity of the map presentation refers to the macro level. A framework is required to model map and database design at various levels (Lamy, 1999).

Clarity software is based in Java script for the algorithm and constraints (set of implementation rules) using XML for loading and defining the parameters. XML offers the capability for displaying and manipulating large databases. In addition each project requires a variety of settings and perhaps a set of algorithms. Clarity provides this opportunity in that users can develop new algorithms in Java tailored for their project.

The Clarity application has been employed in the production workflow of a number of European National Mapping Agencies (e.g. the French Institut Geographique National, the Belgian Institut Géographique National, the Danish Kort and MatrikelStyrelsen, and the OS of the UK). This technology was then exploited to generalise buildings from a 1:10,000 database (TOP10DK) to 1:50,000 scale map at the Kort and MatrikelStyrelsen; road generalisation from a 1:50,000 database (BDCarto®) to produce the 1:100,000 scale maps series at the Institute Géographique National (Lecordix *et al.,* 2005); and by OS for generating 1:50,000 landranger maps from the

OS MasterMap database (Haire, 2001). This provides a cost-effective solution for these agencies' generalisation needs. The Clarity tool allows flexibility and customisation by interfacing Java to a user's setup parameters for the generalisation process. In addition it allows parameters to be loaded from XML files and has the capacity to integrate new cartographic knowledge, and has better performance and interoperability.

This enhancement of Clarity permits the expert to perform new research or to adapt the system to specific generalisation specifications. The majority of NMAs need to produce 1:50,000 or 1:100,000 scale maps from high resolution geographic databases such as the cadastral database. The tools in the Clarity production system can be grouped into three classes:

a) measures to detect conflicts of generalisation, e.g. bend coalescence, overlapping symbols and oblique junctions;
b) generalisation algorithms for simplifying forms, caricaturing bends, i.e. exaggerating or deleting, moving features, maintaining coherence between themes when boundaries are displaced with the roads during generalisation; and
c) conflict resolution strategies: object-oriented techniques and flexibility graphs.

The generalisation results of Clarity are topologically coherent in the resultant database while taking the cartographic rules into account, and outputs are transferable into various vector formats, e.g. shapefile (Haire, 2001) for web delivery to users.

Current generalisation algorithms function in isolation to one another, and require a highly interactive environment with experienced cartographers. There is a need to validate the methodological research for the fully automated design of multiscale thematic representations to create a professional tool such as a vehicle navigation application. The quality, flexibility and usability of geographic databases needs to be adequate for online applications. Methods are being developed for delivering geographic databases at different scales while offering quality cartographic outputs on the web.

Cybercartography is changing the nature of cartography. It involves establishment, production, administration, analysis and communication of spatially-referenced data on a wide range of applications applicable to societies/communities in an interactive,

dynamic, multimedia, multisensory and multidisciplinary mode (Taylor, 2003). Oosterom (1995) defined dynamic generalisation as those generalisation processes that are temporarily applied to the geographic database, and produce data to be visualised on the screen or to produce hardcopy map products. It solves traditional cartographic problems and gives an acceptable visual output in cases such as object symbolisation conflicts solved by object displacement, combination of minor objects and exaggeration of important objects. These cartographic dilemmas are handled by the geometric editing capabilities of the database. It reveals that the system offers a great improvement in efficiency compared with traditional map generalisation process.

There is motivation within the mapping research community to focus on common web mapping platforms with generalisation functionality at varying scales (e.g. Lecordix *et al.*, 1997; Sarjakoski *et al.*, 2002; Jones and Ware, 2005; Ross *et al.*, 2007; Neun *et al.*, 2008). This involves exchange formats defined by the Open Geospatial Consortium (OGC), the World Wide Web Consortium (WWWC), GML, Web Map Services (WMS) (32) and Web Feature Services (WFS) (33) standards. Undertaking research on real time generalisation would make interesting subjects for further studies; however it is beyond the scope of this thesis.

Table 2.1 gives a brief summary of the major systems developed to date for generalisation. Each can be categorised into one of three classes: a) systems which attempt to provide full generalisation capabilities; b) systems with specific goals which attempt to generalise and characterise a set of objects (e.g. roads, buildings); and c) 'GIS operator-aid' systems which provide automated functionality in areas such as line simplification but which still rely on human intervention. Rather than presenting overviews of all theses systems, the most important generalisation algorithms (tools) are briefly discussed.

Table 2.1 Summary of generalisation systems

System Name	Vendor (s) and Reference (s)	Comments
ArcGIS Generaliser	ESRI	ArcGIS™ Generaliser algorithms offer selection and elimination based on size or attributes, simplification of linear features and area boundaries; simplification of buildings, preserving orthogonal corners; aggregation of polygonal features; and area collapse (www. esri.com).
DynaGen	Intergraph Corporation	DynaGen™ is a subsystem of Intergraph Dynamo™, which allows the use of a graphical environment, and topological functions, and the data models of Dynamo™. DynaGen™ software is a well-equipped cartographic suite of tools combined with Data Editor (DIDE) for creating or connecting to a database for further processing. Generalisation processes in DynaGen™ can be very detailed; users can take control of the data and monitor the whole process. Various algorithms, such as the Douglas-Peucker (1973) simplification, and Brophy smoothing algorithm (Brophy, 1973) have been tested with a variety of parameters.
Clarity	Laser-Scan	Laser-Scan developed object-oriented GIS LAMPS2 and now called Clarity that is an automatic solution / object-oriented framework for derivation of multiple databases from a master database. Laser-Scan's generalisation module (LAMPS2 software) matured through sponsorship of several European NMAs through initiation of MAGNET project in association with Laser-Scan to consolidate and extend current generalisation capabilities of Laser-Scan's new Clarity product. Capabilities include feature selection; alternative cartographic representations; thinning and sampling by point frequency; spatial filtering (e.g. removal of small islands); exploitation of multiple geometries; and generated alternative geometries (e.g. points from areas). Laser-Scan's generalisation tools offer a four step solution to generalisation, namely model generalisation using Gothic module, cartographic generalisation using Clarity module, Symbology placement using ClearText module (or other customised in-house product), and interactive editing tool using LAMPS2 (Neuffer *et al.*, 2004). (www.laser-scan.com)
CHANGE	Institute for Cartography of Hanover University	CHANGE emphasises the generalisation of large-scale data and is capable of generalising building and road objects with a scale range from 1:1,000 to 1:25,000. The CHANGE program generalises buildings through its sub-program of CHANGE-Buildings.
GenSystem	Canada Centre for Remote Sensing	Perceptual grouping for generalisation of roads and stream networks (see Thomson and Brooks, 2002).
LineDrive	Stanford University and Vicinity Corporation	LineDrive is an automatic route map generalisation system that is based on cognitive psychology to render route maps in real time using the generalisation techniques found in hand drawn maps. The system uses three main algorithms namely shape simplification, length generalisation, and angle generalisation (see Argawala and Stolte, 2001).
RoadMap Generaliser	Kreveld and Peschier (1998)	Measures such as length, distance, and connectivity were used in implementation of RoadMap Generaliser software.
Generalised Area Partitioning (GAP)	Oosterom (1995 and 2005)	GAP is an interactive generalisation 'on the fly' tool that creates a temporary generalisation data at an arbitrary scale on the screen from one detailed geographic database when maps are presented on the Web (on-line). Each area feature is stored at a hierarchic level that corresponds with its relative importance within the mapping area (a function of size and type in certain context); each point on the map will belong to exactly one of the areas or polygons (Cheng *et al.*, 2008). For example, the GAP-tree techniques used to decide which area to be removed and which area to fill the gap of the removed feature; this generally supports the aggregation and merging operators.
Road Generalisation Algorithm	Visvalingam and Williamson (1995)	The Road Generalisation Algorithm generalises a large scale road database that is operationalised as a point-based filtering and simplification algorithm through removing the least important points rather than selection of points in the Douglas-Peucker and Nth point schemes.

2.7 AVAILABLE GENERALISATION FRAMEWORKS

An excellent classification of generalisation assessment tools based on measures, conditions and the interpretation of generalisation results is provided by Skopeliti and Tsoulos (2001). Peter and Weibel (1999) presented a general framework for the generalisation of vector and raster data to achieve more effective translation generalisation constraints into assessment tools to carry out the necessary generalisation transformation. Peter (2001) developed a comprehensive set of measures that describe the geometric and semantic properties of map objects. These are the core parts of a generalisation workflow from initial assessment of the data and basic structural analysis, to identification of conflicts and guiding the transformation process via the generalisation operators, and then qualitative and quantitative evaluation of the results. The following discussion provides a critical review of the relevant generalisation research based on measures, constraints or limitations, and integration of measures into the generalisation process.

In connection with generalisation constraints, Peter (2001) categorised constraints based on their function (graphical, topological, structural and Gestalt) and spatial application scope (object level – micro, class level – macro, and group of objects/region/partition at the database level – meso). The constraints relevant to the micro level (object) include minimum distance and size (graphical), self-coalescence (graphical), supportability (graphical), separation (topological), islands (topological), self-intersection (topological), amalgamation (structural), collapsibility (structural), and shape (structural). To assess generalisation quality for linear features, constraints have been employed (Peter and Weibel, 1999; Yaolin *et al.*, 2001). Constraints for the micro level (object classes) include size ratio (structural), shape (structural), size distribution (structural) and alignment/pattern (Gestalt). Finally, Peter (2001) divided meso level (object groups) constraints into neighbourhood relationships (topological), spatial context (structural), aggregation (structural), auxiliary data (structural), alignment/pattern (Gestalt), and equal treatment (Gestalt). For a detailed description of the above constraints readers are referred to Peter and Weibel (1999); Peter (2001); and Jiang and Claramunt (2004).

There are several measures for the evaluation of generalisation results. These can be classified as either qualitative or quantitative methods. To date most of the generalisation transformation results have been evaluated qualitatively based on aesthetic measures. However, Skopeliti and Tsoulos (2001) developed a methodology to assess linear feature integrity by employing quantitative measures that determine if specific constraints are satisfied. Researchers began to develop formal approaches that integrated generalisation constraints and measures for development of coherent frameworks and workflows (e.g. Peter and Weibel, 1999; Yaolin *et al.*, 2001). In this regard, Skopeliti and Tsoulos (2001) incorporated positional accuracy measures to quantitatively describe horizontal position and shape, then to assess the positional deviation between the original and the generalised line, and to relate this to line length before and after the generalisation. A technique such as cluster analysis (qualitative assessment) was used for the line shape change and the averaged Euclidean distance (quantitative assessment). Also, McMaster (2001) discussed two basic measures for generalisation that include procedural measures and quality assessment measures. These measures involve the selection of a simplification algorithm, selection of an optimal tolerance value for a feature as complexity changes, density of features when performing aggregation and typification operations, determining transformation of a feature from one scale to another such as polygon to line, and computation of the curvature of a line segment to invoke a smoothing operation.

It should be noted that quality assessment measures evaluate both individual operations, e.g. the impact of simplification, and the overall quality of generalisation (i.e. poor, average, excellent). Despite all these efforts there is no comprehensive, universal and concrete process for generalisation measurement techniques. However, Kazemi and Lim (2007a) provided a review of existing measurement methods for automatic generalisation in order to design a new conceptual framework that manages the measures of intrinsic capability, so as to facilitate the design and implementation of a generalisation measurement library. To apply quantitative measures, Kazemi and Lim (2007b) used two methods of the Radical Law (Pfer and Pillewizer, 1966; Muller, 1995) and an interactive accuracy evaluation method to assess map derivation. The Radical Law determines the retained number of objects for a given scale change and the number of objects of the source map (Nakos, 1999).

While the majority of developed frameworks for the generalisation of cartographic data, such as those of Lee (1993), Brassel and Weibel (1988) and Ruas and Plazanet (1996), deliver generic procedural information (Peter and Weibel, 1999), the one briefly discussed in this chapter is designed more specifically for the derivation of multiple scale maps from a master road network database (Kazemi and Lim, 2007a).

The framework generalises a road network from 1:250,000 national topographic data scale to produce 1:500,000 and 1:1,000,000 scales using six key operations: *Classification, Selection, Elimination, Simplification, Typification, and Symbolisation* as detailed in **Section 4.2.3.** Large portions of the proposed framework may be considered generic (e.g. conditions/parameters/constraints definition). However, most parts deal specifically with road generalisation. Generalisation operators in the ArcGIS™ software are tested to generalise roads above the conceptual generalisation framework for derivative mapping. The method is empirically tested with a reference dataset consisting of several roads, which were generalised to produce outputs at 1:500,000 and 1:1,000,000 scales. The results show that the derived maps have reasonable agreements with the existing small scale road maps such as depicted on the Global Map at the 1:1,000,000 scale. As the methodology is only tested on roads, it is worthwhile to extend it to various other complex cartographic datasets such as drainage networks, power lines, and sewerage networks, in order to determine the suitability of the methodology proposed here. Additionally, various kinds of linear, areal and point cartographic entities (e.g. coastlines, rivers, vegetation boundaries, administration boundaries, land cover, localities and towers) should also be studied.

There is no universal semi-automatic cartographic generalisation process (GA, 2000; Lee, 2002), because off-the-shelf tools do not provide an aesthetically robust and pleasing cartographic solution. The current map production tools are significantly better than the map production systems of the 1990s in finding a reasonable solution to the challenge of deriving multiple scale data from a master database (Lee, 2004), and hardware performance and cost makes them suitable for implementation in a full production setting (Forghani *et al.,* 2003). For example, ESRI's current object-oriented software (version 9.3.2) provides a spatial framework to support generalisation needs by introducing geoprocessing concepts and map generalisation tools that have been

enhanced and implemented in the geoprocessing framework (Lee, 2003). The issue is to incorporate cartographer knowledge into the generalisation process to aid automation.

2.8 CHALLENGING PROBLEMS AND REMARKS

Research and development of mapping solutions over the past three decades has failed to provide the GIS community with reliable and robust computer generalisation tools that can compete with cartographers. Much more still remains to be done in generalising areal and linear features (e.g. vegetation, buildings and roads) in order to create maps at different scales, to maintain only one database and to derive the data for smaller scales.

This chapter firstly introduced 'derivative mapping' from a seamless database as an active research and development topic. This is an area of interest to many national mapping agencies, academia, map and spatial data providers and users across the spatial industry. It deals with a derivation of smaller scale map products from a detailed single master database. Then the chapter provided a brief review of 'generalisation'. This covers the concepts of cartographic generalisation, model generalisation and generalisation operators. Thirdly, the chapter provided an overview of well known generalisation tools which were developed and employed by the GIS and computer science communities over many years. The chapter then highlighted generalisation operations such as feature selection, feature simplification, feature classification and feature symbolisation. Finally, it presented research efforts from concepts to a set of recommendations for the development of practical generalisation frameworks. The motive is to generalise road network databases in association with integrating generalisation algorithms together with skilled cartographers' intuition in order to achieve the desired results.

A review of the literature demonstrates that future research and development work on automatic generalisation should focus on the following major streams. This judgment is supported by other researchers in the field of map generalisation (Meng, 1997; Lee, 2003):

- A need to evaluate and validate existing generalisation tools as identified by researchers (e.g. Visvalingam and Herbert, 1999), as well as improvements in editing tools (e.g. Muller, 1995) for both area generalisation and line

generalisation applications. To fulfil the need to evaluate and validate existing generalisation tools, the author's research will focus on the development of a detailed generalisation framework to derive multiscale spatial data. It focuses on integration and utilisation of generalisation operators as well as applying cartographer's intuition/skills using the ArcGIS™ Generalise and DynaMap™ Generaliser software in order to generalise a road network database from 1:250,000 national topographic data to produce smaller scale maps at 1:500,000 and 1:1,000,000.

- Maintaining a single sophisticated database that supports many applications (rather than multiple simplistic map layers), as well as a well-designed database, and provides a platform to support data derivation, generalisation, symbolisation, and updating (Lee, 2002). The idea is to associate geographic objects/features to multiple scales and maintain the cartographic quality of spatial data products. This requires the development of data models that support derivative mapping concepts. The appearance of many geographic objects varies with scale, so that it is difficult to encapsulate all possible details for all probable scales within a single data model. The way forward is to model data in an object-oriented solution.
- Development of universal guidelines to derive smaller scale products from a master database. As NMAs (e.g. Land Information New Zealand, Geoscience Australia, and Ordnance Survey) migrate their dataset into multiscale national seamless coverage, it is essential to develop guidelines and tools to derive smaller scale products from their fundamental spatial information (e.g. Geoscience Australia's 1:250,000 national topographic data) at a consistent level, as well as providing a basis for generalising other data sets at different levels of generalisation. The guidelines should also highlight both essential and desirable steps for generating smaller scale maps in line with a production environment focus. These include topological relations between the object types and classes, how the objects have to be selected, how to generalise, when to smooth, when to delete, when to merge, how to do reclassification of roads, and so on.
- A set of automatic generalisation tools, and efficient post-editing and cartographic editing tools are needed (Lee, 2002).

It should be noted that all the above developments and improvements will not be possible without a close cooperation between researchers, map producers, GIS software vendors and NMAs.

CHAPTER 3: STUDY AREA AND RESOURCES

3.1 STUDY AREA

The study area covers approximately 23,630 km² of Australia's capital city, Canberra (**Figure 3.1**). The area coverage of the chosen sites are in the longitude and latitude ranges 148° 42' 7" to 149° 25' 56" and 35° 55' 35" to 35° 7' 22" respectively, with an elevation of 550-700m above sea level. The study site was chosen because it provided a mix of different roads with a reasonable amount of feature density. The selection of the area was based on testing the generalisation concepts over an urban environment, since the density of roads was a determinant factor, and for ease of access to the datasets.

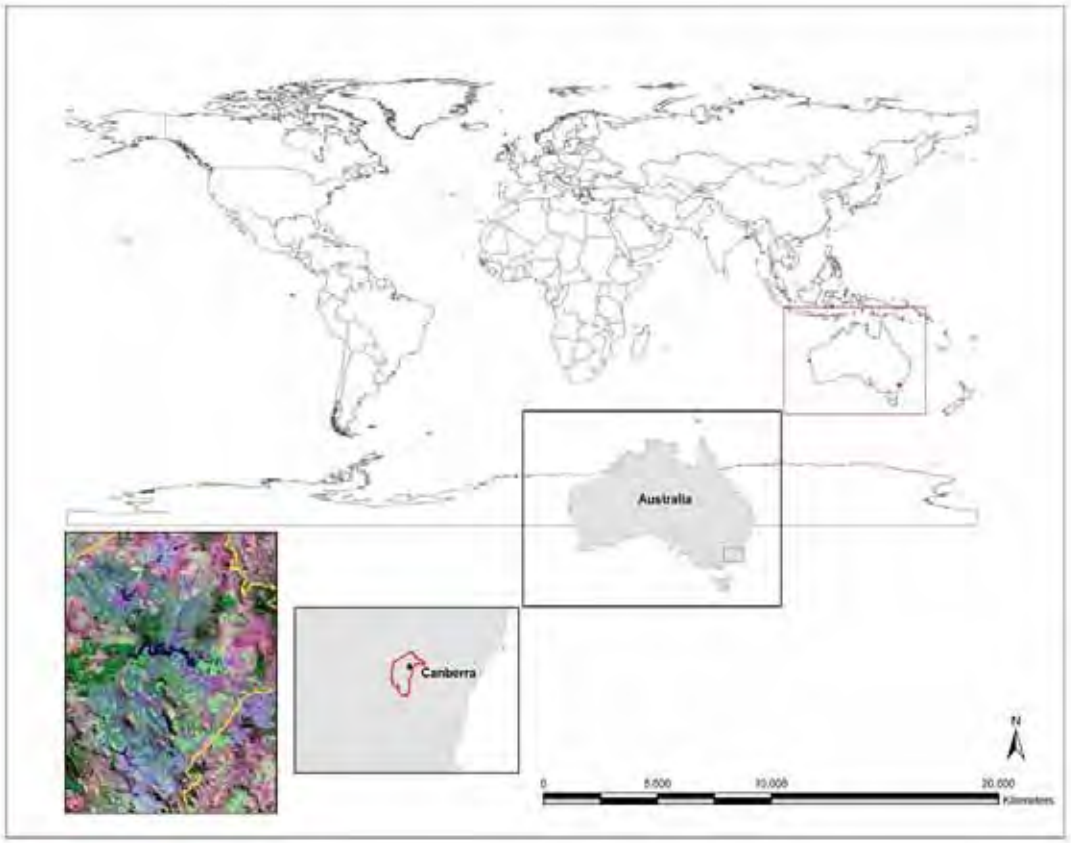


Figure 3.1 Location of study area, Canberra, Australian Capital Territory (ACT) Australia; Study site is shown in Landsat ETM+ image (composite bands 6,4,2), courtesy of Geoscience Australia © 2001. The Landsat image is not to the same scale.

It is worthwhile to discuss this data in some detail. Roads in the 1:250,000 national topographic data have varying widths and types, but six road classes can be distinguished (GA, 1999): highways, major roads, secondary roads, minor roads, and foot tracks. Because roads in the area have different widths, the vector coverage of roads was overlaid on a panchromatic sharpened Landsat Enhanced Thematic Mapper (ETM+) image (15m resolution with bands 2, 4, and 7), and every road segment was labelled with the corresponding widths, based on visual assessment. The ETM+ imagery was used as a backdrop layer to validate the roads class of 1:250,000 national topographic data, and also facilitated determination of road widths for the buffering process.

The roads were classified into six categories with buffer distances ranging from 1 to 6 metres to overlay the road polygons over the ETM+ data. Principal roads (e.g. main roads in urban areas) can be extended in two directions, starting from north to southeast or to southwest of the study area. Their widths range from 5 to 10 metres. Minor roads are concentrated in the central part of the study area. Their widths are between 4 and 8 metres. Minor roads are located mostly in the recreational areas such as parks. These roads include rural roads, access roads and tracks. Their widths vary from 2.5 to 5 metres. Many of these roads (e.g. tracks, minor roads) were not discernible on ETM+ imagery due to the resolution of the image and the tree canopy.

3.2 AVAILABLE DATA

3.2.1 Data Collection

Acquiring a dataset over Canberra was one of the key tasks for this study (**Table 3.1**). The digital and hardcopy data were partially provided by Geoscience Australia. Additional information, such as topographic maps at 1:10,000, 1:25,000 1:50,000, 1:100,000, 1:190,000 and the digital 1:250,000 national topographic data, town planning maps and land tenure maps were also collected to assist road generalisation. Ground truth information was also collected to confirm some of the road labelling from the imagery and the 1:250,000 national topographic data. Geoscience Australia (GA) 1:250,000 national topographic data, International Steering Committee for Global Mapping (ISCGM) 1:1,000,000 Global Map and Geoscience Australia's National Earth

Observation Landsat 7 imagery were the main reference spatial datasets used in this research. In addition, GPS survey points and GPS tracks were acquired in order to perform accuracy assessment of generalised road maps. The survey point dataset was obtained from GA in February 2007 and post-processing for the 70 GPS survey points was performed by Ultimate Positioning Pty Ltd. All points are road intersections. There are 10 GPS points covering the study area that were used for analysing the locational accuracy of simplified roads**.** The study also used about 50 ground control points (GCPs) that were identified from the GPS track data and the 1:25,000 topographic map. These GCPs and GPS survey data were overlaid onto the generalised output maps for accuracy assessment of the average shape changes caused by the simplification operation (**Chapter 7**).

Table 3.1 Collected data over the study area

Data and Maps	Description
Hardcopy	1:10,000, 1:25,000 1:50,000, 1:100,000, 1:190,000, 1:250,000 and 1:1,000,000
Digital Vector Data	1:250,000 national topographic data
Satellite Imagery	Landsat-7 Pan and Multispectral as well as SPOT-4 images

3.2.2 Image Acquisition

A set of satellite ETM+ imagery and SPOT-4 was selected by searching the GA archive for cloud-free images (**Table 3.2**).

Table 3.2 Remote sensing dataset specification

Imagery	Pixel Size (m)	Mode	Bits	Band Width (μm)	Date
ETM+	15	Pan	8	0.52 - 0.90	15 August 2001
ETM+	30	MS	8	0.45- 2.35	15 August 2001
SPOT-4	10	Pan	8	0.51 - 0.73	10 August 2001

3.2.3 Base Maps

The process used to prepare data is depicted in **Figures 3.2** and **3.3**. Database development includes processing vector data, scanning hardcopy maps and geocoding raster maps. The spatial data processing and manipulation steps can be categorised as follows:

- Data entry; the first step in a GIS project is data capture. This phase (data entry in both raster and vector forms) is one of the most costly, time-consuming and

tedious tasks in the development of any GIS project. Hardcopy maps were scanned with a resolution of 500 DPI and stored in TIFF format. The digital vector data was provided by GA in Arc/Info Export format, which was converted into ESRI shapefiles.

- Database development; a sub-region of each dataset was separately created for the process of image geocoding over the trial test area, which is measured 8 km by 8 km.
- Geocoding; for better visual interpretation in relation to evaluation of the accuracy of output maps, it is crucial to geometrically correct the data in order to match the corresponding raster pixels to points in the vector data. All scanned maps were geocoded to the digital 1:250,000 national topographic dataset reference system (geodetic datum GDA94) with less than 5 metre Root Mean Square Error (RMSE) per map using ERDAS IMAGINE™ software. The topographic maps and SPOT-4 as well as Landsat ETM+ imagery served as reference bases for creation of georeferenced raster scanned maps. GCPs were selected from the 1:250,000 national topographic data. Thus raster data were geo-referenced to GDA94. Ten well-distributed GPCs were selected from the images. The RMS residual for these control points was about 5 metres and the maximum residual was 4.3 metres in the X-axis component and 3.8 metres in the Y-axis component. The maps were then gecoded with first order polynomial transformation and nearest neighbour–interpolation resampling techniques. An additional qualitative check was made on the accuracy of the geocoded maps by displaying maps in ArcGIS™ (versions 8.2 and 9.0) and overlaying road networks on the scanned maps. Visual checks of the maps and images revealed that the geocoding results were acceptable.
- All satellite images were radiometrically enhanced to the required colour; contrast and brightness, using a Histogram Equalisation technique.
- Spatial resolution merge; the resolution merge method offered by the ERDAS IMAGINE™ software was used to integrate imagery of different spatial resolutions (pixel size); ETM+ (30m) multispectral (MS), and ETM+ (15m) Pan imagery to produce high resolution, multispectral imagery. This improves the interpretability of the data by having high resolution information, which is also in colour. Resolution merge offers three techniques: Multiplicative, Principal

Components, and Brovey Transform, to which the Principal Components algorithm was applied. Descriptions of these techniques are available in Leica (2004). Finally, a Bilinear Interpolation resampling technique was adopted to resample the multispectral input to match the high resolution image. The final enhanced output is a 15m MS with an 8-bit colour image. This data will be used mainly for visual assessment of derived maps.

3.2.3.1 Roads Classification

Roads in the 1:250,000 national topographic data specifications have varying width and type. In the classification, six road classes can be distinguished in the study:

- Dual Carriageway (DC) – divided highway, freeway (e.g. Federal Highway), tollway, and other major roads with separated carriageways in opposite directions.
- Principal road (PR) – highways, major through routes and major connecting roads as described by the Australian Automobile Association (AAA) (both sealed and unsealed).
- Secondary Road (SR) – connecting roads (both sealed and unsealed) that provide a connection between major through routes and major connecting roads or between regional centres. All principal and secondary roads are shown including those in built-up areas.
- Minor Road (MR) – all other roads (both sealed and unsealed) that form part of the public roads system between principal roads and secondary roads.
- Vehicle Track (VT) – public or private roadways of minimum or no construction which are not necessarily maintained.
- Foot Tracks (FT) – these tracks are for pedestrian traffic only. Tracks with a length of less than1.25 kilometres have not been captured.

Major roads (e.g. Northbourne Avenue) are located in the north and south of the study area. Because roads in the study area have different widths, the road vector coverage was overlaid on the fused ETM+ imagery (15m colour), and every road segment was labelled with a specific distance for its width based on visual assessment. The roads were classified into eight categories with buffer distances ranging from 1 to 6 metres. Principal roads (e.g. urban main roads) can be seen extending in two directions, starting from the north and extending to the southeast and to the southwest. The width of these

is between 5 to 10 metres. Minor roads are concentrated mainly in the central part of the study area. The width of these streets is 4 to 8 metres. Minor roads are located mostly in the recreational areas. These roads include rural roads, access roads, and unpaved tracks. The width of these roads varies from 2.5 to 5 metres.

Figure 3.2 illustrates the different roads in the study area. The process of developing GIS data for this study is the most important step in developing a guideline for derivative mapping using software applications to generalise 1:250,000 national topographic roads to produce 1:500,000 and 1:1,000,000 roads spatial data and maps. This process is depicted in **Figures 3.3** and **3.4**.

(a) (b) (c) (d)

Figure 3.2 Examples of roads within the study area (a) Corner of Hindmarsh Drive and Jerrabomberra Avenue, Symonston (Dual carriageway), (b) Commonwealth Avenue, Barton (Principal road), (c) Intersection of Melrose Drive and Yarra Glen Drive, Woden (Secondary road) and (d) Intersection of Leichhardt Street and Dawes Street, Kingston (Minor road).

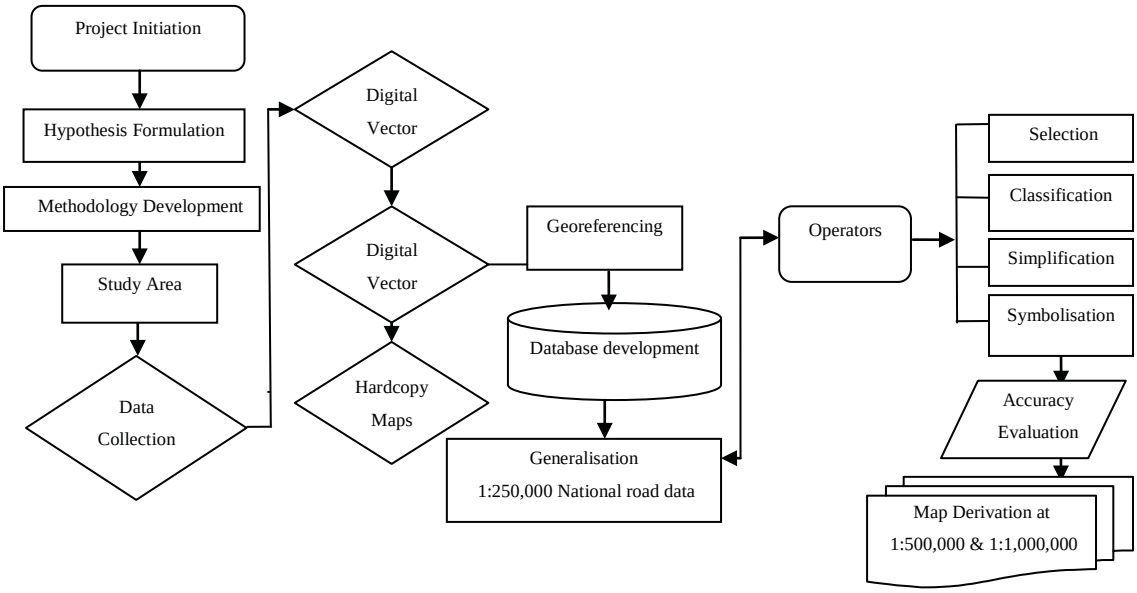


Figure 3.3 An overview of the methodology

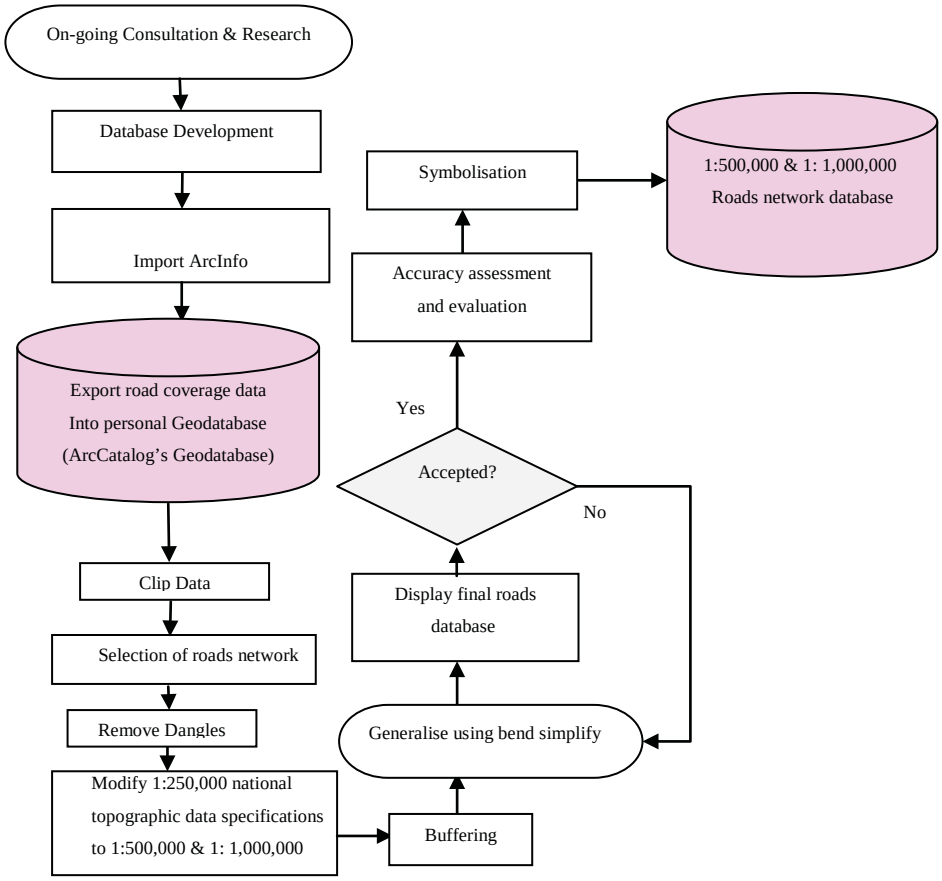


Figure 3.4 Flow chart of derivation of 1:500,000 and 1:1,000,000 spatial datasets from 1:250,000 national topographic data

3.3 COMPUTING RESOURCES

ArcGIS™ and Intergraph DynaGen™ were used as key GIS software. GES was interfaced with ArcGIS™. Other software such as ERDAS IMAGINE™ and Laser-Scan Clarity™ were also used. GES implementation was carried out in Java-Python-C programming environments that were selected for their user-friendliness, flexibility, interface capabilities, high level programming language and the familiarity of the author with these languages.

Table 3.3 Computing resources information

Type	Name , Model/Version	Comments
Laptop	Compaq Presario 2000	A laptop with 80 Gbytes disk space and 1GB RAM was used mainly for data processing, analysis, and writing the documents.
Image Processing	ERDAS IMAGINE™ Version 8.5 software	This software was used for part of the image geo-referencing for this project.
GIS	ArcGIS™ Version 8.2 and 9.2 software	ArcGIS™ was used to pre-processing, manipulation, generalisation and map presentation in this study. The ArcToolbox Generalise™ was also used for line simplification applying *Pointremove* algorithm.
	Intergraph DynaGen™ Version 3.0	DynaGen™ was used as a testbed for roads generalisation in this study.
	Laser-Scan Clarity™ Version 4.0	Clarity™ was also used as a testbed for roads generalisation in this study.
GES	GES	GES implementation was carried out in Java-Python-C programming environments.

3.4 SUMMARY

This chapter provides a description of the study area and the available data. The study site is approximately 23,630 km² in area, contains urban and rural areas, and is located on the southern fringe of Canberra, Australia. The area was chosen because it provides a mixture of different roads, and because of ease of access of the datasets. The chapter also lists the hardware and software utilised in this research, consequently outlining the implemented generalisation software for the development and testing of the methodology.

CHAPTER 4: DEVELOPMENT OF A GENERALISATION FRAMEWORK TO DERIVE MULTISCALE SPATIAL DATASETS

4.1 INTRODUCTION

National mapping agencies (NMAs), spatial data providers and map producers have been maintaining multiple databases at different scales for different uses. They are also committed to maintaining a set of cartographic data with synchronised updates. Maintaining multiple databases is resource-intensive, time-consuming and cumbersome (Arnold and Wright, 2005). This is a major challenge for NMAs and other map producers (Sarjakoski *et al.*, 2002). A seamless master database should offer capabilities to derive various types of maps at different scales, e.g. at scales ranging from 1:250,000 to 1:10,000,000. Derivative mapping has been identified as an active research and development area for NMAs, industry and academia to accomplish the requirements for evaluation and validation of generalisation (Muller, 1995; Visvalingam and Herbert, 1999; Kazemi and Lim, 2007a). Therefore this research has focused on application of generalisation tools to the development of a detailed generalisation framework for deriving multiscale spatial data. More particularly, the research has focused on utilising the generalisation operators, combined with the skills of a trained cartographer, to generalise a road network database from 1:250,000 national topographic data and in order to produce smaller scale maps at 1:500,000 and 1:1,000,000.

4.1.1 Relevant Work on Thresholding for Line Simplification

Thresholding techniques have been applied in both image processing and map generalisation. When using the ArcToolbox Generalise tool, an appropriate weed threshold level must be determined. For the appropriate tolerance selection Shea and McMaster (1989) argue: '*…the input parameter (tolerance) selection most probably results in more variation in the final result than either the generalisation operator or algorithm selection.*' To decide how much information to discard while retaining useful information is a difficult task in line simplification. In fact, it is a very difficult and tedious task to set an appropriate threshold (Zhao *et al.*, 2001).

The basic question remains: what is the best threshold? Generally, the threshold value is determined by a given application. Mackaness *et al.*, (1998) stated '*…the loss in total*

line length proceeds at a rate that is approximately linear to the increase in tolerance. Low tolerances produce high rates of change in vertex numbers and low changes in total line length and using higher tolerances increasingly fewer vertices are removed for a rapidly increasing loss in total line length.' The threshold value depends mostly on map content, map quality, map geometry, and the amount of noise present in the database. For operational production cartographic environments, generalisation capabilities of current GIS software tools such as Generalise™ are likely to use automated threshold selection in order to allow the complete cartographic production workflow to be carried out in a uniform environment (Hardy *et al.,* 2004).

Line simplification algorithms are mainly based on geometric principles (Cromley, 1992) with the reduction rate affected by a predefined tolerance (Nakos, 1999). The current way of selecting tolerance is by trial and error. Different levels of tolerance, ranging from 10m to 70m, were tested in the course of road network generalisation. Tolerances of 25m and 35m were acceptable for the 1:500,000 and 1:1,000,000 data, respectively. Similar tolerances have been used in other studies (Visvalingam and Williamson, 1995; Nakos, 1999). In practice, generalisation involves thinning out coordinates (vertices) to simplify line strings. In this way, redundant points along lines can be efficiently removed and the changes of line shape are minimised. Approximately 25% of vertices in this study were eliminated, while the length of the lines did not change significantly. From a geometry perspective, a vertex (plural vertices) is a special kind of point that describes the corners or intersections of geometric shapes. Vertices are commonly used in computer graphics and cartographic mapping to define the corners of surfaces (typically triangles) in 3D models, where each such point is given as a vector.

The *Pointremove* operator (based on the DP algorithm implemented in ArcGIS™) proved to achieve satisfactory line thinning/ results (Lee, 2004). It removes vertices quite effectively, but produces angularity along lines that is not very pleasing from an aesthetic point of view (Lee and Hardy, 2006). In addition, in some instances, cartographic manual editing is required. The *Bendsimplify* operator is designed to preserve cartographic quality. It removes small bends along the lines and maintains the smoothness (Lee, 1992). Bends are defined as sections of curves between two inflection

points. Applications of these algorithms are well documented in the literature (e.g. Jenks, 1989; Mroz *et al.*, 1996; Nakos, 1999). It would be interesting to compare the performance of the two in future research. For example, at 1:500,000 the *Bendsimplify* operation resulted in reduction of 760 vertices (after generalisation) out of 4,750 vertices (before generalisation) over a subset of road dataset.

Wang and Muller (1998) introduced the idea of an iterative bend simplification approach using Generalise™ *Bendsimplify* operator that detects bends in lines to drive the line generalisation process; it was proposed as a form of structure-based generalisation. The operation reduces the number of vertices significantly.

This chapter gives a brief discussion of a conceptual generalisation framework that forms the methodology for road network generalisation with special emphasis on simplification. This is followed by the analysis of results and evaluation of derived maps based on the *Radical Law* approach (Pfer and Pillewizer, 1966; Muller, 1995). Finally, it concludes with discussion and indicated research directions.

4.2 METHODOLOGY

4.2.1 Software and Generalisation Tool

The ArcToolbox Generalise™ tool offers a set of line simplification algorithms such as *Pointremove* that uses the *Douglas-Peucker* (DP) algorithm, which keeps the so-called critical points that depict the essential shape of a line and removes all other points. The algorithm connects the end nodes of an arc with a *trend line*. The distance of each vertex to the trend line is measured perpendicularly. Vertices closer to the line than the tolerance are eliminated. The line (arc) is divided by the vertex farthest from the trend line, which makes two new trend lines. The remaining vertices are measured against these lines, and the process continues until all vertices within the tolerance are eliminated (Mroz *et al.*, 1996).

The DP algorithm attempts to preserve directional trends in a line using a specified tolerance factor (Taylor, 2005) by removing vertices from the selected lines and simplifying the line bends. The DP algorithm is one of the most popular and accurate generalisation algorithms available (Jenks, 1989) and has been widely used for many

cartographic applications such as coastline generalisation (Dutton, 1999; Nakos, 1999), river generalisation (Rusak and Castner 1990; Moreno *et al.,* 2002), and road generalisation (Kreveld and Peschier, 1998; Skopeliti and Tsoulos, 2001). The area includes a range of road types described as part of the 1:250,000 national topographic data specifications. Consideration is given to show appropriate level of detail on the maps rather than producing the maps to the scale.

4.2.2 Roads Classification

The National Mapping and Information Group of Geoscience Australia (GA) produce 1:250,000 seamless topographic data and associated cartographic databases across Australia that is used in this research. Roads in the 1:250,000 national topographic data have varying widths and types (GA, 1999). The roads classification is detailed in **Chapter 3** (**Section 3.2.3**).

4.2.3 Conceptual Framework

In order to gain an appreciation of the conceptual generalisation framework, it is worthwhile to look at the main steps of the research methodology in **Figure 4.1**. This is linked to the conceptual framework that is presented in **Figure 4.2**. To date, many frameworks/workflows have been developed for the generalisation of cartographic data (see Kazemi *et al.,* 2004b). It seems that many of these frameworks are generic and do not offer a total solution for the operational environment. The motive of this research is to develop a workflow specifically for derivation of multiple scale maps from a master road network database. A large portion of this framework may be considered as generic too. However in this chapter all the processes are considered specifically for road generalisation (refer to Kazemi and Lim, 2007b, for further details).

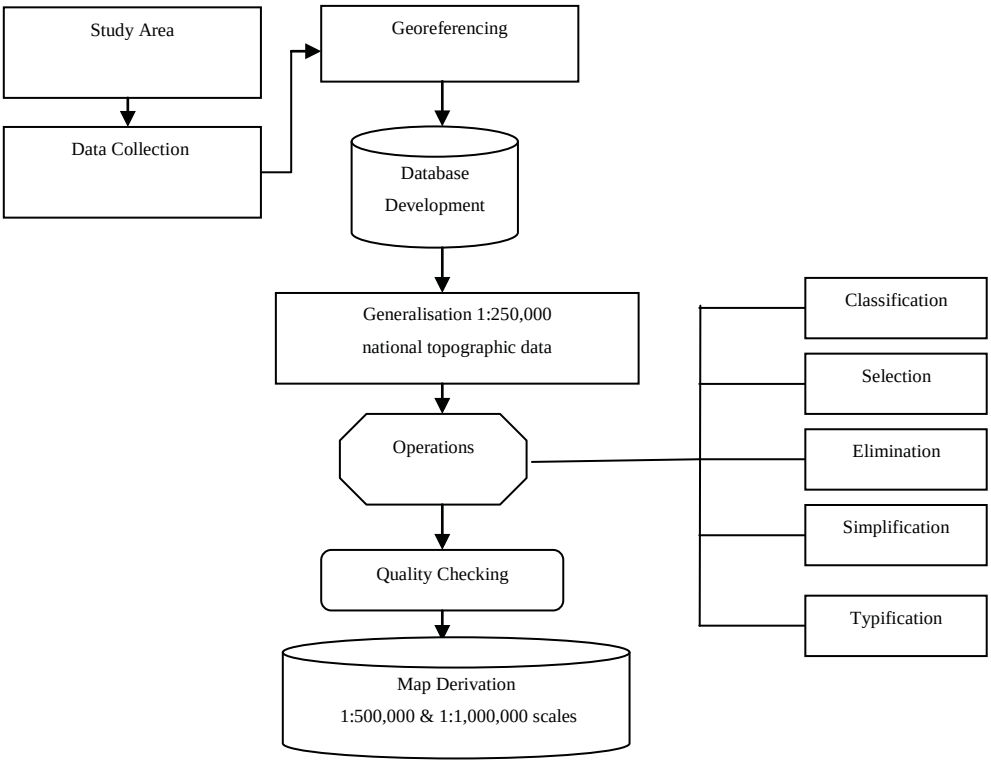


Figure 4.1 Schematic representation of the research methodology

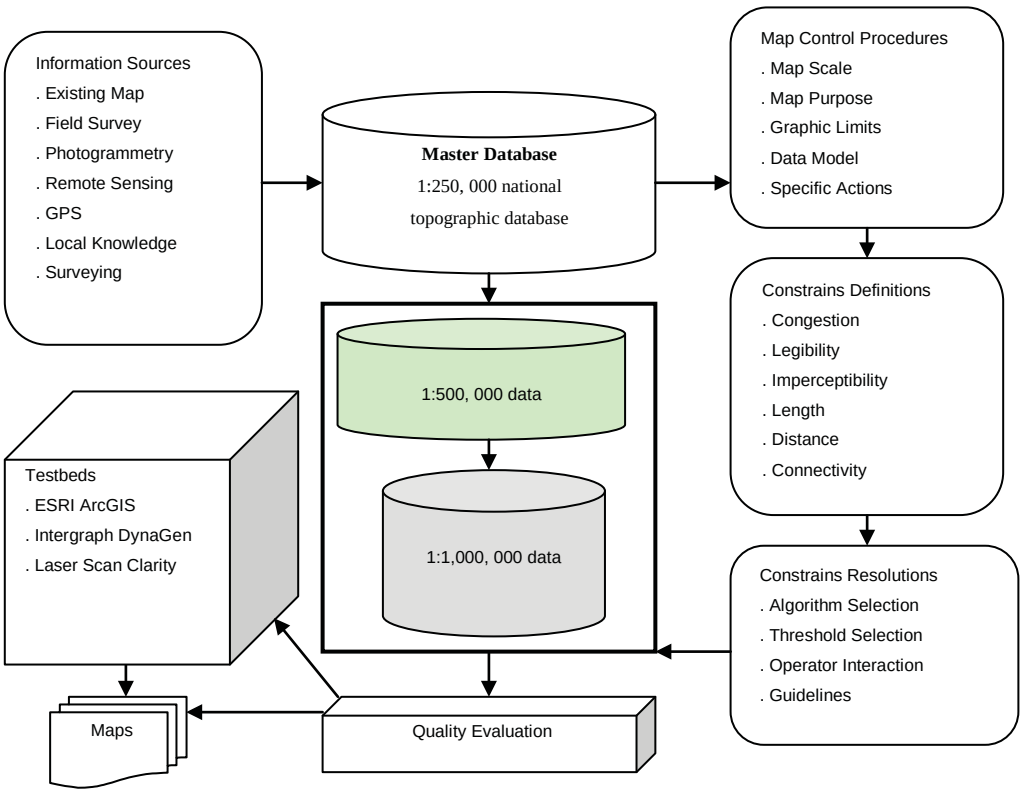


Figure 4.2 Framework for road generalisation

There are a number of properties that could be considered for generalisation of roads to maintain the legibility, the visual identity of each road segment and the pattern. Out of the six generalisation properties (reducing complexity, maintaining spatial accuracy, maintaining attribute accuracy, maintaining aesthetic quality, maintaining a logical hierarchy, and consistently applying generalisation rules) highlighted by Shea and McMaster (1989), the following four are taken into consideration in this research: *Congestion/Legibility, Coalescence, Imperceptibility* and *Length/Distance*. Some of the Shea & McMaster (1989) and Robinson *et al.,* (1984) properties (e.g. length, connectivity, elimination) have been also used by Kreveld and Peschier (1998).

According to Lee (2003), and collected feedback through a Cartographic Generalisation Survey (**Chapter 6**), generalisation of a network (for example, roads, rivers, or other linear features) requires at least six key operations/processes: *Classification, Selection, Elimination, Simplification, Typification, and Symbolisation* (**Figures 4.3 – 4.5**). They are therefore used in the generalisation framework of this research discussed in the next section. The generalisation operators of ArcToolbox Generalise™ were tested for the generalisation of roads by employing the conceptual generalisation framework for derivative mapping, but the results showed that the algorithms of *Douglas-Peucker* built into ArcToolbox Generalise™ (*Bendsimplify and Pointremove)* do not support dynamic generalisation. However, the way to work around this problem is to use Intergraph DynaGen™ generalisation system which enables the derivation of a multiscale database from a master database. The author has demonstrated that testing and incorporating of the DynaGen™ generalisation system enhanced the conceptual generalisation framework (**Chapter 5**).

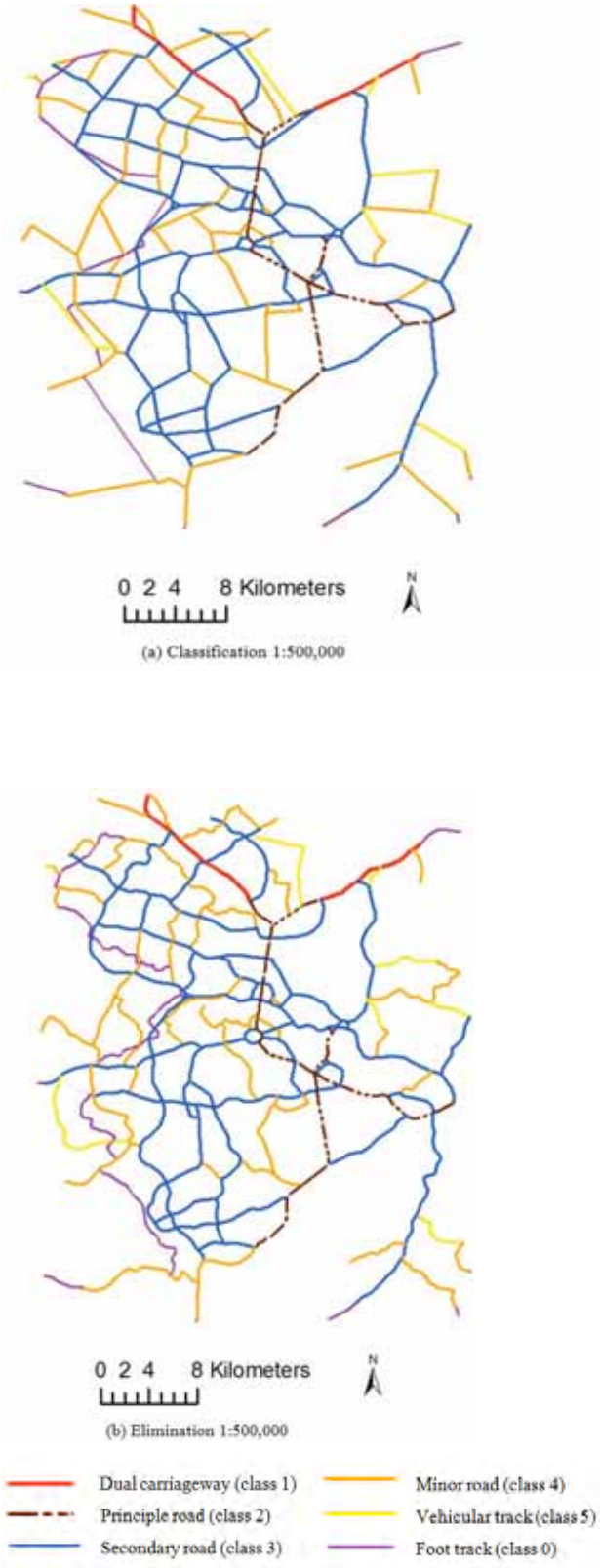


Figure 4.3 Road generalisation outputs (stage 1); deriving 1:500,000 maps from the source data at 1:250,000; (a) Classification and (b) Elimination

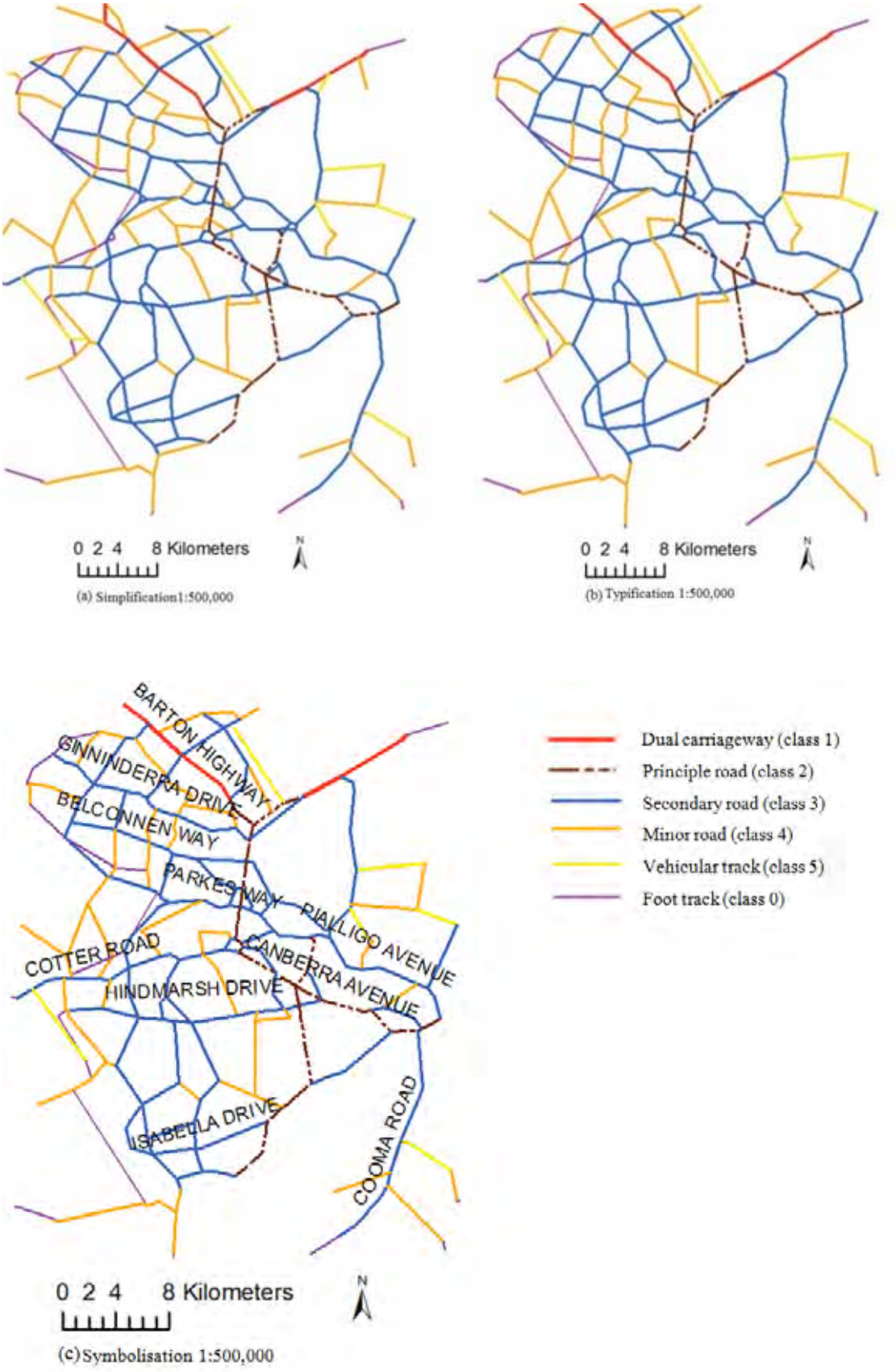


Figure 4.4 Road generalisation outputs (stage 2); deriving 1:500,000 maps from the source data at 1:250,000; (a) Simplification, (b) Typification and (c) Symbolisation

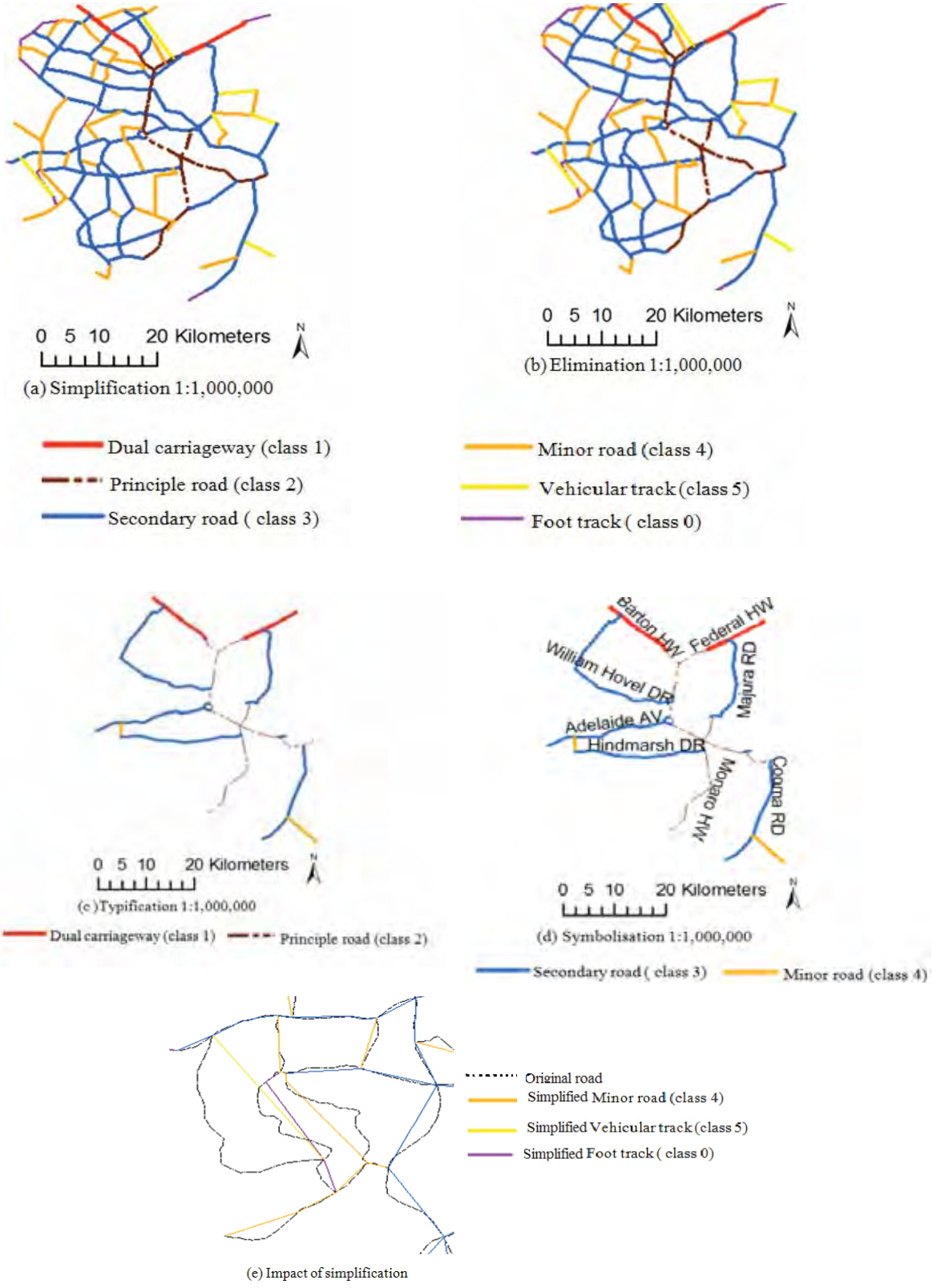


Figure 4.5 Road generalisation outputs; deriving 1:1,000,000 maps from newly derived 1:500,000 scale data; (a) Simplification, (b) Elimination, (c) Typification, (d) Symbolisation and (e) a zoom area from middle-left of (a) showing the impact of simplification on a portion of the road network

4.2.4 Road Generalisation

The following operations are required to generalise a road network (Lee, 2003) (the outputs of generalisation workflow applying these operations are shown in **Figures 4.3 – 4.5**):

- *Classification:* a good classification of roads makes the selection easier and more accurate. This step identifies objects that are placed in groups according to similar properties. It also reduces the complexity and will improve the organisation of a map. For example, roads were categorised into 5 classes with widths ranging from 2m to 12m.
- *Selection:* only certain road classes are selected for inclusion at the target scale. For instance, all vehicle tracks are dropped from the 1:250,000 road database when deriving 1:500,000 scale roads data.
- *Elimination:* mapped features that are not relevant to the map's purpose (Robinson *et al.*, 1984) - e.g. a road branch shorter than a certain length or small road segments that can cause conflict in the final map and are not significant for representation on the map - can be eliminated. GA (2000) applied this operation to delete small buildings, short roads, and small villages in a generalisation of 1:250,000 scale data to generate the 1:1,000,000 *Global Map*. The Global Map Australia 1:1,000,000 (2001) is a digital spatial dataset covering the Australian continent and island territories at 1:1,000,000 scale which consists of eight spatial layers in vector form (administrative boundaries, drainage, transportation and population centres) as well as incorporating raster images (elevation, vegetation, land cover and land use). This is created from GA's 1:250,000 national topographic data. The United States Geological Survey (USGS) provided the raster images.
- *Simplification:* selected roads can be simplified to reduce the details. This research used the *Pointremove* and *Bendsimplify* operators to remove vertices from the selected lines and to simplify extraneous bends of roads.
- *Typification:* this function is not directly available in ArcToolbox Generalise™, but a manual cartographic editing approach was used. The basic aim is to reduce the density of features and simplify the distribution and the pattern of the network. The result should be a connected and less congested network that represents a similar pattern at the smaller scale.

- *Symbolisation:* the systematic process of creating graphic marks, which represents the objects and features to be mapped. For example, roads are symbolised according to their class.

Generalisation operators are applied in order to derive small scale maps in this investigation. To commence generalisation, necessary objects in the map should be chosen. The next step is to apply simplification, which may include smoothing of curves and straightening paths. The objective of simplification is to increase the legibility of the map by removing unnecessary details of the road network. In this simplification process the road line features are simplified to remove extra vertices in the road segments. ArcToolbox Generalise™ (*Pointremove*) is used to simplify the lines for this process, and is based on the DP algorithm that removes vertices from the road linear features with a user-defined tolerance. To simplify the line bends, the bend operator is used to omit extra bends. Tolerances for each scale of the roads were based upon comparison of the generalisation results with cartographic products of the same scale.

4.2.5 Road Generalisation Assessment Methods

In this study an enriched database (1:250,000 national topographic data) was used as the main reference dataset. In total 1,202 road segments were present in the 1:250,000 national topographic data database over the study area. Changes in the representation of road features at 1:250,000, 1:500,000 and 1:1,000,000 scales due to generalisation were assessed using the *Radical Law* (Pfer and Pillewiser, 1966; Muller, 1995) and an interactive accuracy evaluation method (**Figures 4.6 – 4.11**). The road type was also considered a major index of importance. During the transformation of road representation from the 1:250,000 to 1:1,000,000 scales some of the less important road segments were deleted. At each scale the number of road segments classified by road types was counted and the results were graphically illustrated (**Section 4.4**).

4.3 RESULTS AND DISCUSSION

All vehicle tracks disappeared at the 1:1,000,000 scale. In addition there was a 61% reduction of minor roads at 1:500,000 and a further 45% reduction of minor roads at 1:1,000,000 (**Figure 4.6**). For example at 1:500,000 scale, there are 10 Minor Road

(MR), 8 Secondary Road (SR), 3 Principal Road (PR), and 1 Dual Carriageway, but no Vehicle Tracks (VT).

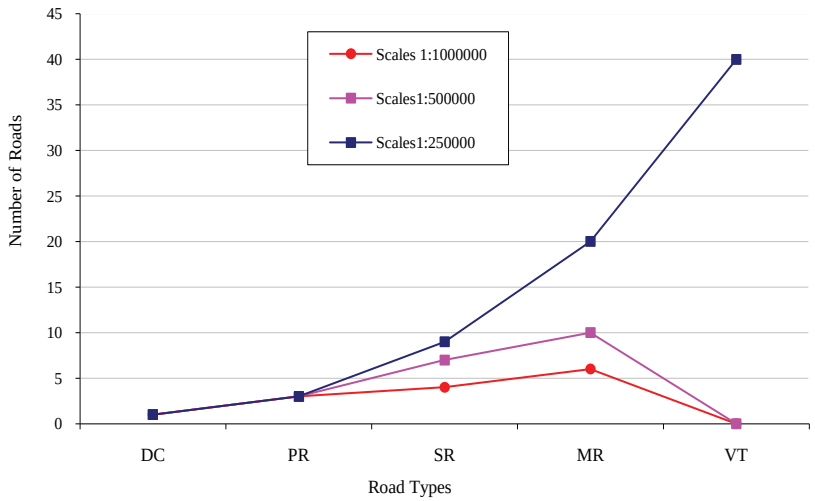


Figure 4.6 Road classification at different scales ranging from 1:250,000, 1:500,000 and 1:1,000,000 and number of road segments in different classes

The results of map derivation were considered using two methods: the *Radical Law* and an interactive accuracy evaluation method. The *Radical Law* determines the retained number of objects for a given scale change and the number of objects of the source map (Nakos, 1999). The original 1:250,000 national topographic data roads has 1,202 segments and the derived roads have 856 segments before the line simplification operation. The equation (**4.1**) computes the number of roads in the map after the line simplification operation:

$$n_T \approx n_S \sqrt{S_S / S_T} \qquad \textbf{(4.1)}$$

where n_S is the number of objects in source dataset; S_S is source scale; S_T is scale after transformation; and n_T is number of objects in the dataset after transformation.

- Number of lines in 1:500,000 scale roads map => $n_T \approx 1202\sqrt{250/500} = 849.9$
- Number of lines in 1:1,000,000 scale roads map => $n_T \approx 856\sqrt{500/1000} = 605.3$

The numerical value 849.9 represents the predicted number of segments after generating the 1:500,000 scale road map, and the value 605.3 predicts the number of segments after

generating the 1:1,000,000 scale map. These results have also been analysed visually since evaluation is a critical element of semi-automated road generalisation. Cartographic knowledge, such as the presentation of features on the map and the contextual information of features, has been combined with generalisation operators to derive multiple scale maps. This includes cartographic knowledge and information about roads and surrounding areas. In order to perform an intelligent and efficient generalisation framework, Raisz (1962) suggested that the processes of combination, omission and simplification should address a linkage between having a good knowledge of geography and a sense of proportion. Design and representation of map aspects are considered in relation to text portrayal. Map symbols must exhibit a number of key characteristics, including a) designative, reflecting the features portrayed by other map symbols; b) analytical, linking features on the map with their attributes and analysing relationships amongst features; c) positional, describing or confirming the location of features; and d) informative, giving a description of the nature of the source data (Fairbairn, 1993).

Thematic properties such as length, width, and connectivity were also used. Although these measures are not directly available in the Generalise™ tool, the roads database specification, local and cognitive knowledge, and other ancillary information were employed to characterise these thematic properties. The information was used to determine which road should be deleted or merged in the generalisation process in order to retain a certain percentage of roads at a smaller scale. In this regard, Choi and Li (2000) believe that incorporation of thematic information can control geometric operations (e.g. simplification) and topological information (e.g. connectivity). Thematic information can be used to validate the geometric operation and to control the quality of map generalisation. For example, the use of Elimination involves removal of very short branches and unimportant roads, which is mainly based on cognitive knowledge. The generalisation processes such as Elimination and Typification results in a reduction of the complexity and the density of roads (**Figure 4.7** - examples of the effects of various tolerance levels).

Figure 4.7 The impact of generalisation using 25m tolerance

The results were evaluated through cartographic visual interpretation The Generalise™ tools (Bendsimplify and Pointremove) did not smooth some portions of roads satisfactorily. Previous studies (e.g. Visvalingam and Williamson, 1995) confirmed that the DP algorithm, by applying very low tolerances, resulted in removal of an appropriate number of vertices. It was found that generalisation of roads requires a very large volume of processing to transform data from 1:250,000 scale to 1:500,000 and 1:1,000,000 scale maps. The DP algorithm eliminates shorter branches and results in simplification of the road network, but also improves the preservation of the characteristics of the structure.

The impact of generalisation tolerance can be seen in various portions of the roads (e.g. see circle around Antill Street) database; road network is overlaid on merged Landsat 7 ETM+ Panchromatic (15m) and Multispectral (30m) imagery (composite bands 1, 2, 3).

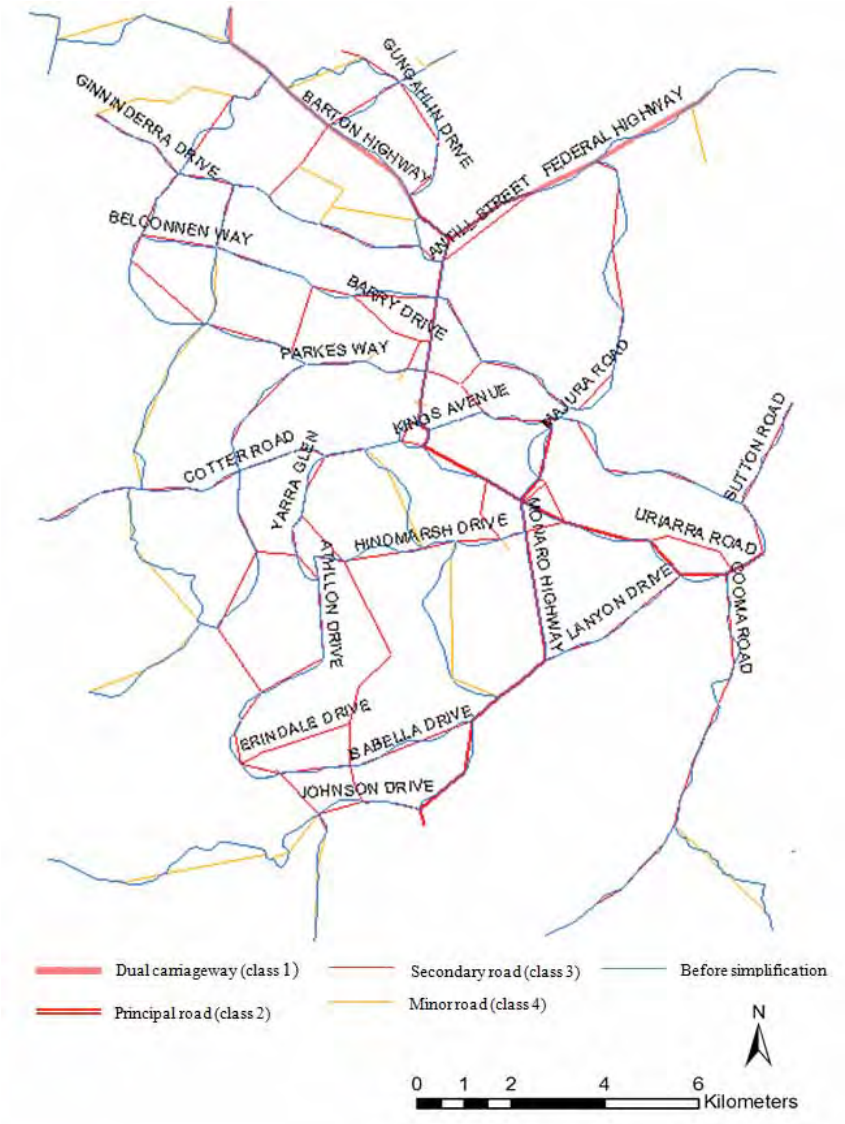


Figure 4.8 The impact of generalisation tolerance of 45m

Figure 4.8 and **Figure 4.9** compare the road network before simplification (blue lines) and after simplification (those lines shown in colours other than blue) and show the results of derived maps at 1:500,000 and 1:1,000,000 scales. Then the road network was symbolised. The reduction of the 1:500,000 scale map is less readable than the 1:1,000,000 map. The objects are too small and too close to be readable. By comparing

the two, some objects have been enlarged. At a higher level, some objects have been removed, but use of space has been preserved at the 1:500,000 scale.

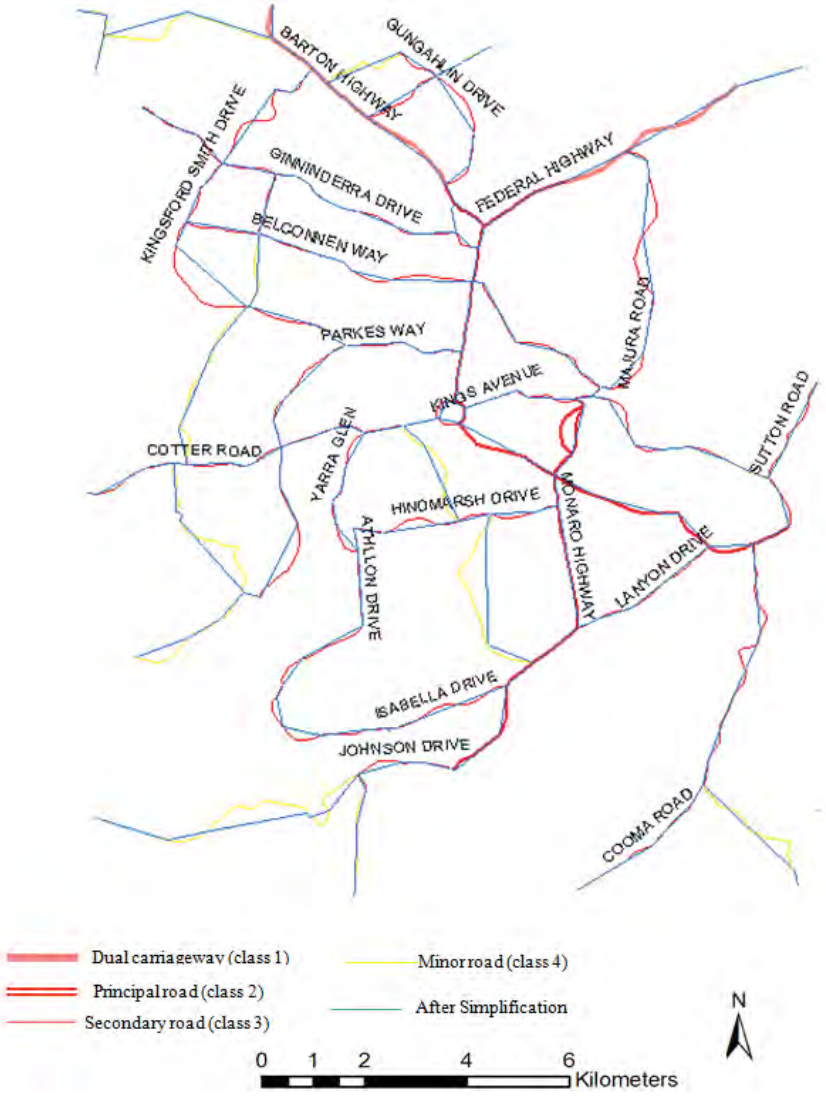


Figure 4.9 The impact of generalisation tolerance of 70m

Qualitative or quantitative assessment of generalisation results is necessary. In this study the derived maps were compared with existing maps. This involved experienced cartographers carrying out a visual assessment/inspection of the results. From the cartographic intuition and perspective, through consultation with experienced GIS operators/cartographers, evaluation by independent assessment helped to define the best levels of tolerance ranging from 10m to 70m that produced satisfactory simplification outputs. The tolerance value (positive number greater than zero) determines the degree

of line simplification. It reflects the distance that two points in a road segment can be apart and still be considered the same to produce generalised data. It requires setting the tolerance equal to or greater than the threshold of separation (the minimum allowable spacing between graphic elements). The threshold value relates to the map coordinate system. The tolerance value is expressed as a number of metres. Various tolerance parameters (values) were tested in the course of road network generalisation.

Based on several tests, it became possible to optimise the process of map derivation by semi-automated cartographic and database generalisation by means of simplification and thresholding to derive small scale maps from a master database. Two different values based on the introductory experiment demonstrated that threshold parameter values of 25m and 35m were the most useful tolerance levels to be applied for the 1:500,000 and 1:1,000,000 maps, respectively.

Similar tolerance levels (20m using Douglas-Peucker and 40m with Reumann-Witkam algorithms) have been reported by Nakos (1999) to generalise a coastline database of Greece. The Reumann-Witkam simplification algorithm uses two parallel lines to describe an area of interest (AOI) after calculating the original slope of the AOI, the line is processed successively until one of the edges of the search corridor intersects the line. **Figures 4.8 and 4.9** illustrate the effects of thresholding versus original data where the impact of the *Bendsimplify* operations with a tolerance of 25m and 35m to produce spatial data products at the 1:500,000 and 1:1,000,000 scale respectively.

In **Figure 4.10** (c), vertices in the original line are shown in green squares. Clearly, applying a higher threshold (threshold 35m) levels resulted in removal of extraneous vertices in the road segments that are shown here. It required manual editing to correct the over simplification, whereas in **Figure 4.10** (a), the simplification (threshold 25m) led to satisfactory results from a cartographic perspective.

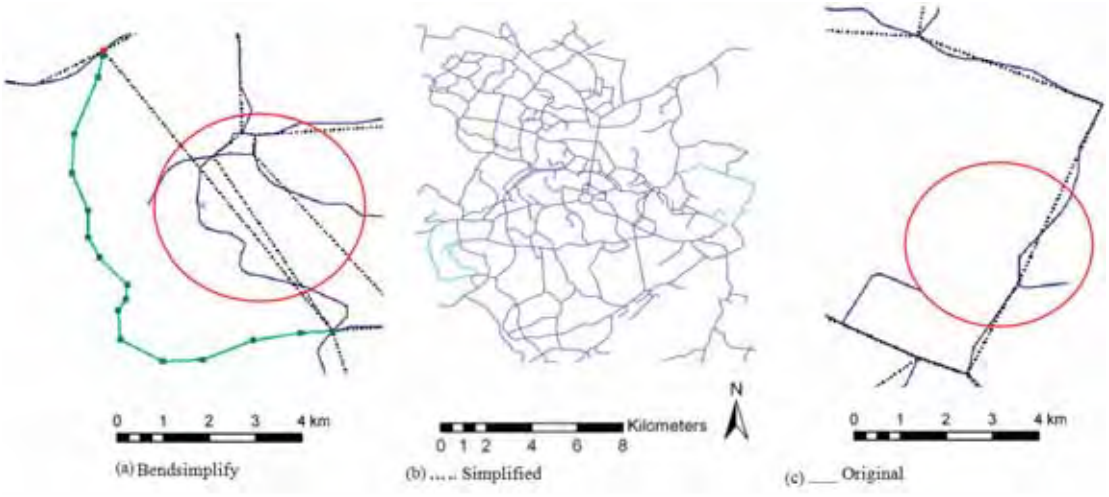


Figure 4.10 The impact of simplification using *Bendsimplify* as seen in a portion of the road database

Figure 4.10(a) shows an over simplification output. In the graph (**Figure 4.10**(a)), vertices in the original line are shown in green squares. Clearly, applying higher threshold (threshold 35m) levels resulted in removal of extraneous vertices in the road segments that are shown here. It required manual editing to correct the over simplification, whereas in **Figure 4.10**(c), the simplification (threshold 25m) led to satisfactory results from a cartographic perspective.

Positional accuracy measures are used to evaluate the quality of road generalisation. Accuracy of simplified and generalised outputs is determined by comparing the positions of 50 well defined ground control points. The point locations are assessed on the generalised maps (1:500,000 and 1:1,000,000) and corresponding positions from the published 1:250,000 national topographic maps. The source map '1:250,000 digital topographic data' has a basic horizontal accuracy of approximately ±120 metres (GA, 2009). Cartographic generalisation of line and area features introduces errors into the derived map. Generalised output maps have been checked from multiple sources including Landsat ETM+ satellite imagery and published 1:250,000 digital topographic data. Positional accuracies for generalised 1:500,000 and 1:1,000,000 maps are approximately 160m and 190m respectively; thus measured errors are within mapping standards. In **Figure 4.11** the results achieved for 1:500,000 and 1:1,000,000 scales are shown.

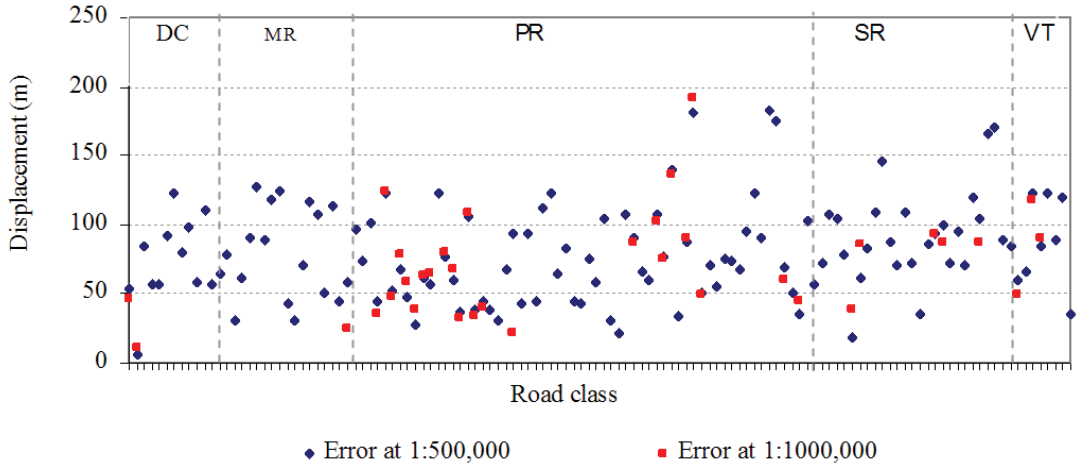


Figure 4.11 Assessment of the average shape changes caused by the simplification operation in relation to the selected control points, showing the displacement values of the coordinate differences on the derived roads map. Blue legend (dot) shows the displacement errors for derived 1:500,000 product from the original data (TOPO250K), whereas the red legend (dot) shows the displacement errors for derived 1:1000,000 product from the original data (1:500,000 scale).

Despite the fact that results from the Generalise™'s DP and *Bendsimplify* are satisfactory, they do not support dynamic generalisation. Therefore there is an opportunity to evaluate other generalisation systems, such as Intergraph's DynaGen™ and Laser-Scan's Clarity™ software, to derive multiscale spatial data products in future investigations. It is worth noting that the importance of expert system application in generalisation operations and cartographers' experience has often been ignored. Kazemi and Lim (2005b) highlighted the importance of integrating cartographers' knowledge with the generalisation system to facilitate the development of a powerful, flexible and robust expert system capable for semi-automated road network generalisation. A comprehensive evaluation of generalisation systems and their performance is essential to marry the cartographic knowledge of experts and bring this into a generalisation framework.

4.4 SUMMARY AND CONCLUSIONS

This chapter presented a generalisation methodology to derive multiscale spatial data through an evaluation of ArcToolbox Generalise™ software. It focused on integration and utilisation of generalisation operators in order to generalise a road network database and produce small scale maps at 1:500,000 and 1:89,000,000 from 1:250,000 national topographic data. The derived maps are satisfactory when compared with the existing small scale road maps such as the Global Map at a scale of 1:1,000,000. It is suggested that a comprehensive evaluation of other generalisation systems and their performance is essential in order to integrate the cartographic knowledge from experts into a generalisation framework.

Generalisation operators in the Generalise™ application were tested to generalise roads by employing the above methodology for derivative mapping. The method was empirically tested with a reference data set at 1:500,000 and 1:1,000,000 scales. According to visual interpretation and quantitative analysis, the results show that the derived maps are satisfactory when compared with the existing small scale road maps such as the *Global Map* at a scale of 1:1,000,000. The derived 1:1,000,000 map was then compared to the 1:1,000,000 *Global Map*. There were no existing 1:500,000 reference data to compare with the derived 1:500,000 map.

As the methodology is only tested on roads it is worthwhile to extend it to other complex cartographic datasets such as drainage networks, power lines and sewerage networks. Additionally, various kinds of linear, areal and point cartographic entities (e.g. coastlines, rivers, vegetation boundaries, administration boundaries, land cover, localities and towers) should also be studied.

To evaluate and validate existing generalisation tools the author’s research focus was on the development of a detailed generalisation framework to derive multiscale spatial data. The work was concerned with the integration and utilisation of generalisation operators as well as cartographer's intuition/skills using the Generalise tool (and possibly DynaGen) software in order to achieve acceptable results.

Previous studies (e.g. Mroz *et al.*, 1996; Nakos, 1997 and 1999; Kazemi and Lim, 2007a; Lee, 2004) have also achieved good results in automatic generalisation; however significant manual work was required since the *Pointremove* generalisation algorithm (Douglas-Peucker, 1973) and the *Bendsimplify* algorithm do not support dynamic generalisation. Other generalisation systems such as Intergraph's DynaGen™ and Laser-Scan's Clarity™ support such generalisation and enable users to derive a multi scale database from a master database (Watson and Smith, 2004; Kazemi *et al.*, 2005a). Therefore, it is suggested that the proposed methodology be enhanced by testing and incorporating tools that will be discussed later.

CHAPTER 5: KNOWLEDGE-BASED GENERALISATION OF ROAD NETWORKS

5.1 INTRODUCTION

The problem of deriving a range of map scales from a common database in today's map production environment is considered a challenging area of research and development (e.g. Arnold and Wright, 2005; Hardy *et al.,* 2004). Technological developments in dynamic cartographic generalisation applications (e.g. mobile mapping, vehicle navigation) call for a robust, practical and cost effective way of fulfilling automated generalisation needs. The trends toward automated generalisation require a powerful set of software tools to replace traditional methods for cartographic map production. A review of the literature on automated generalisation has found a need for intensive experiments with the existing tools developed in the industry rather than the development of new generalisation systems (Lecordix *et al.,* 1997; Kazemi and Lim, 2007a). Therefore this research has focused on the application of the generalisation tools to the development of a detailed generalisation framework for deriving multiscale road data.

Much work has been carried out in the last decade on the development of various generalisation algorithms, but the need to evaluate and validate existing generalisation tools has been overlooked. It was revealed that an assessment of these existing systems will facilitate the development of a detailed generalisation framework for deriving multi scale data and map products from a single high resolution database.

More particularly, the research by Kazemi and Lim (2005b and 2007a) discussed utilising the generalisation operators provided by ESRI ArcGIS™ and Intergraph DynaGen™, combined with the skills of a trained cartographer, to generalise a road network database from 1:250,000 national topographic data and produce smaller scale maps at 1:500,000 and 1:1,000,000. The method was tested (**Chapter 4**) and the results show that the derived maps are satisfactory when compared with the existing small scale road maps such as the Global Map at scale of 1:1,000,000. The *Global Map* Australia 1:1,000,000 (2000) is a digital spatial dataset covering the Australian continent and island territories which consists of eight spatial layers in the vector form

(administrative boundaries, drainage, transportation and elevation as well as several raster forms (elevation, vegetation, land cover and land use). This was created from Geoscience Australia's (GA) 1:250,000 national topographic data.

The generalisation operators of ArcGIS™ have been tested (Kazemi and Lim, 2005a) for the generalisation of roads by employing the conceptual generalisation framework for derivative mapping, but the results showed that ArcGIS™ algorithms of *Douglas-Peucker* and *Bendsimplify* do not support a dynamic generalisation. The way to work around this problem was to use Intergraph™ generalisation software which enables the production of a multiscale database from a master database. As noted in earlier (**Section 2.1**), a multipurpose spatial corporate database offers capabilities to derive different maps at different scales from objects (e.g. topographic objects), at scales ranging from, say, 1:250,000 to 1:1,000,000. This capability is referred to as a 'derivative mapping' capability.

This chapter focuses on the principles of generalisation that were researched using Intergraph DynaGen™ as a testbed. It then compares the results derived from DynaGen™ with earlier work using ArcGIS™ road generalisation. Intergraph's DynaGen™ software is one of the most detailed generalisation software products. The components of this technology together with detailed functionalities were described in **Chapter 3** and **Section 5.2**.

According to existing literature and Intergraph, this generalisation system has been tested for various generalisation tasks in several countries but not in Australia. Iwaniak and Paluszynski (2001) investigated the application of cartographic skills, together with Intergraph MGE Map Generaliser™ software tools, to automate the generalisation of topographic maps of urban areas from 1:10,000 to 1:50,000 scale. The system uses the MGE Map Generalise™ and a rule-based system implemented in C Language Integrated Production System (CLIPS), developed by Purdue University for controlling the generalisation process via a knowledge-based expert system that generates results similar to those obtained with the manual procedure. One of the key features of this system is its efficient handling of conflict resolution among objects. The expert system enables the integration of generalisation operations, generalisation rules and manual

intervention. The authors suggested that this approach warranted further research. Chybicka *et al.,* (2004) applied the same approach to the generalisation of the Polish topographic database at a scale of 1:10,000 to derive 1:50,000 databases.

Several researchers (e.g. Meng, 1997; McMaster and Shea 1989; Mackaness *et al.,* 2007) have highlighted the questions of 'how', 'when' and 'why' to generalise. Building an automated generalisation presents an immense challenge when handling national / global spatial coverage at widely varying levels of detail, as digital and paper products while maintaining currency of spatial information products; it therefore requires practical solutions to the aforementioned questions. Mackaness *et al.,* (2007) noted that recent developments in cartometric analysis techniques are able to support high levels of automation among multiscale derivation techniques within existing and emerging technologies such as mobile mapping. For example, which generalisation algorithm(s) is (are) the most suitable for point generalisation for real time execution, and how it (they) can affect map reading tasks (in terms of efficiency and accuracy) with minimum operator intervention (Bereuter and Weibel, 2010).

In relation to an integrated approach, Yang and Gold (2004) proposed a system which combines database generalisation and object generalisation to overcome generalisation problems (e.g. feature displacement and smoothing) which have been widely reported in the literature (e.g. Ruas, 1998; Stoter *et al*., 2004). They observed that, since current generalisation practices involve human interaction rather than dynamic generalisation, there is a need to develop a generalisation framework for the generalisation of spatial databases. They documented the development of the generalisation of topographic data from a purely manual process to interactive generalisation using LAMPS2™ software and ArcGIS™ for the generalisation of buildings.

It was suggested that future research should concentrate on the development of a robust core data model and evaluation of generalisation systems. Since an effective data model stores special relationships among features, by designing a good data model relationships such as adjacency and connectivity can be established so that generalisation operations (e.g. aggregation) will be more effectively defined through topological relationships between features. In this regard ESRI geo-database technology

provides a framework for objects to maintain geometric attributes, spatial references, relationships, domain and validation rules, topology and custom behaviours (Zeiler, 1999). For instance on a very large scale map roads appear with detailed multiple lane, and at medium range scale lanes are formed as two-way traffic directions, and at small scales appear as a single road. This study will therefore contribute to addressing this issue by combining cartographers' knowledge with different generalisation algorithms through an evaluation of a variety of software systems in order to achieve cartographically acceptable results.

Kazemi *et al.,* (2005b) noted that, for the automation of the map generalisation process it is necessary to integrate generalisation algorithms with cartographer's intuition and skills within a GIS, as this approach leads to more desirable outputs (Joao, 1998). To date there have been no systematic attempts to undertake a comprehensive evaluation of generalisation systems and their performance. This chapter discusses the DynaGen™ generalisation capabilities using road datasets.

In the next section a brief overview of the study area, the Intergraph™ software, and the research dataset is provided. **Section 5.2** describes the developed generalisation methodology using the software cartographic generalisation capability in a 'dynamic mode'. In particular, this section elaborates on the application of different generalisation algorithms used by software and their performance. Then, generalisation performance for the target small scales (1:500,000 and 1:1,000,000) is briefly discussed. Finally, conclusions and recommendations for future research are given.

5.2 METHODOLOGY

A schematic representation of the research methodology is presented in **Figure 5.1**. The motive for this research is to develop a workflow specifically for derivation of multiple scale maps from a master road network database. A large portion of this approach is similar to the previous study by the author in which ArcGIS™ generalisation capabilities were tested. However, all the processes in the flowchart are considered specifically for road generalisation using the DynaGen™ system. The difference between these two, the approaches is in the underlying principles of the two

generalisation systems and the authors' earlier work with ArcGIS™ (Kazemi and Lim, 2005b).

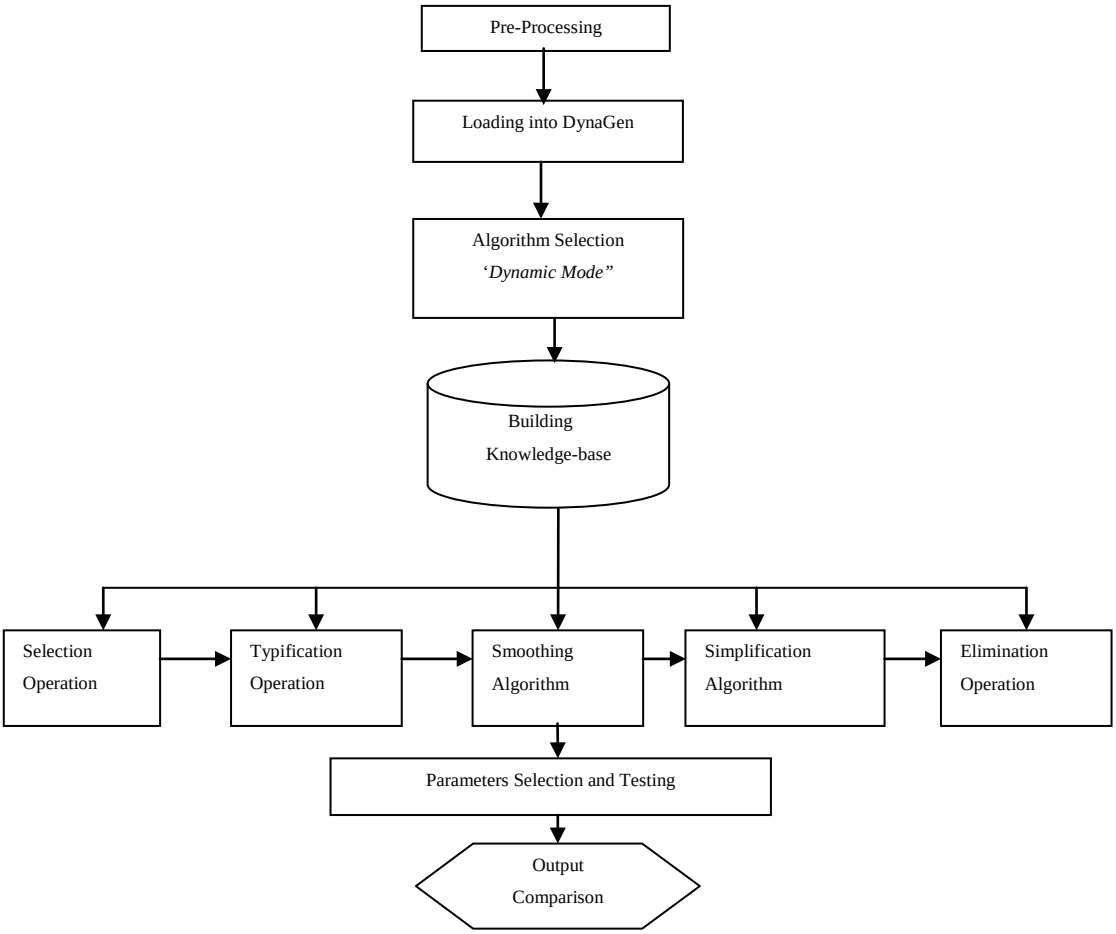


Figure 5.1 Road network generalisation workflow using DynaGen™

5.2.1 System Architecture

The Intergraph GIS system consists of three major components, namely GeoMedia™, GeoMedia WebMap™, and DynaGen™. GeoMedia™ enables appropriate spatial data integration including the capability for incorporating data from disparate sources and multiple formats into a master GIS for viewing, analysis (e.g. query, buffering, overlaying) and presentation. Intergraph's web-based map visualisation and analysis system offers real time access to a data warehouse to publish data on the Web.

DynaGen™ is a subsystem of Intergraph Dynamo™ which allows the use of a graphical environment, and topological functions, and the data models of Dynamo™. DynaGen™ software is a cartographic suite of tools combined with Data Editor (DIDE) for creating, or connecting to, a database for further processing. Generalisation processes in

DynaGen™ can be very detailed; users can take control of the data and monitor the whole process.

Intergraph's DynaGen™ automates the production of multiscale maps from a master database. It operates using two modes of generalisation: the batch mode (automatic) and the cartographer-assisted mode. Methods discussed in this chapter are more related to vertex reduction using line simplification and smoothing. Various algorithms, such as the Douglas-Peucker (1973) simplification (**Section 5.3.3**) and Brophy smoothing algorithm (**Section 5.3.2**), have been tested with a variety of input parameters.

Automated (Batch) Operations - generalisation operators are either based on mathematical formulae or by unequivocal procedure descriptions (algorithm). This transformation is named the generalisation step. DynaGen™ offers a number of generalisation operators, such as simplify, smooth, and feature elimination, that have been applied to road generalisation in this chapter. DynaGen™'s automated generalisation uses a sequence of such transformations through selection and incorporation of parameters in such a way as to maintain certain characteristics and to establish relationships between generalised objects. The cartographer also has access to a number of operators including simplify, smooth, aggregate, change the presentation of objects, extend borders, select representative objects, angle straightening and amalgamation of objects.

Dynamic (Interactive) Mode - DynaGen™ enables 'dynamic' generalisation since the operator is able to change any parameter value using sliders, and visually inspect the dynamically changing generalisation results in real time. This allows the cartographer to fine-tune the process when selecting individual features or a group of features during the generalisation process. The main engine of DynaGen™ is Dynamo, which allows the user to control the changes in the database. This useful capability enables the cartographer to visualise the results even before running the process, which could lead to an overall cost saving compared to the automated mode since the cartographer is able to visually assess, validate, and accept the generalised output 'on the fly' (Iwaniak and Paluszynski, 2001).

5.2.2 Data Pre-Processing

Because data could come from disparate sources and often the user has limited knowledge of the source, its capture method, and accuracy of GIS data in terms of completeness and currency, it is essential to perform some validation and testing of data prior to starting generalisation. Topology validation, geometric feature validation and feature attribute quality checks were performed on the 1:250,000 national topographic roads data used in this research. This resulted in an improvement of quality of the research data. The improvements include fixing attribute and topological (under-shoot and over-shoot) errors. Overshoots occurs when a line (arc) does not end at its termination point on another line (arc) and goes beyond it is called as overshoot. Undershoots occur when a line (arc) finishes before connecting / intersecting to another line (arc) on desired location.

5.3 BUILDING KNOWLEDGE BASE

DynaGen™ validates the generalisation output against the operator's generalisation knowledge, ensuring that changes do not violate topological relationships, and it calculates and updates feature attribute information for changed or recently generalised features (Watson and Smith, 2004). It is used to construct a knowledge base that conforms to two principles; applying generalisation rules executed in the automated mode, and using fundamental generalisation processes executed and monitored by cartographers.

The DynaGen™ knowledge-base uses specific generalisation steps, namely: 1) the name of the generalised object, 2) the operator, 3) the algorithm, 4) the values of the algorithm parameters, 5) the names and values of attributes referring to objects created as a result of the generalisation, 6) a condition implementing a particular approach, and (7) prescribing prohibited topological changes in terms of spatial relations between generalised objects.

The DynaGen™ system incorporates two sets of principles: a) those containing rules executed in an automatic mode and in a restricted sequence, serving as preliminary data preparation, and b) those describing fundamental generalisation processes executed interactively and managed by the cartographer.

5.3.1 Feature Elimination

This function enables the user to define feature class and attribute conditions of features that are not needed and to remove these from the object space. This is especially useful for enforcing map specifications by removing features that do not meet product requirements. For an efficient feature elimination function, the database needs to be made more 'intelligent' by incorporating object-oriented technology. DynaGen™ therefore allows the user to define spatial feature relationships that may impact on the process of feature elimination. An example of the application of this operation is to inspect the Initialisation Data Editor (IDE) tool, allowing the user to build parameter sets containing information on generalisation operators in order to generate parameter files when initiating batch processing and interactive generalisation. For example, road segments which are shorter than a certain length, which cause conflict in the final map and which are not significant for presentation on the map, may be eliminated. For all these features, it would be ideal to show a screen-shot of the pull down menu listing options/parameters that is (are) applied to each of four subsets. However, the DynaGen software is no longer available for this study due to an expired grant licence at the time of revision of this Chapter (March 2011).

5.3.2 Line Smoothing

Smoothing and simplification operations result in a reduction of plotting time, a reduction of storage space, faster vector-to-raster conversion, and faster vector processing. However, there is a difference between simplification and smoothing. Line simplification deals with the representation of the curvature of the line by fewer points, while the smoothing operation deals with representation of the line with fewer sharp angles to improve aesthetics.

Three well known smoothing algorithms (Brophy, Simple Average and Weighted Average) were applied in this study (**Figure 5.2**). These are embedded in DynaGen™. Each algorithm uses a variety of parameters. The Brophy algorithm uses *look ahead factor*, *densification* and *smooth factor*. The Simple Average algorithm uses *look ahead factor* and *densification.* The Weighted Average algorithm shares the principles of two earlier algorithm's parameters, and brings *Weight* into the equation (Intergraph, 2004). Applied parameters include:

a. *Densification*, which temporarily adds vertices along the geometry in order to move the real vertices further, the temporary vertices being deleted once the smoothing process is completed,
b. *Look ahead* which determines how many points ahead of and behind the point that is to be smoothed are used in the smoothing of that point,
c. *Smooth* which determines the distance a point is to be shifted over the radius of the Brophy circle, and
d. *Weight* which determines the weight given to the points during the averaging process.

The smoothing process has been run on a number of road samples from 1:250,000 national topographic dataset (**Figure 5.2**). The effects of the selected smoothing algorithm on that particular segment of the road are highlighted within the red circles.

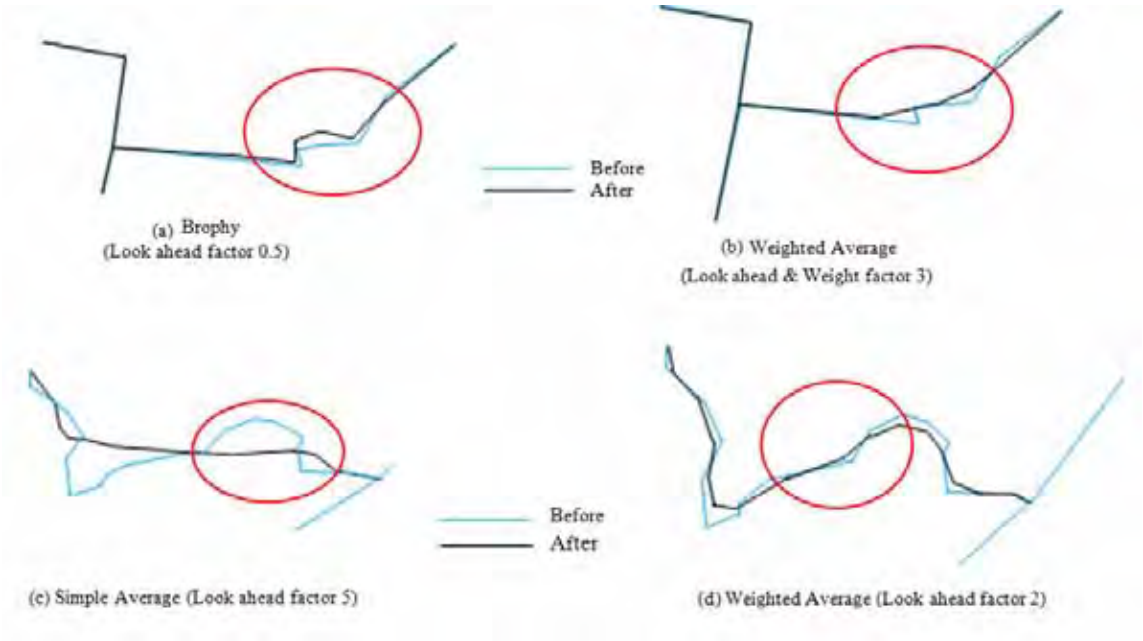


Figure 5.2 Results of smoothing algorithms that have been run on a number of road samples from 1:250,000 national topographic road databases over the study area. Data courtesy of Geoscience Australia © 2004

The road shape is changed during the vertex reduction. The road shape is important when it comes to generalisation due to its effect on coincidence (e.g. roads passing over the top of buildings), features, and accuracy. Producing a readable map and, at the same

time, preserving accuracy is the core task of the science of cartography. It is a trade-off that must be balanced in order to achieve a desirable number of vertices while not causing deterioration of the quality of the remaining linear features. However, to overcome this problem generalisation with an appropriate level of tolerance, suitable settings and appropriate tools is required. This was achieved by several attempts to find an appropriate tolerance setting. It has to be noted that reduction of the vertices will result in low data volume.

Smoothing operators do not change the topology of the feature; all existing topological relationships are maintained without creating new topological relationships. This ensures beginning and ending points of lines are not moved or removed.

5.3.3 Line Simplification

DynaGen™ offers six simplification algorithms which can be used to simplify roads data. The results of the Douglas-Peucker simplification algorithm are presented below (**Figure 5.3**), and the line simplification algorithms and their parameters are tabulated (**Table 5.1**). The minor bends along the road segments are deleted to improve the cartographic representation of the roads.

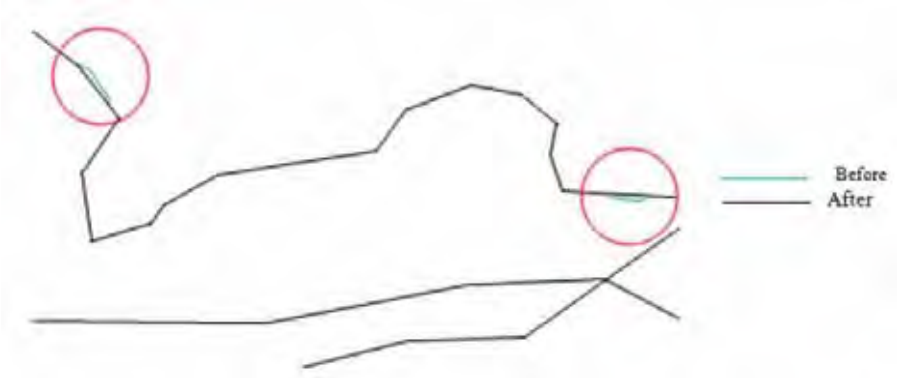


Figure 5.3 The results of Douglas-Peucker simplification algorithm using tolerance of 3m. Data courtesy of Geoscience Australia © 2004

Table 5.1 Line simplification algorithms and their parameters (Intergraph, 2004)

Algorithm	Parameters	Comments
Douglas-Peucker	Z-retention flag and tolerance	Z-retention flag of 1-10m and tolerances of 0.1 to 5 is used.
Lang	Z-retention flag and tolerance	Z-retention flag of 1-10m and tolerances of 0.1 to 5 is used.
Nth Point	Z-retention flag and N	Z-retention flag of 1-10m and point spacing of 1 to 20 is used.
Point Relaxation	Z-retention flag and relaxation radius	Z-retention flag of 1-10m and relax circle radius 0.2-5m is used. But the optimum value is found 3.93m.
Reumann-Witkam (Reumann and Witkam, 1974)	Z-retention flag and tolerance	Z-retention flag of 1-10m and Corridor Tolerance of 0.2-5m is used.
VectGen	Z-retention flag and tolerance	Z-retention flag of 1-10m and tolerances of 0.2-5m is used.

Line simplification demonstrated that the Douglas-Peucker algorithm produces the most desirable generalised results of all the applied simplified algorithms (Alves, 2010). It operates on entire lines. This is because topological and geometrical relations of adjacent objects are appropriately handled when the generalised elements are close to each other, and where there is a possibility of either overlap or intersect. Also, the Point Relaxation simplification algorithm performed better in terms of aesthetic quality of simplified lines than the other remaining algorithms of Lang, Nth Point, Reumann-Witkam, and VectGen. For example, Reumann-Witkam simplification algorithm involves three main characteristics: applying two parallel lines to identify search region; calculating initial slope of the search region; and line processing sequentially until one of the edges of the search corridor intersects a line. These algorithms have been widely discussed in the literature and are not described here (e.g. Visvalingam and Williamson, 1995; and Muller, 1995).

5.3.4 Line Typification

This operation is considered a cartographic generalisation task that makes maps more readable by reducing density and details of lines, points and polygons (Kazemi *et al.*, 2004a) while retaining the character of the original features. In the context of linear feature generalisation it gives a representative pattern of the more significant features in the road network while removing line features based on their proximity to each other.

Two popular line typification algorithms, namely tree levelling and conflict resolution with a number of parameters, were tested in this study. The tree-levelling algorithm uses three parameters: minimum terminal branch length, maximum branch retention level, and identify critical lines. This handled conflict resolution reasonably well as demonstrated in **Figure 5.2.** Similarly, conflict resolution utilises three parameters: *minimum spacing tolerance, proximity limit,* and *identifies critical lines*. Description of these typification parameters follows. *Minimum terminal branch length* specifies the minimum terminal line length. For example, roads shorter than the minimum line length and having no branches are removed. *Maximum branch retention level* retain the maximum number of levels in the tree. If the value is set to 0, all input roads are removed. (c) *Identify critical lines* enables selection of a set of lines as critical lines that are always a part of the representative pattern of lines to retain on the map. *Minimum spacing tolerance* determines the minimum line spacing tolerance. In this regard roads shorter than the minimum spacing tolerance are selected for removal. The algorithm basically reduces the density and simplifies the distribution and pattern of the network. The result should be a connected and less congested network that represents a similar pattern at the smaller scale.

5.4 ASSESSING GENERALISATION PERFORMANCE

The line simplification algorithms change the geometry of a line by eliminating a number of vertices applying tolerance values. This is a factor used to determine the influence of the algorithm on cartographic line simplification processes. Several quantitatively and qualitatively measures have been developed to assess the shape of the simplified lines. Researchers (e.g. McMaster, 1986; Buttenfield 1991; Skopeliti & Tsoulos, 2001) have used a number of mathematical measures (i.e. length, number of line reductions / density, angularity, curvilinearity and density) for evaluating line

simplification. McMaster (1986)'s cartometric measure of total length and number of roads (density) based on feature types is considered here as an index of generalisation to quantify generalisation performance for the target small scale (see **Figure 5.4**). There are 1202 road segments in the source scale map (1:250,000 national topographic data) over the study area. Changes in the representation of road features at 1:250,000, 1:500,000 and 1:1,000,000 scales as a result of generalisation were quantified. Roads outputs from map derivation have been analysed applying the *Radical Law* (Pfer and Pillewizer, 1966) both employing ArcGIS™ and DynaGen™ systems. The *Radical Law* (**Equation 4.1**) determines the retained number of objects for a given scale change and the number of objects of the source map.

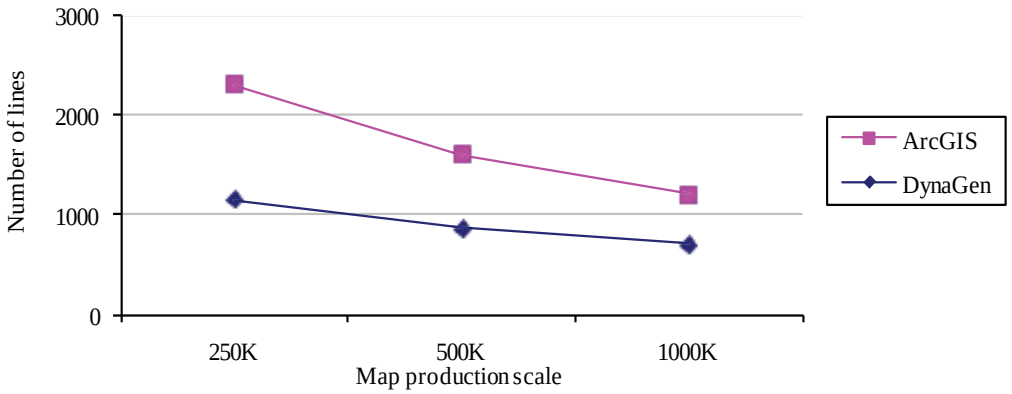


Figure 5.4 Assessment of generalisation results through ArcGIS™ and DynaGen™ systems; the source scale database is 1:250,000 (250K) national topographic data and outputs of generalised maps are road features at 1:500,000 (500K) and 1:1,000,000 (1000K) scales. The variation between ArcGIS™ and DynaGen™ generalisation results could be related to the implementation of simplification algorithms.

The source scale database of 1:250,000 national topographic roads have 1,202 segments and the derived roads have 856 segments after the line simplification and smoothing operations. When applying the *Radical Law* formula, there are 765 lines lines in 1:500,000 scale and 584 lines in 1:1,000,000 scale roads respectively (**Figure 5.5**). This means reduction in the complexity and the density of roads.

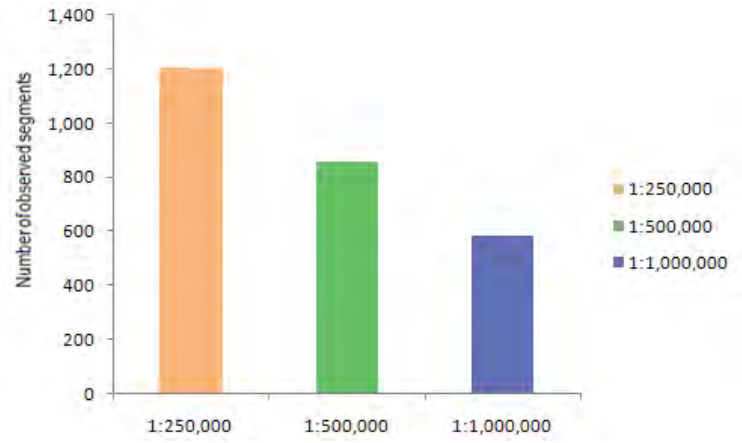


Figure 5.5 Number of observed road segments (lines) for 1:250,000 (the source scale database, TOPO250K), 1:500,000 (outputs of generalised road segments using TOPO250K), and 1:1,000,000 scales (outputs of generalised road segments using 1:500,000).

These results have also been analysed qualitatively through visual assessment/inspection of the results of generalisation that was also carried out by superimposing the versions of roads at scales of 1:250,000, 1:500,000 and 1:1,000,000 successively with ETM+ imagery (composite bands 1, 2, 3) and respective raster maps (**Table 5.2**). The derived maps have reasonable agreements with the existing small-scale road maps such as the Global Map at 1:1,000,000 scale. The outputs from map generalisation have been analysed applying the *Radical Law* and changes in the representation of road features at 1:250,000, 1:500,000 and 1:1,000,000 scales as a result of generalisation were quantified (see **Figure 4.6** and **Figure 5.4).**

In addition, a quantitative evaluation of the results of generalisation is attempted applying positional accuracy measure. Accuracy of simplified and generalised outputs is determined by comparing the worst case and best case scenarios of positional displacement errors. The locations are assessed on the generalised maps (1:500,000 and 1:1,000,000) and corresponding positions from the published 1:250,000 national topographic maps and Landsat 7 ETM+ imagery. The source imagery has a basic horizontal accuracy (RMSE x,y) of approximately ± 15m (Smith *et al.*, 2002) and the '1:250,000 digital topographic data' has a basic horizontal accuracy of approximately ± 120 metres (GA, 2009). Positional accuracies for generalised roads (using 18 points from the Landsat imagery and the 1:250,000 data) are of the order of 140 - 180m RMSE

values respectively; the measured displacements are approximately within the standards.

Table 5.2 A summary of the operations and algorithms available on the employed commercial generalisation platforms (adapted from Mackaness *et al.*, 2007) and their effectiveness (based on the qualitative assessment and experts knowledge). The + and ++ mean low and high respectively.

Operations / Algorithms	**ArcGIS™ Generaliser**	**DynaGen**
Aggregation	Merging +	Line and area aggregation ++
Collapse	Area to Point Line to Point Point to Point Area to Line Area to Edge 2 Lines to Line ++	Area to Point Line to Point Point to Point Area to Line Area to Edge 2 Lines to Line +
Displacement	Editing mode +	YES +
Selection	Selection by Location Selection by Attribute ++	Selection by Location (queries) Selection by Attribute (using Intergraph Geomedia™) +
Simplification	Line and Area +	Line and Area ++
Smoothing	Line and Area +	Line and Area using Brophy & Averaging algorithms ++
Typification	NA	Yes +

5.5 CONCLUSIONS AND RECOMENDATIONS

This chapter introduced a knowledge-based approach to the generalisation of a road network database. This was conducted through application of the principles of generalisation using Intergraph's DynaGen™ software. The derived maps were satisfactory when compared with the existing small scale road maps such as the *Global Map* at scale of 1:1,000,000. The evaluation suggests that such generalisation methodology will be useful for building a practical generalisation framework and workflow. It has been experienced that the generalisation operations/algorithms, and

their parameters embedded in the DynaGen™ system deliver coherent capabilities to automate the generalisation process for practical production applications.

Dynamic generalisation has potential advantages. The following observations should be taken into consideration to maximise its advantages:

- To simplify roads, the Douglas-Peucker simplification algorithm produces satisfactory outputs. Topological and geometrical relations of adjacent objects are well managed when the generalised elements are close to each other and may overlap or intersect.
- For the feature elimination function, it is suggested that the database should be made more 'intelligent' by incorporating object-oriented technology to enable the software to define feature class and attribute conditions of features and enforcing this into map specifications by removing features that do not meet product requirements.
- To increase the efficiency of this approach, cartographic knowledge can be encoded into an expert system as was mentioned here by means of an example. This would be a very fruitful research direction and the implementation of an expert system is considered (discussed) in **Chapter 6** and is implemented in GES (**Chapter 7**).
- To enhance current generalisation practice in national mapping organisations it is important to communicate generalisation problems and requirements, and to evaluate existing generalisation systems (e.g. comparing algorithms and approaches) and measure the fitness of a generalisation approach applying existing software. This recommendation is also supported by Stoter *et al.*, (2004).

It is worth noting that the generalisation operations/algorithms and their parameters embedded in the DynaGen™ system deliver capabilities comparable to other existing systems such as ArcGIS™. The results from the work conducted here show the dynamic generalisation approach has potential for the generalisation of road networks. Building an expert system is recommended in order to integrate generalisation algorithms with cartographers' experience that will assist cartographers in choosing the appropriate techniques for map and database generalisation tasks such as feature displacement.

CHAPTER 6: CARTOGRAPHIC KNOWLEDGE ACQUISITION

6.1 INTRODUCTION

This chapter firstly provides a brief overview of expert systems and their application in Geographic Information Systems (GIS) with a particular emphasis on cartographic knowledge capture. An expert system consists of four main components: (a) knowledge acquisition, (b) inference engine, (c) knowledge representation, and (d) user interface (Forghani, 1997). Expert systems have played an important role in automatic generalisation in different cartographic software applications. These are briefly discussed in **Section 6.2**. The application of expert systems to spatial generalisation systems and in harnessing cartographers' experience has not been investigated thoroughly. This chapter is therefore intended to provide a brief review of expert systems in the context of GIS.

Later, this chapter (**Section 6.3**) describes the process of a heuristic natural knowledge transfer from cartographers for building an artificial intelligence system using an international Cartographic Generalisation Survey. It was conducted at several NMAs, SMAs and a number of software vendors, in order to capture the cartographers' knowledge about the principles of cartographic generalisation and their experience with existing generalisation software. The survey results are utilised for defining the basis for a knowledge-based expert system. Key findings are then formulated into a series of rules as part of the conceptual spatial databases framework.

6.1.1 Relevant Studies

Expert systems have been widely discussed in the literature relevant to knowledge-based research (e.g. Smith, *et al.*, 1987; Walker and Moore, 1988; and Tapiador, 2008). An expert system consists of four key components: (a) knowledge acquisition, (b) inference engine, (c) knowledge representation, and (d) user interface. These are discussed in **Section 6.2** and applied in **Section 6.3**.

An expert system accommodates a large amount of judgmental interpretation and heuristic knowledge or 'rules of thumb' which specify a set of actions to be performed for a given situation. This is done through simulating the element of a human

specialist's knowledge (e.g. cartographers or image interpreters in the application under investigation here), and reasoning that can be formulated into ‘chunks’ of knowledge typified by a set of facts and heuristic rules. In other words, an expert system is a form of ‘communication device’ between an experienced user and the computer program in order to solve cumbersome problems. An expert system tries to reduce cost and time, while increasing (or at least maintaining) accuracy, stability and consistency. Examples of the use of rule-based systems in the mapping and computing field are feature extraction from remotely sensed data, detection of road networks (Wang and Newkirk, 1988; Domenikiotis *et al.*, 1995; Forghani, 1993; Forghani, 1997; Forghani, 2000), or map generalisation (Armstrong, 1991; Weibel *et al.*, 1995).

In reality the use of expert systems is the execution domain of Artificial Intelligence (AI) (Domenikiotis *et al.*, 1995). AI began during the 1940s and 1950s with one of its objectives to make computers more ‘intelligent’ and thus more effective tools. One of the most mature areas of AI research and development is expert systems. Since the early 1950s, the AI community has focused on two main areas of research (and development): cognitive science and search methods (Carrico *et al.*, 1989).

Iwaniak and Paluszynski (2001) attempted to combine the expert knowledge of a human cartographer with Intergraph MGE Map Generaliser software tools to automate the generalisation of topographic maps of urban areas from 1:10,000 to 1:50,000 scale. The system uses the MGE Map Generalise and a rule-based system implemented in the C Language Integrated Production System (CLIPS) developed by Purdue University for controlling the generalisation process through development of a knowledge-based expert system that generates results similar to those obtained employing a manual procedure. The assessment showed that the Intergraph MGE Map Generaliser software system handled conflict resolution among objects efficiently. The authors recommended that the integration of generalisation operations, rules, and manual intervention should be pursued.

Knowledge Based Systems (KBS) assist humans with decision-making by applying probabilistic rules within a knowledge base to specific conditions. This is done through two main approaches in developing expert systems: (a) using a programming language

(e.g. LISP and PROLOG, FORTRAN, C, JAVA), and (b) using expert systems shell tools which require minimal knowledge of any high level programming. Many expert system tools that were originally written in LISP, PROLOG, and FORTRAN which have been coded in C++ and VISUAL BASIC to improve their speed and increase portability (Ball and Moody, 1993). Fundamentally there are three knowledge levels for expert systems: facts, rules and an inference engine. These can be translated into the four main components of expert systems architecture: knowledge acquisition, knowledge representation, inference engine, and user interface (Nikolopoulos, 1997).

The adoption by GIS of machine learning methods (e.g. decision trees and artificial neural networks) and expert systems to build KBS is well established. Discussing in depth the application of these methods to GIS applications is beyond the scope of this study. However, the chapter only attempts to provide a summary of the core parts of an expert system (knowledge acquisition, knowledge representation, and inference engine).

Machine learning deals with the studies and modelling of the learning process in an attempt to develop methods that can automatically construct rules (or other forms of knowledge representation) from a set of examples, in order to make the knowledge acquisition paths of building knowledge-based systems simpler, more productive, more efficient, more flexible and more user-friendly (e.g. McKeown *et al.*, 1985; Mitchell, 1997; Khoshnevis and Parisay 1993).

The decision tree is a method of knowledge extraction in which knowledge is represented as a series of rules for use in construction of expert systems (Hunt, 1962; Quinlan, 1983, 1986 and 1990 and Mitchell, 1997).

Artificial neural networks (ANN) have been used extensively to recreate biological information processing models that are based on the way the brain processes information. ANN is used in many diverse disciplines, including medical diagnosis, electronic signal analysis, medical image analysis, radiology, psychology and mapping science. A generalised model of a neural network comprised of: (a) input ‘layer’ where values are applied to the inputs, changed (based on some mathematical rule) and then accumulated at the nodes; and (b) output ‘layer’ where the inputs are processed and

classified (Michie *et al.*, 1994). In relation to map generalisation, Rdenas *et al.*, (2009) applied the ANN-based data model to represent geographical objects. Two types of neural units, integrated with two types of activation function (hard and soft transition zones) at multi-resolution scales to perform smoothing operation over maps derived from remote sensing data. Almeida *et al.,* (2008) applied empirical models to simulate and predict urban land-use change in real situations in which a supervised back-propagation neural network has been employed. The ANN generated output maps of transition probabilities reflect the influence a set of selected variables has in defining the compatibility extent between the predicted and actual land-use changes.

Cognitive modelling is of interest in the map generalisation context. For the automation of the map generalisation process, it is necessary to integrate cartographers' experience with the generalisation operations within GIS. To automate generalisation of spatial data, a combination of technologies should be used, such as combining expert systems, GIS, and cartographers' experience for a comprehensive evaluation of generalisation systems and their performance. This is necessary in order to pave the way for developing an expert system that will assist cartographers in choosing the appropriate techniques for map and database generalisation tasks such as feature displacement.

6.2 COMPONENTS OF EXPERT SYSTEMS

An expert system consists of four main components: (1) knowledge acquisition, (2) inference engine, (3) knowledge representation, and (4) user interface. These are briefly discussed here. In connection to this, a conceptual expert system for road generalisation is shown in **Figure 6.1**. This architecture forms part of the generalisation framework for semi-automated road network generalisation.

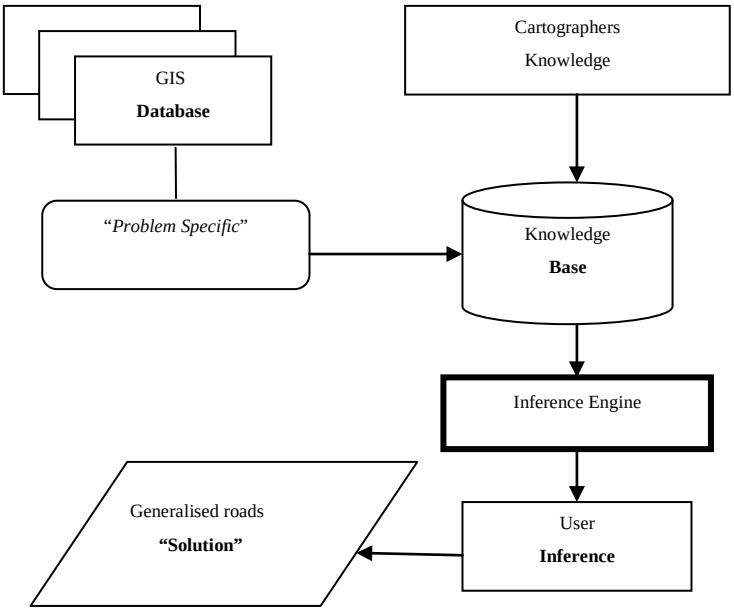


Figure 6.1 Conceptual expert systems architecture for road generalisation

6.2.1 Knowledge Acquisition

Radke (1995 and 2007) noted that a key component of GIS is information which is translated into knowledge acquired from evidence and data. These evidences (facts) and data are characteristics abstracted from phenomena under investigation. By collecting, assembling and integrating these data, the GIS analyst derives knowledge and intelligence about the phenomena being studied. The knowledge acquisition module of an expert system converts storage (or collection) of facts from knowledge sources (e.g. textbooks, manuals, case studies, cartographers, technical publications, learning from examples, discovery, etc.) to software systems. This could include extraction of rules for problem solving from an expert (i.e. cartographer), and make it adequate for machine intelligence processing in order to perform complex search strategies and apply inference engines (Mason, 1995).

The process of transferring knowledge from a cartographer to map generalisation software is not a natural knowledge transfer process, however transferring knowledge from lecturers to students or from parents to children, is a natural process. Several phases are noted in the knowledge acquisition stream, such as knowledge elicitation, knowledge extraction from the knowledge sources, knowledge encoding into symbolic form, knowledge-based organising, and modifying to gain the best performance (Marshall, 1990). In the context of digital cartographic generalisation it involves the

examination and interpretation of manual generalisation processes. Translating cartographer's thoughts into a set of explicit and well-defined processes is a major challenge (Lee, 1992).

Knowledge extraction approaches are sometimes termed 'methods of inference'. Giarrantano and Riley (1989) suggested that a knowledge extraction approach can be categorised into deduction, induction, intuition, heuristic, generate and test, abduction, default, auto-epistemic, non-monotonic, and analogy. This research will mainly build knowledge through interviews and surveys of cartographers to determine the basis on which the generalisation decisions can be made. This is the framework of knowledge engineering. It attempts to acquire from the cartographers all the elements of heuristic experience.

A major challenge in this process is the intensity of the knowledge that is not well organised. Therefore a bottleneck in developing such expert systems is to compile it into a machine-readable format. This 'knowledge acquisition bottleneck' in the field of cartographic knowledge acquisition is noted in Weibel *et al.,* (1995). However, in Mustiere *et al.,* (2000) this problem is overcome by analysing different types of knowledge involved in the cartographic generalisation process. It is necessary to gauge the depth of knowledge, find the right amount of knowledge and accomplish map generalisation by adding some learning abilities to the software and database system. It is noted that cartographic rules are numerous, contradictory and often not formalised. This is because the knowledge 'mixture' is not easy to maintain and limits the comprehensibility of the reasoning done by the system. In Meng (1997), inadequate knowledge formalisation was identified as a problem in successful implementation of generalisation in geographic databases.

Other researchers (Visvalingam and Herbert, 1999) have suggested that evaluation of computer-aided generalisation versus manual generalisation is required. Often generalisation algorithms ignore the role of cognitive issues through knowledge discovery techniques such as decision trees, fuzzy logic, data mining and neural networks to extract the hidden knowledge. Once knowledge is discovered, it can be represented in a suitable form to build an expert system. Examples of generalisation

rules are: (1) contours never intersect; (2) water bodies are located in the bottom of valleys; (3) roads can cross each other; and (d) symbology as well as colouring of land-use data is best displayed with colour-hue for visual perspective. These are used as general guidelines when a cartographer makes maps. But cognitive methods introduce more specific knowledge that is ignored in standard generalisation algorithms. Establishing cartographic rules dynamically provides a possible solution to automated generalisation. Different tasks have different rules and require a different knowledge base. Therefore creating a distributed knowledge base with respect to symbolisation, colour schemes, layout, object displacement, etc., is an essential part of this process.

Examples of generic rules which will be incorporated into the GES in **Chapter 7** are:

- If road connectivity is greater than a given threshold, then keep it; otherwise remove it.
- Removing features that are too short to be represented at the smaller scale.
- Minor roads that are shorter than a given length, and do not reach a built-up area can be deleted.
- Selection of roads must be according to their hierarchy.
- Roads should be removed only if it enhances the overall legibility and connectivity of the map.
- Roads within a network should be large enough to be clearly visible.

This study has evaluated generalisation tools and their functions in order to develop workflows and procedures for generalisation of geographical features such as road networks. The results need to be compared with maps of similar scales. Current generalisation systems, such as Intergraph's DynaGen™, formalised the learning process between cartographer and the generalisation function through an interactive mode. The data analysis (e.g. cluster analysis) and decision-making (e.g. the identification of critical points) are done visually. The drawback of this approach is the subjective nature of the generalisation. Manual generalisation operations are implemented in generalisation systems as functions. An example of a workflow for road data generalisation is: (a) *elimination* – very short branches and unimportant roads to be eliminated, (b) *simplification* – the complexity and the amount of data representing the

roads to be reduced, and (c) *smoothing* – the simplified roads to be smoothed to improve visual impression of the output.

6.2.2 Knowledge Representation

Knowledge representation brings collected knowledge into a suitable form such as decision trees (Forghani, 2000). For example, the KnowledgeSEEKER algorithm (De Ville, 1990) produces the knowledge from example data and represents it in the form of decision trees, and offers the capability to convert it into both generic rules and programming statements. Rules are formulated as If-Then or If-And-Then statements into a knowledge base containing separate Conditions using Boolean operators (And, Or, inequalities such as >, <, =) and Actions. The Action segment of the rule is: If the condition is satisfied, then the relevant rule is executed. Rules consist of premise-action pairs, for example:

If R_1 *&* ... *&* R_n, *Then* C_1 *&* ... *&* C_n.

Which reads: *If* premises R_1 and ... and R_n are true, then actions C_1 ... C_n, are taken, where R_i and C_i are *'conditions'* and *'conclusions'*, respectively.

In cartographic generalisation knowledge representation mainly deals with symbolising geographic objects and is guided by the abstracted object such as how to represent a road so that it is legible (Mustiere *et al.,* 2000). The research is more interested in intuition and heuristic approaches in generalising roads, and will be discussed in future work.

6.2.3 Inference Methods

A rule-based expert system requires control architecture to decide which rule would need to be applied first, or next, and which rules should be combined. As rules get more complex, computers face increasing difficulty in decision-making. To overcome this problem, forward and backward chaining (Watson, 1997) can be applied. The inference mechanism in the context of its use in spatial context problem solving has been investigated in a number of studies (e.g. Domenikiotis *et al.,* 1995; Nishijima and Watanabe, 1997; Forghani 1997).

Forward chaining (data-driven) attempts to reach a conclusion through bottom-up reasoning where reasoning starts as the original state of problems from the evidence and facts, to the top-level conclusions that are based on facts. Bielawski and Lewand (1988) noted that the inference method does not compare the information in the goal database with the *Then* part of a rule in the knowledge base, but rather with its *If* statement. If all the conditions are satisfied, then the conclusion is reached (Giarrantano and Riley, 1989).

Backward chaining (goal-driven) starts processing the data and associated rules from the hypotheses (top-down inference) to the lower level facts; in that it supports the choices and conclusions. This begins with a conclusion and proves the conclusion by providing the truth of each premise in a left to right, or top to bottom order. In contrast to forward chaining, the operator begins by assuming a conclusion to be true and then applies the rules to prove it (Giarrantano and Riley, 1989).

6.2.4 Interface

An expert system consists of a number of major system components and interface performing a range of functions. An expert system must offer a graphical interface so that even an inexpert user should be able to express the ideas, explanations, update and check the knowledge base when running the system in the absence of an operator (Boss, 1991). User interface can influence the applicability of an expert system. User-friendlier interfaces make a reasonable use of menus, and windows, e.g. setting threshold parameters for line generalisation, etc.

6.3 APPLIED COGNITIVE KNOWLEDGE ACQUISITION METHOD

AI has played an important role in spatial data management and map production processes across GIS applications. Studies by the author show that knowledge acquisition within existing generalisation systems, e.g. generalisation of line and area features, has not been fully implemented (Kazemi *et al.*, 2004a). Also many of the implemented generalisation algorithms are generic and are unable to offer a total solution for the operational environment. The motivation of this research is to develop a workflow for derivation of multiple scale maps from a master database. For the

automation of the map generalisation process, it is therefore necessary to integrate cartographers' experience and intuition with the generalisation operations within GIS.

Knowledge discovery has led to the amassing of very large repositories of customer, operations, scientific, and other types of data using a number of techniques such as predictive modelling (Provost and Kolluri, 1999). Survey research in general aims to collect information from representative samples of the total population of survey targets. Then the information gathered from the survey sample is used to make a generalisation about the view of the total population with the limitation of random errors. Two major criticisms are regularly made in the literature when discussing surveys of this nature, one is sampling errors due to the sample size, and the other is responses vs. non-response bias (Wunsch, 1986). It is essential that these two issues be addressed when designing a quantitative survey. A key benefit of quantitative survey research is the ability to use small samples to make inferences about the larger population that would be prohibitively expensive to study (Holton and Burnett, 1997). The question is, how large a sample is required to make valid inferences about the target population?

Statistical measurements are often used to determine the correct sample size for a survey, being one of four inter-related features of study design that can affect the discovery of significant differences, relationships or interactions (Peers, 1996). Bartlett *et al.,* (2001) noted that survey design often attempts to minimise both an 'alpha error' (identifying a dissimilarity that does not actually exist in the population), and a 'beta error' (failing to find a difference that appears in the population). However, need to address problems associated with the survey process, such as no responses, no comments/opinions, missing data, and small size when the target population size is not large.

The cartographic knowledge acquisition is undertaken through an International Cartographic Generalisation Survey in order to build a rule-based expert system. The next step will be to develop generalisation workflow to make generalisation as efficient as possible. The procedures should also highlight both essential and desirable steps for generating smaller scale maps in line with the production environment. These include

topological relations between the object types and classes, how the objects have to be selected, how to generalise, when to smooth, when to delete, when to merge, how to reclassify roads, and so on. It should be noted that such developments and improvements will not be possible by the efforts of a single university, map producer, GIS software vendor or national mapping agency.

The next sections (**6.4** to **6.6**) present the process of a heuristic natural knowledge transfer from cartographers in order to build an AI system using an International Cartographic Generalisation Survey. A survey of cartographic generalisation practices was conducted from November 2005 to May 2006 at several NMAs, SMAs and a number of software vendors. The survey was designed to collect experts' recommendations in relation to new technologies and future generalisation research that could be undertaken by universities and the spatial information industry. The survey questionnaire (knowledge acquisition questionnaire) is presented in **Appendix 1.** The survey results were utilised to build a knowledge-based expert system as explained in **Chapter 7**. Cartographers' feedback was provided in the form of broad qualitative statements, and was analysed to obtain the most pertinent comments. Statistical responses were assessed in quantitative terms. This chapter presents key findings from an analysis of the survey results that were subsequently incorporated into GES software.

6.4 SAMPLING TECHNIQUES

A robust sampling method reduces the noise in the target population and in turn generates more sensible results (Laffan, 2002). In this regard sample designs for a probabilistic approach include random, systematic, stratified, and cluster samples (Yates, 1981). In random sampling, each element has the same probability of selection and every combination of elements has the same probability of selection to ensure that the sample is representative. In fact all members of the population have an equal chance of being selected. The surveyor can use random number tables or statistical software tools to generate random numbers. In systematic sampling, however, each element has the same probability of selection, but not every combination can be selected (Foot-Retzer, 2003).

When applying the stratified sampling method one must ensure that each sample represents some subgroup of the target population, e.g. female employees by age in the workforce, executive personnel by race, and so on. Finally, in the cluster sampling technique, which is generally used in face-to-face surveys, the target population is divided into clusters. The major advantages of this sampling method are that it is inexpensive, no standard sampling framework is required, and the sample size is *not* dependent on population size. Cluster sampling has been applied in a wide range of fields, from engineering (machine learning, artificial intelligence, pattern recognition, mechanical engineering, electrical engineering), computer sciences (web mining, spatial database analysis, textual document collection, image classification and segmentation), life and medical sciences (genetics, biology, microbiology, palaeontology, psychiatry, pathology), to earth sciences (geography. geology, remote sensing), social sciences (sociology, psychology, archaeology, education), and economics (marketing, business) (Hartigan, 1975; Everitt *et al.*, 2001).

To manage, store and analyse a large amount of information an effective way of dealing with the data is to classify or group them into a set of categories or clusters. This can be done either by supervised or unsupervised processes, depending on whether new inputs are to be assigned to one of a finite number of discrete supervised classes or unsupervised categories (Xu and Wunsch, 2005). The aim of clustering is to separate a large number of unlabelled data into a finite and discrete set of 'natural', hidden data structures, rather than to provide an accurate characterisation of unobserved samples generated from the same probability distribution (Cherkassky and Mulier, 1998). Clustering algorithms partition data into a certain number of clusters such as groups, subsets, and categories, although there is no universally agreed definition (Everitt *et al.*, 2001). The discussion of algorithmic clustering methods is beyond the scope of this chapter as classification of the target population was straightforward in this survey. Interested readers are referred to statistical textbooks for a detailed discussion (e.g. Everitt *et al.*, 2001).

The cluster sampling is combined with judgment sampling that selects the training data based on judgment, which is convenient for drawing conclusions from the entire target population. In judgment sampling the surveyor uses his/her judgment in selecting the units from the population for study based on the population’s parameters. This sampling

technique could be the most appropriate if the population to be studied is difficult to locate, or if some members are thought to be better (more knowledgeable, more willing, etc.) than others to interview. This determination is often made on the advice and with the assistance of the client. The target population refers to groups of clients for whom this survey is designed to survey.

6.4.1 Sample Size Selection

A sample is a small subset of observations selected to characterise/generalise the findings about the entire population of interest. Therefore all members of the target population must have a known chance of being included in the sample. This makes the survey less expensive and less time-consuming, whilst allowing accurate statistical inferences to be made about the entire population. Samples can be probabilistic or non-probabilistic. A probabilistic sample generalises the entire population and leads to unbiased results, while a non-probabilistic sample takes a more exploratory approach, which is often more convenient (Foot-Retzer, 2003).

The determination of the appropriate sample size is an important task for conducting research surveys. Taking inappropriate, inadequate, or excessive sample sizes adversely influences the quality and accuracy of research. The survey of this study was constrained by the number of NMAs that participated in the survey (due to its nature). This chapter describes the procedures used to conduct a cartographic generalisation survey based on a sample size for categorical variables using Cochran's (1977) formula:

$$n_1 = \frac{n_0}{\left(1 + n_0 / m\right)} \qquad \textbf{(6.1)}$$

where n_0 *is* required return sample size according to Cochran's formula, and n_1 is required returned sample size because sample > 5% of population m .

A table developed by Bartlett *et al.*, (2001) was used as a guide to select the sample size for this research based on three alpha levels and a set error rate. In addition their procedures for determining the appropriate sample size for multiple regression and factor analysis, common issues in sample size determination, and non-respondent sampling matters were taken into consideration.

Krejcie and Morgan (1970) developed an equation to determine sample size for categorical data, but it is only applicable to a target population size of less than 121. Hence it is inappropriate for the survey as the target population is around 174. Cochran's (1977) method for determining sample size was used to specify margins of error for the substance. For error estimation, it exploits three key parameters: (i) the risk the surveyor is willing to accept in the research, generally called the margin of error, (ii) the 'alpha level', which is the level of risk the surveyor is willing to accept that differences identified by statistical analyses actually do not exist, and (iii) the 'beta error' which is when statistical procedures result in a judgment that no momentous differences exist when, in fact, they do.

A correct *t* value must be used when the survey involves smaller populations. For example, for an alpha level of 0.05 and a population of 60 a *t* value of 2 is appropriate. Generally, an alpha level of 0.05 is satisfactory for most survey research. An alpha level of 0.10 or more may be attractive if the surveyor is more concerned with determining insignificant associations, differences or other statistical phenomena as a precursor to further research. An alpha level of 0.01 may be used in situations where decisions based on the survey are critical and errors may cause substantial costs (Bartlett *et al.*, 2001).

The surveyor does not have much direct control over variance but should incorporate variance estimates into the survey design. Cochran (1977) summarised four methods of estimating population variances when determining sample size. These include: (a) the use of a step-by-step approach to sampling, using the results of the first step to determine how many additional responses are required to obtain an appropriate sample size based on the inconsistencies observed in the first step data; (b) the application of pilot study results; (c) the utilisation of data from previous research of the same or a similar population; and (d) the estimation of the structure of the population assisted by some reasonable mathematical results. The first three are logical and produce valid estimates of disagreement. An acceptable margin of error for categorical data is 5% and for continuous data it is 3% (Krejcie and Morgan, 1970). As the target population is categorical, this method will be discussed in detail.

Cochran's sample size formula and procedures were used here. For this particular survey, the alpha level is assigned a value of 0.05 with an acceptable error of 5% and a standard deviation of 0.5. In statistics, survey sampling is a random selection of a sample from a finite population. It is an important part of planning statistical research and design of experiments. Sophisticated sampling techniques that are both economical and scientifically reliable have been developed (Bartlett *et al*., 2001). Equation **(6.2)** computes:

$$n_0 = \frac{t^2 pq}{d^2} \qquad \textbf{(6.2)}$$

where *t* is the value for the selected alpha level of 0.025 in each tail = 1.96 (the alpha level of 0.05 demonstrates the level of risk the surveyor is willing to take that the true margin of error may exceed the acceptable margin of error); *pq* is the estimate of variance = 0.25 (maximum possible proportion (0.5) * 1- maximum possible proportion (0.5) which generates maximum possible sample size) where *p* is the estimated proportion of an attribute that is present in the population; *q* is 1-*p*; and *d* is the acceptable margin of error for proportion being estimated = 0.05 (error the surveyor is willing to except).

Thus, for a population of 174 the recommended sample size is 120. However, in this study, the sample size exceeds 5% of the population (174*0.05 = 8).

Cochran's (1977) correction formula was applied to calculate the final sample size. The calculations are shown below (Bartlett *et al.*, 2001):

$$n_1 = \frac{n_0}{\left(1 + n_0 / m\right)} \qquad \textbf{(6.3)}$$

where the population size is 174; n_0 is the required return sample size according to Cochran's formula= 384; and n_1 is the required returned sample size because sample > 5% of population m.

These measures result in a minimum returned sample size of 120. The *response rate* describes the extent to which the final data set includes all sample members (observations). It is the number of respondents divided by the total number of target organisations in the entire population, including those who refused to participate and

those who were not available (Smith, 1983). Conclusions were drawn using a target population of all agencies and software vendors who were a part of the process. These calculations are based on the following factors:

Anticipated return rate is 25%, n_2 is sample size adjusted for response rate; minimum sample size (corrected) is 120. Therefore, n_2 is 120/0.25 which equals 480.

The power analysis ideally requires 120 responses. The United Nations had 192 members in 2008 and the US State Department recognised 194 independent countries around the world (Worldatlas, 2008). Not all these countries' national mapping agencies were easy to access. During development of the cartographic survey questionnaire, the Australia's national mapping agency was approached when compiling the list of mapping agencies (state, national and international) and commercial spatial data producers. These entities are directly or indirectly involved in cartographic or digital map generalisation (Holland, 2005). It became apparent that there were at least 174 agencies / companies around the world directly or indirectly involved in cartographic or digital map generalisation. In this context the target population is therefore considered to be 174. Some agencies still using traditional cartographic methods for generalisation of spatial data and maps were selected, as well as those that employ a modern generalisation environment. Expressions of interest were received from 75 agencies and software vendors to participate in this survey. A total of 26 surveys were completed by 26 cartographers. The spatial distribution of participant agencies is shown in **Figure 6.2** and **Table 6.2**.

A number of authors (e.g. Donald, 1967; Hagbert, 1968; Johnson; 1959; Miller and Smith, 1983) have discussed the issue of sampling non-respondents. They suggested that the surveyors might take a random sample of 10-20% of non-respondents for use in non-respondent follow-up analyses. If non-respondents are considered as a potentially different population, it does not appear that this recommendation is valid or adequate. Instead, the surveyor could use Cochran's formula to determine an adequate sample of non-respondents for the non-respondent follow-up analyses.

Figure 6.2 Locations of survey respondents

Prior to collecting data on the selected observations procedures on how the information will be captured need to be developed. In sample surveys of cartographic knowledge acquisition, the procedures may be the construction of a questionnaire in the form of telephone interviews, face-to-face interviews, or e-mail/postal surveys. How questions are phrased, the order in which they are presented, the time it takes to complete the questionnaire or interview, all influence how people answer. For example, two different versions of the same question can lead to different answers from an individual, potentially invalidating the survey. How data is collected also impacts the interpretation of the survey results. An explicit, precise protocol is needed for each type of data to be collected. The protocol may specify what type of technical questions to use, how to frame the question, the type and length of questions, as well as many other items. The determination of procedures is termed the 'response design'. The goal of the response design is to ensure the collection of consistent information for all sampling units.

Those surveys received from the same mapping agencies and software vendors were merged into a single response in order to reflect the overall cartographic knowledge within the organisation. Cartographers from three major streams, including NMAs, SMAs and private industry (software vendors), would be responsible for completing the survey. The Cartographic Generalisation questionnaires were e-mailed to respective target population samples during the period November 2005 to May 2006, and recipients were asked to respond within 30 days. Reminders were e-mailed to all survey recipients that their completed questionnaire was due within the next 15 days. A second

attempt was made to remind those who delayed sending the completed survey, in order to minimise the number of non-responding organisations. Out of 75 agencies who were contacted, only 26 cartographers completed the survey from 15 agencies. This response rate was attributed to the fact that the information gathering was conducted by e-mail survey due to a lack of funding and resources for face-to-face surveys / interviews. There were only two face-to-face surveys. The author had access to National Mapping in Australia and also visited Iran's National Cartographic Centre while attending a conference in that country. The same constraints also dictated that the sample sizes were smaller than statistical theory would suggest.

6.5 SURVEY INSTRUMENTS

This section details the process of conducting the survey (**Figure 6.3**). The key participants have been identified; the survey design completed and survey testing undertaken. During each of these steps the best survey methodology, and the advantages and disadvantages of each method were considered (**Table 6.1**). The survey instrument (questions) is detailed in **Appendix 1**.

Survey Participants: Two types of mapping agencies were targeted; those using traditional cartographic methods for generalisation of spatial data and maps, and those operating in a modern generalisation environment (**Table 6.2**).

Table 6.1 Summary e-mail/ postal questionnaires *vs.* face-to-face interview (Zikmund, 2000)

E-mail / postal Questionnaires	Face-to-face Interviews
Enable respondents to collect facts of relevant methods used and effectiveness	Difficult to collect detailed statistics
Better chance that respondents will take more time when thinking about their replies – necessary for these open-ended big picture strategic survey	Limited duration of interview (< 40-60 minutes) More interactive than e-mail Able to establish closer relations with respondents
The degree to which the interviewer can influence the answers is absent	Interviewer inconsistency can affect results
Respondents can answer questions at own convenience	Limited response convenience
Time to reflect on answers that are critical in this case.	Limited time to reflect on responses
Provide advance notification, cover letter, follow-ups and incentive	Better visual medium
Low cost and convenient for both parties	High cost
Reduced speed of data collection	High speed of data collection
Respondents can misinterpret Questions	Respondents can seek clarification immediately
Follow up would increase return rate	Return rate high

Table 6.2 Mapping agencies and software vendors participating in the survey research

Survey Participants	Sector
Kort-og Matrikelstyrelsen (KMS) Denmark	National Mapping
Institute Géographique National France	National Mapping
Geospatial and Earth Monitoring Division, Geoscience Australia	National Mapping
Land Information New Zealand	National Mapping
Iranian Cartographic Centre	National Mapping
BAE Mapping Australia	Private (map producer)
Department of Sustainability and Environment Victoria	State Mapping Agency
Land & Property Information NSW	State Mapping Agency
Department of Environment and Resource Management QLD	State Mapping Agency
National Cartographic Centre of Iran	National Mapping
Ordnance Survey of UK	National Mapping
Intergraph USA	Private (software vendor)
Laser Scan UK	Private (software vendor)
Sweden Lantmäteriet	National Mapping
ESRI USA	Private (software vendor)

Selection of Survey Method: The face-to-face and e-mail survey methods were chosen in order to:

a) Improve the response with interviews.
b) Increase the response by acknowledging their feedback in publications.
c) Use bulk e-mailing to keep the cost down.

Beta Test Survey: The survey was pre-tested before sending it to survey participants, permitting the identification of any problems with the survey, and facilitating revision as necessary. The beta test survey was reviewed by 6 independent subjects. Their comments related to the content, flow, logic and structure. Some modifications were undertaken, resulting in an improved version of the questionnaire.

Initial Contact: Selected target agencies were initially contacted by e-mail to advise them of the survey and its purpose. The survey asked potential respondents if they were willing to participate in the survey, and assuring them that the responses would be treated in confidence. The survey was generally conducted by e-mail, but the opportunity was taken to carry out face-to-face interviews at two NMAs that were easily accessible. The potential respondents were advised of the timeframe for completion of the survey. Respondents were advised that the survey would be sent electronically, and it was asked if they could complete and return the survey within the designated time.

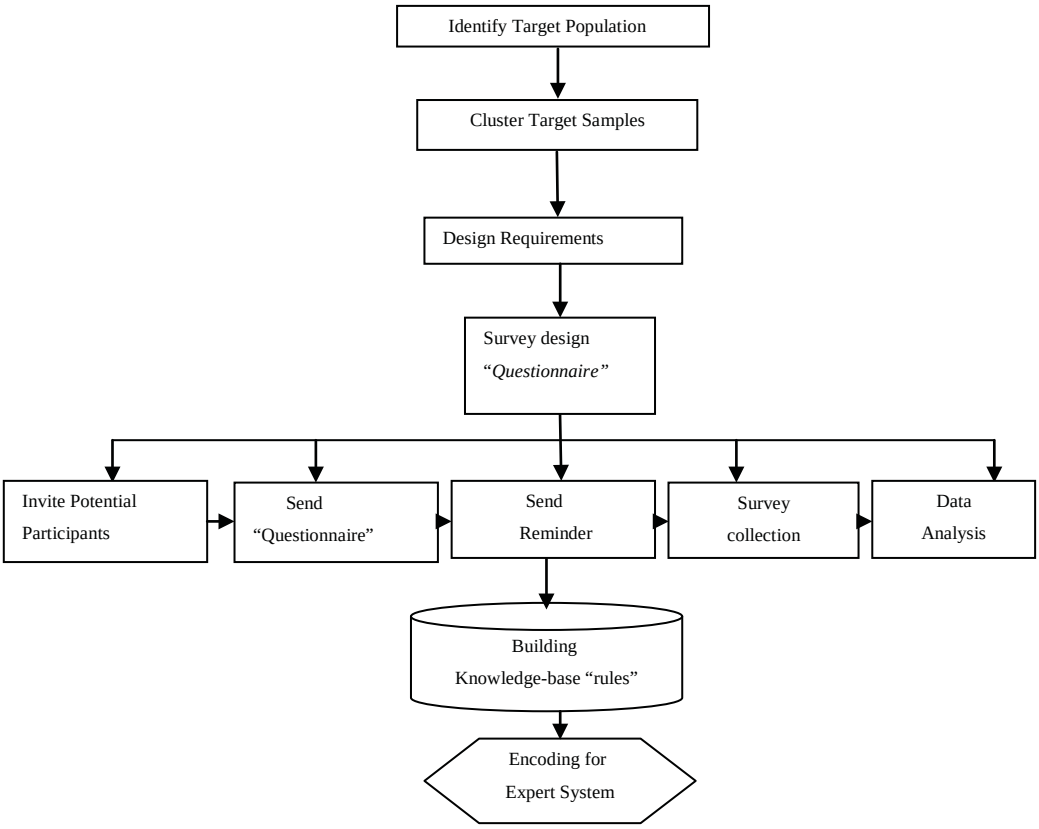


Figure 6.3 The process of conducting the cartographic generalisation survey

The survey questionnaire was primarily completed by mapping operators (cartographers) but not by their supervisors. This is because the operators possess detailed knowledge of cartographic mapping and generalisation functionalities, etc.

6.6 DATA ANALYSIS

Among the four sampling techniques mentioned in this chapter, the focus was on cluster sampling because it best fitted the purpose of the survey. The diversity of cluster analysis options may cause confusion. Xu and Wunsch (2005) provided a detailed review of different clustering algorithms for data sets appearing in statistics, computer science, and machine learning, and illustrated their applications in some benchmark data sets, the travelling salesman problem and bioinformatics, a new field attracting intensive efforts. Once the target population had been selected and data obtained, a statistical summary was needed in order to provide information to meet the survey objectives. For this cartographic knowledge capture work the summary may be as simple as the percentage of cartographers who completed the survey. How the percentage is calculated depends on the survey design used to collect the data. When the survey

design is a cluster sample, the target population is subdivided into three major streams: NMA, SMA, and private industry (software vendors).

Respondent's Subset: Cartographers' feedback was largely provided in the form of broad qualitative statements and was analysed to extract the most pertinent comments. Statistical responses were assessed in quantitative terms. The six themes that emerged will be discussed below. Of the 75 agencies that expressed interest in participating in the survey, only 26 responses were received from 15 agencies, representing a 20% response rate.

The majority of respondents were from the NMAs and SMAs. The results indicated that a cross-section of participants' categories was represented (**Figure 6.4**). However, when the NMAs and SMAs categories were collapsed into one subset they were somewhat over-represented compared to the private industry subset. This was because a significant proportion of participants from the latter group declined to participate in the survey. Three incomplete surveys were received from state mapping agencies (QLD, VIC and NSW). The survey analysis results are graphically presented in **Figure 6.4-6.13.**

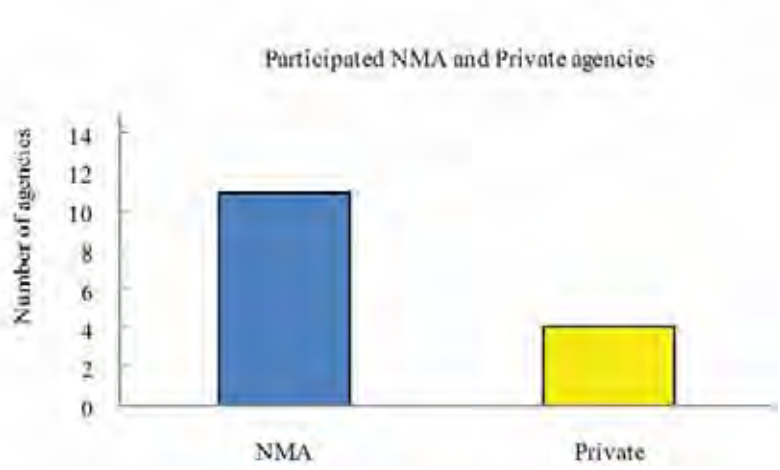


Figure 6.4 Respondents category

Examples of suggested cartographic rules by participants include:

- Roads can intersect. Short segments less than 1250m with a 'dangle' cannot exist. Priority is given to road hierarchy such as retaining principal roads. Tracks generalised for deriving 1:500,000 spatial data from the source scale at

1:250,000. The survey responses indicate the important role played by cognition, or so-called common sense, in problem solving. Like all rules there are some areas where cartographic rules cannot be planned. In other words if a particular type of error has not previously occurred, a problem solving strategy cannot be developed for it. This means that considerable weight must be placed on the cartographer's initiative and experience to ensure the quality of maps.

- Often there is no well documented process regarding the selection of right/proper tolerance. In general, test and trial is the optimal way since each data set may require a specially tailored tolerance to minimise the manual work.

The responses to key questions are discussed below.

Time spent on generalisation for both modern and traditional cartographic environments: **Figure 6.5** illustrates the time spent on the generalisation process. The time spent on processing the data has a significant effect upon delivery of the product in terms of both quantity and quality. Perhaps it is time that mapping agencies gave more serious consideration to automating the generalisation process where high quality generalisation should be an objective rather than a fast and simple approach. Time should be considered as a source of error and uncertainty by NMAs. Hunter and Goodchild (1995 and 1996) discussed uncertainty in spatial databases, and recommended that the sources of errors should be well explored. The complexity of the process is always an issue and represents a significant barrier to progress. Errors come from a wide range of sources, such as map registration, data conversion from raster to vector, interpretation, analysis, etc. It is useful to measure errors for every given application.

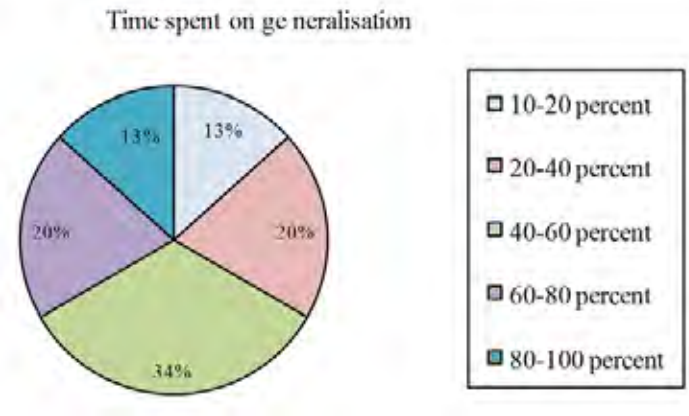


Figure 6.5 Time category

Generalisation competency in both traditional cartographic environments: The majority of respondents demonstrated good or better generalisation competency: 36% had an extensive level of competency in generalisation tools, 29% reasonable, 21% moderate, only a small proportion (7%) had limited, minimal or no competencies (**Figure 6.6**). This is particularly important as the results of the process totally rely on the level of familiarity with the generalisation tools in order that the best results can be obtained. The respondents were given a clear indication of what is meant by 'generalisation tools', as many people will have different ideas of what are simple or sophisticated generalisation tools. Their level of understanding would obviously influence their response to these qualitative questions.

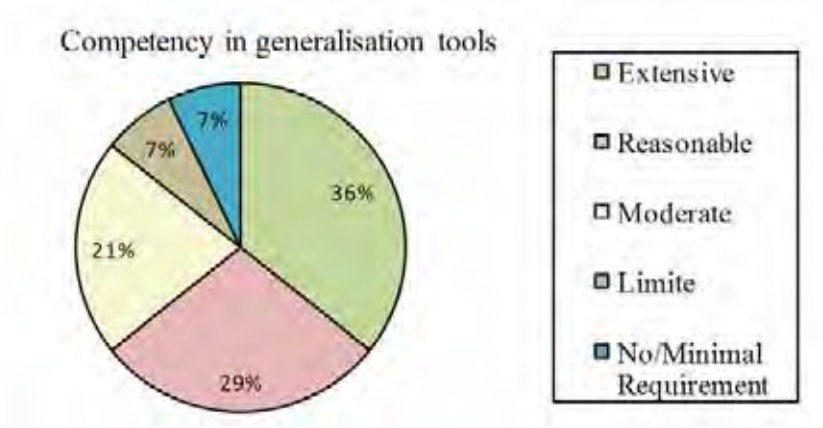


Figure 6.6 Generalisation tool competencies

Expertise in generalisation: In response to question 3 (Thinking about the transfer of map and database generalisation knowledge within your agency/company, how would you rate the expertise in generalisation there?) **Figure 6.7** shows the largest number

recorded, i.e. that map generalisation technology needs improvement. Due to the rapid development of software and new technologies, it was required that the respondents clearly articulate their expertise in using generalisation tools.

Figure 6.7 Expertise in generalisation tools

Importance of generalisation to your project: The majority of respondents identified the critical importance of generalisation techniques to their project (**Figure 6.8**). However, even though agencies rated generalisation very highly, their expertise ranged from ‘beginner’ to ‘advanced’ level. This could be due to the fact that the technology is developing so quickly. There may also be economic impacts on the organisation's funds and human resources that may affect the rate of implementation.

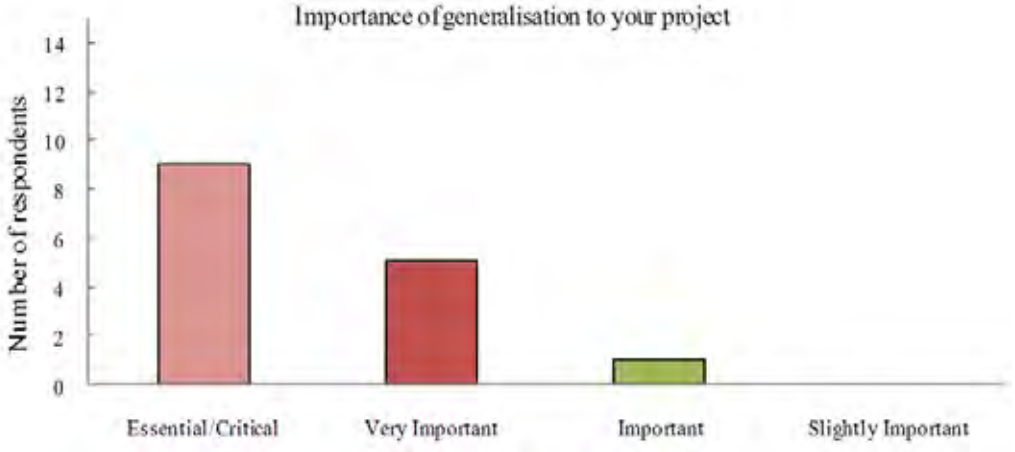


Figure 6.8 Importance of generalisation to target sample’s project

Importance of generalisation to overall success of your organisation: **Figure 6.9** speaks for itself. It indicates that 57% of the respondents rated generalisation as being ‘very important’ to the overall success of their organisation, and 29% respondents categorised it as 'essential'. Only 14% answered 'no opinion '. Most of the projects in NMAs involve

the updating of various databases from road to vegetation, and other map features. All these processes are time-consuming, costly and, possibly, lead to inconsistent results (e.g. due to the use of different operators).

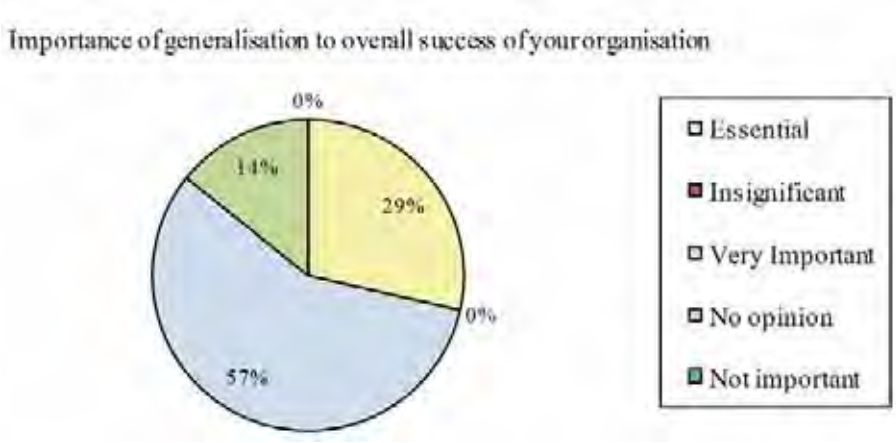


Figure 6.9 Importance of generalisation to overall success of your organisation

Understanding of emerging technology in generalisation: This question draws attention to the importance of an awareness of generalisation techniques or technologies, and their progress. It is often worth testing new techniques in a pilot project so as to evaluate its usefulness, and also to introduce and educate the relevant staff. **Figure 6.10** illustrates that the majority of respondents had limited knowledge of emerging techniques/technologies in generalisation. A key conclusion is that this lack of understanding will affect the results, causing inconsistency and errors, and therefore affecting the accuracy and integrity of the database. It was taken into account that the respondents have a good grasp of 'emerging technologies' such as cutting edge R&D and new products on the market. It was also assumed that respondents have a similar level of understanding of the concept.

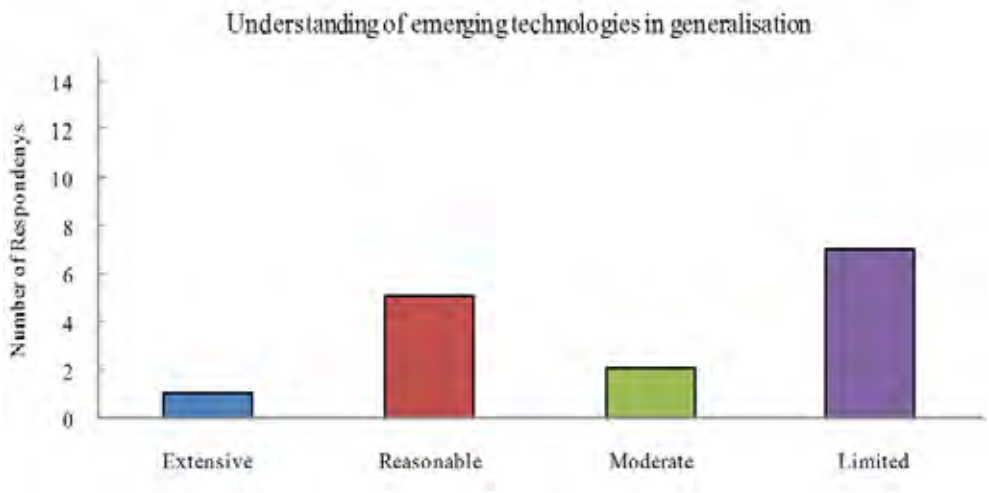


Figure 6.10 Understanding of emerging technologies in generalisation

Satisfaction rate with current performance of generalisation: Virtually all the respondents recorded a positive rating for the increasingly important role of generalisation in current and future projects, although a small proportion rated it as 'not important'. Also it should be acknowledged that there were a large number of 'no opinion' responses. This inconsistency implies that the new technology should be capable of satisfying a range of needs (**Figure 6.11**).

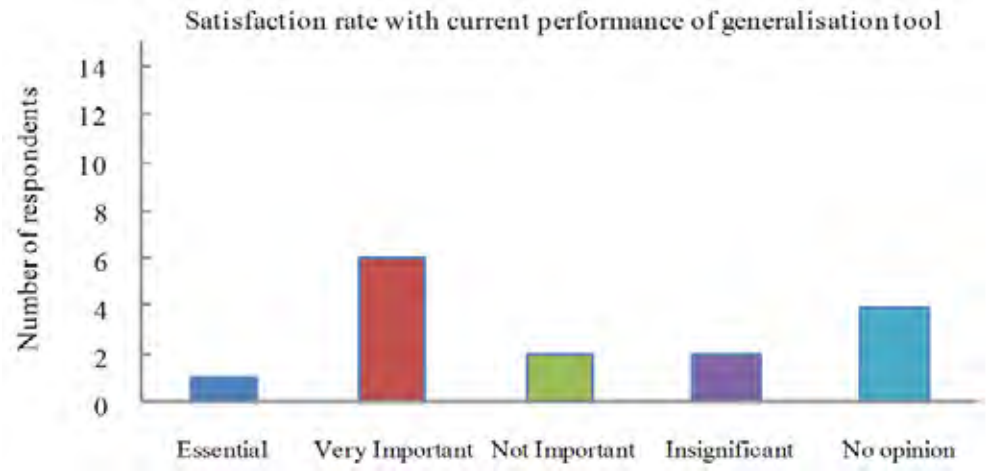


Figure 6.11 Satisfaction rate with current performance of generalisation

Generalisation operation used for roads: A significant proportion of respondents exploit symbolisation, simplification and selection operations in their generalisation process (**Figure 6.12**). The survey focused on the most commonly used generalisation operators. The components of a generalisation process fit together like a 'jigsaw

puzzle', and the whole process should work in harmony to deliver an accurate, consistent and well-structured result. There is a wide range of software offering a choice of algorithms and operations, and choosing the right one for the data in hand is not a straight forward or automatic process. However, various software tools were tested in order to select the optimal one to meet the requirements for a given dataset.

Figure 6.12 Performance of frequency use of generalisation operator

A significant proportion of respondents exploit symbolisation, simplification and selection operations in their generalisation process. They focus on the most commonly used operators. There is a range of software offering a choice of algorithms and operations, and choosing the right one for the data in hand is not an automatic process.

Handling selection of appropriate/optimal tolerance in simplifying lines using simplification: This question was intended to draw attention to the range of approaches taken to the selection of the right/proper tolerance value(s). Two respondents had 'no opinion', but others made comments and suggestions for improvement which are summarised here:

- By testing various tolerances, and selecting the appropriate tolerance by viewing the results.
- Using detailed specifications and applying size criteria to the algorithms.
- By setting the tolerance at a smaller value there will be less displacement.
- By manually setting the tolerance.
- By editing at 1:250,000 and comparing with reference datasets.

- In general, test and trial is the optimal way since each data set requires a specially tailored tolerance to minimise the manual work.

It was revealed that there is no documented process regarding the selection of the right/proper tolerance. In general, ‘test and trial’ was the optimal way to determine the tolerance value that minimises the manual work.

Do you use cartographic rules, guidelines, workflow when undertaking generalisation: Overall the answers were divided into two broad categories: rule based decision-making; and practical decision-making without reference to specific rules:

- Using experience and not a list of rules.
- Referring to spatial data specifications.
- Rules for roads include: roads can intersect; short segments less than 1250 m with a dangle cannot exist. Priority is given to road hierarchy such as retaining principle roads. Tracks are often generalised.
- All generalisation is output scale driven.
- Maintaining the topological relationships between features.
- Using combination of experience and rules.
- Mostly according to cartographic rules.
- Experience rather using just written rules.
- Sometimes there are no rules to follow just relying on experience.
- Mostly cartographic rules plus common sense.
- By knowing the common errors and trying to fix them according to cartographic rules.

These responses indicate the important role played by cognition, or common sense, in problem solving. If a particular type of error has not previously occurred, a problem solving strategy cannot be developed for it. This means NMA must place considerable weight on the cartographer’s initiative and experience to ensure the quality of maps (e.g. Lee, 1992; Kilpelainen, 2000).

In response to the question of *'How would you evaluate accuracy'*, the following are the most important recommendations by respondents:

a) Because a 100% automatic generalisation is not able to be achieved, the output that requires the least manual editing afterwards is considered the most valuable. Sometimes it is not the most advanced generalisation which generates the most valuable output. The assessment of the output is important.
b) Evaluation during the process (AGENT technology) that offers some measures of displacement.
c) Critical – most of this decision-making is undertaken: a) with a reasonable appreciation of the region, and b) with a cognitive approach to what are major and minor roads. There is no set standard, each situation is considered separately based on available information and 'gut feel'.
d) This assessment is extremely important. Ideally the user wants to have a scale-less mapping environment for map production with an output of varying map scales. In the case of roads deriving the hierarchy of features between scales will increase or decrease the number of features per scale but will maintain the relationship between databases.
e) Accuracy (and tolerance) is of critical importance in large scale mapping output.
f) Scale plays an important part in map production. When producing a map of 1:500,000 from 1:250,000, some of the map details will be removed due to limited space, but most of them will stay. Producing a map of 1:1,000,000 from 1:250,000, most of the details in 1:250,000 national topographic data will be removed; the vertex and sharp angles in arc features will be removed; the displacements between features will increase. The degree/processes of generalisation will be higher. As a result the quantity of features in the map will decrease. However, if the details of the map present correctly in topological relationships, the quality of the map can be maintained.
g) By referring to the small scale and final checking.
h) Using 1:25,000 as a source map.
i) Double checking, which is time consuming.
j) Mostly rely on software and, in some cases, checking by a different cartographer.
k) Not all the errors are related to process. In some cases they are related to the data source, but will be checked and controlled several times.

Major comments to *'what emerging generalisation research and development area will have the greatest impact on the future direction of automated generalisation technology'*. Examples of feedback are:

- Decision-making systems rather than new specific algorithms.
- The development of Clarity™.
- Roads and stream features – much beyond this and you are moving into purely subjective decision-making.
- Annotation.
- Internal development of spatial information resources (Arc database connection).
- The method of handling Annotations.
- With regards to the cost of this sort of research it depends on budgets.
- With new technology it is good if universities and NMA's put some effort in this regard.
- Quality is better than quantity.
- More research in particular area.

Is there any specific topic area that you feel universities and the spatial information industry should be pursuing for future research and development? Examples of feedback are:

- The universities should look at how to solve the complexity when generalising a dataset with many object types, rather than looking at making a good algorithm for a specific object type seen in isolation.
- Label selections.
- Feature link annotation, size of file structures, and advance technologies, e.g. communications.
- Locating of budget in these fields.
- Automation of the process as much as possible.
- Automation rather than manual editing is preferred.
- More automatic process.
- 100% automatic process.

How satisfied are you with the current overall performance of the generalisation tools? The majority of respondents selected 'satisfied' and 'no opinion'.

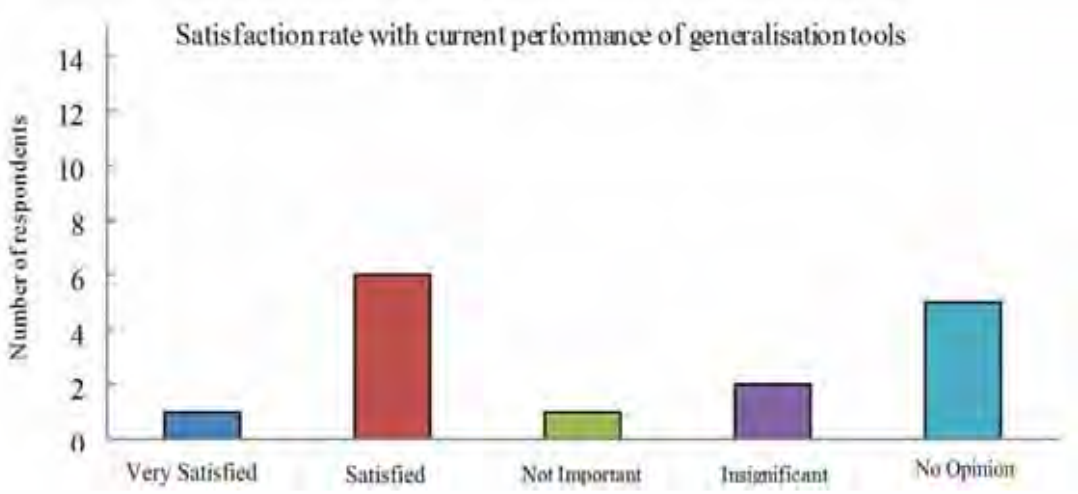


Figure 6.13 Satisfaction rate with current generalisation tools

What is the most effective generalisation framework to derive various types of maps at different scales, e.g. at scales ranging from 1:250,000 to 1:10,000,000?

There are many generalisation frameworks developed by researchers and cartographers that emphasise the graphical and semantic properties of certain features. These generalisation frameworks enable cartographers to effectively handle spatial information and represent it in multiscale databases. Semi-automated frameworks and procedures are both in use for cartographic and database generalisation applications. However, the quality assessment in map generalisation, formalising knowledge in the generalisation process, and feature conflict detection and resolution are examples of challenging issues for researchers, in order to develop efficient automated generalisation workflows for deriving various types of maps at different scales.

Virtually all the respondents recorded a positive rating for the increasingly important role of generalisation. However there were a large number of ‘no opinion responses’.

Exploiting the Results: Fundamentally a sample survey is only effective when its results are shared with the intended audience, and if it is used to solve a particular problem. A carefully conducted cartographic knowledge collection survey can be negated by a poorly written report. NMAs hope to convey the message to both participants and to

computer scientists to highlight the views of cartographic practitioners so that the heuristic knowledge derived can be drawn upon to build an expert system.

6.7 CONCLUDING REMARKS

This chapter firstly provides a brief summary of an expert system. It articulated that the cartographers' knowledge acquisition is an integral part of generalisation systems. This will facilitate the development of a powerful, flexible and robust expert system, which is capable of generating (composing and editing) and manifesting (exhibiting and demonstrating) an innovative method for semi-automated road network generalisation. Without doubt, a comprehensive evaluation of generalisation systems and their performance is essential to embed the cartographic knowledge from experts and bring it into a generalisation framework.

While obtaining representative samples in this survey has proven to be a difficult process, the survey revealed that cartographers' knowledge is not being consistently communicated to generalisation software developers. Nor is that knowledge documented consistently across mapping agencies. Out of the 75 agencies that expressed interest in participating in the survey only a total of 26 responses were received from 15 agencies, representing a 20% response rate. The majority of respondents were from the NMAs and SMAs.

Both cartographic knowledge and rule-based decision-making are used. However, it was suggested that knowledge-based rules should be formulated in the software. The survey responses indicate the important role played by cognition, or common sense, in problem solving. If a particular type of error has not previously occurred, a problem solving strategy can not be developed for it. This means that considerable weight must be placed on the cartographer's initiative and experience.

The aim is to reach widespread agreement among cartographers in relation to specific knowledge about map and spatial data generalisation. The agreed methodological guidelines and procedures could then be incorporated into future software tools to make generalisation operations and algorithms fairer and more reliable. In order to achieve this goal, a set of tools, guidelines and protocols that incorporates a standardised

cartographic generalisation methodology needs to be developed and made available to the cartographic and GIS software communities.

CHAPTER 7: DEVELOPMENT OF A GENERALISATION EXPERT SYSTEM

7.1 INTRODUCTION

Spatial intelligence decision-making relies on accurate, economical and viable digital spatial information products which underpin ‘e-government’ initiatives and location-based services. Automated generalisation systems reduce the cost of maintaining multiple data models and digital maps at different scales. To facilitate automated generalisation, a detailed generalisation framework for deriving multiscale spatial data has been developed by the author (Kazemi and Lim 2007a and 2009). This involved an assessment of existing generalisation systems, undertaking an international survey of cartographic generalisation practices and developing a knowledge-based expert system on feature generalisation. In order to overcome inadequate handling of scale inconsistency in modern day spatial updating efforts, a Generalisation Expert System (GES) has been built in Java-Python-C, enabling automated generalisation of thematic data at a range of scales. This is achieved by a new approach to spatial data revision that assimilates a scale-independent data model and interactive cartographic generalisation processes.

The approach takes advantage of NMA, SMA and a number of software vendor’s inputs in relation to the knowledge acquisition process of cartographic practices, via the cartographic generalisation survey reported on in Chapter 6. The findings from the survey are used to help define the requirements for a knowledge-based spatial database in which the cartographic knowledge and heuristic rules are formulated as a series of rules. These feed into a GES to deliver coherent capabilities and automate the generalisation of features as much as possible for 'derivative mapping'.

This chapter reports on the development of a GES. A brief description of key steps in building a GES and its components are presented. Its capabilities will be demonstrated in a case study involving the simplification of 1:250,000 national topographic data to 1:500,000 scale over Canberra, Australia. The GES has a simple Graphical User Interface (GUI) that can assist users without requiring a high level of technical skill and knowledge of spatial data management.

7.2 METHODOLOGY

In this study a rule-based generalisation expert system was developed. It is interfaced with GIS software. Fundamentals of the expert system incorporate cartographers' knowledge and intuition to generalise lines and polylines. Rules are symmetrical so they can be processed in either a forward or a backward approach. A forward chaining (data-driven) approach attempts to reach a conclusion through bottom-up reasoning; initially starting from the proof and details, to the top-level conclusions that are based on facts. The operator begins by assuming a conclusion to be true and then applies the rules to prove it (Giarrantano and Riley, 1989). In contrast, a backward chaining (goal-driven) approach starts processing a given problem from the hypotheses (top-down inference) down to the lower level facts that support the hypotheses. It begins with a conclusion and proves the conclusion via providing the veracity of each premise in a left to right, or top to bottom order. As discussed in **Chapter 6**, expert systems are composed of four key parts: (a) knowledge representation; (b) inference engine; (c) knowledge representation; and (d) user interface (see **Figures 1.1, 6.1** and **7.1**). These are implemented specifically for semi-automated road network generalisation.

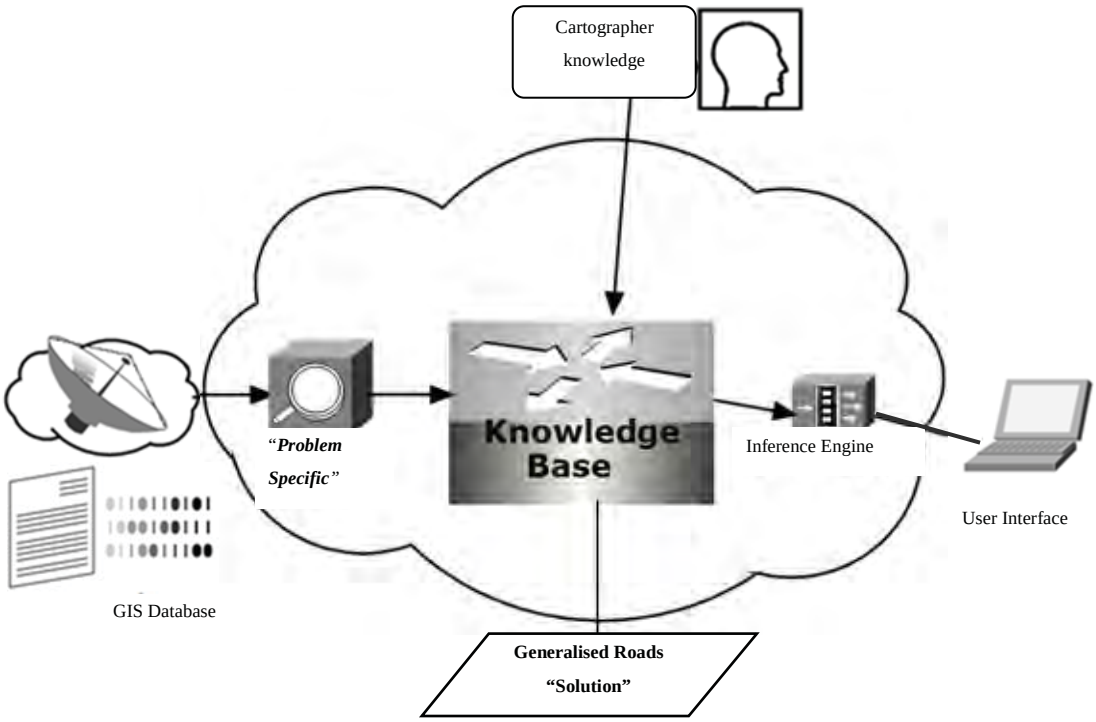


Figure 7.1 Components of a conceptual expert system

GES uses operations, algorithms and knowledge-based rules that are elaborated on in the following sections of this chapter. Previous research on road generalisation was reviewed in **Chapter 2.** Key components and the methodology of this research are shown in **Figure.1.1** (**Chapter 1**) and include the following:

- A detailed review and evaluation of current generalisation methods and solutions.
- Developing a conceptual methodology to generalise 1:250,000 national topographic data into 1:500,000 and 1:1,000,000 scales.
- Cartographic knowledge acquisition by undertaking an International Cartographic Generalisation Survey and analysis of the survey results to build a rule-based expert system.
- Rule collection and encoding in Java-Python-C programming languages to construct the GES.
- Demonstration of GES application for spatial data mining and generalisation for three different spatial features.
- Assessment of the overall generalisation performance using referenced maps and experienced cartographer's feedback.

7.2.1 System Architecture and Key Features

A graphical overview of GES architecture is shown in **Figures 7.2 and 7.3**, and its GUI, key features, tools, and windows are presented in **Figure 7.4**. The system is built in the Java-Python-C programming environments with input from the cartographic knowledge acquisition process, based on the International Cartographic Generalisation Survey (Kazemi and Lim, 2007a). The system consists of four main components: graphical interface, setting, algorithms, and outputs of spatial attribute data. Each component is explained in the following sections.

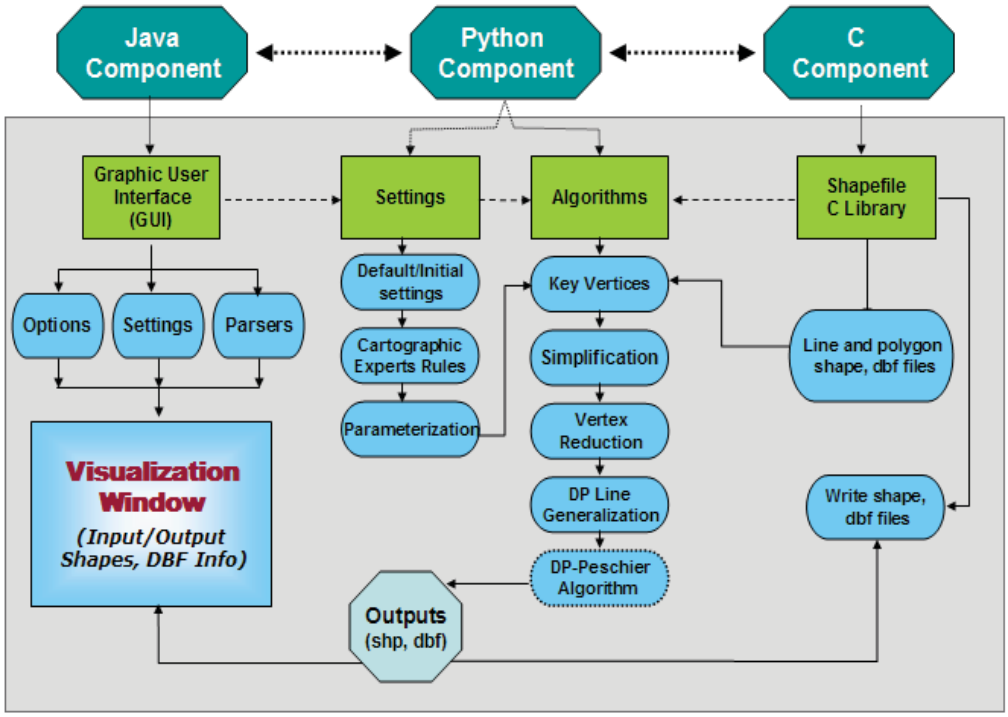


Figure 7.2 Architecture of GES

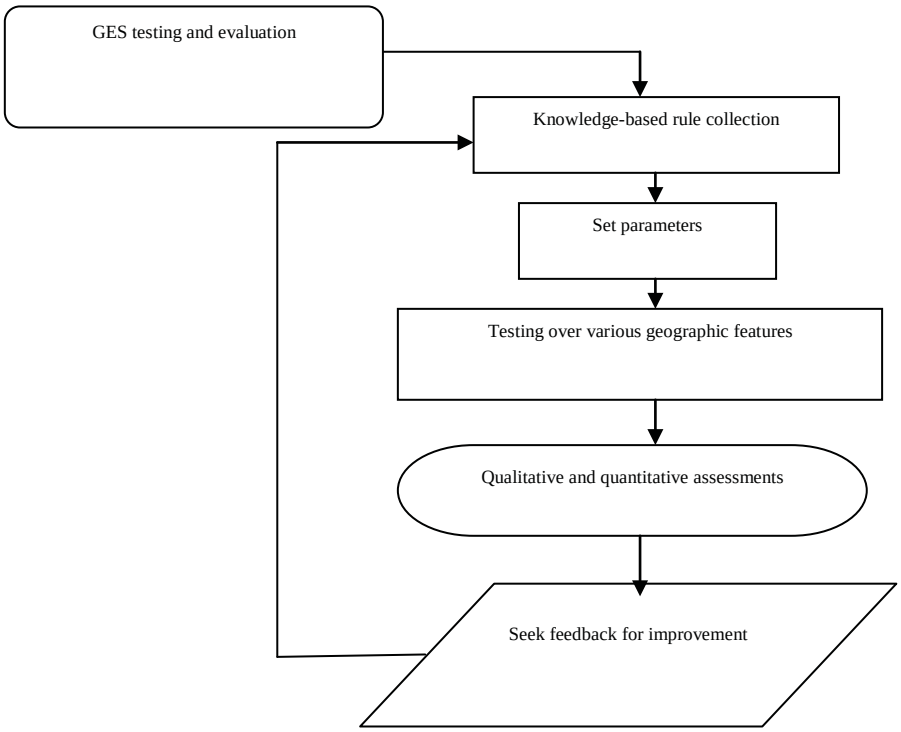


Figure 7.3 Roadmap showing the rationale and steps for organisation of the chapter and testing GES

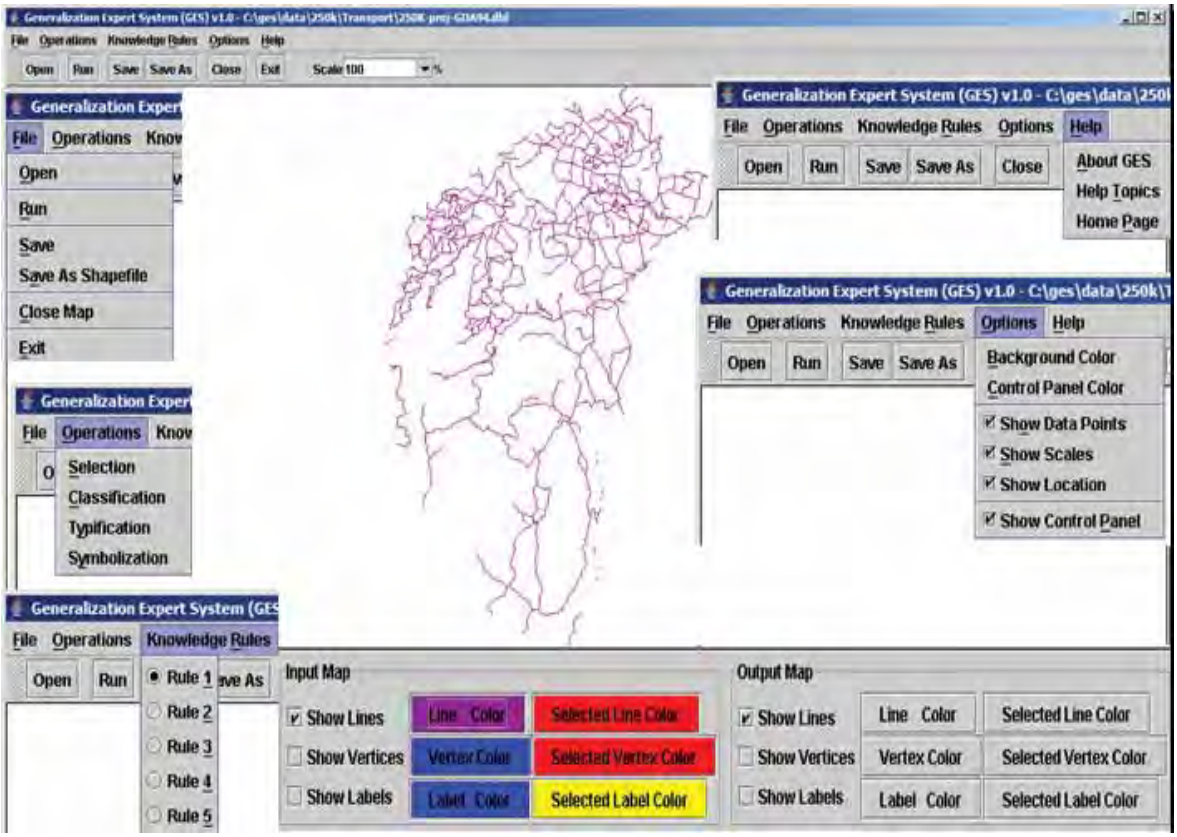


Figure 7.4 Graphical user interface of GES

The main knowledge-based contextual information is as follows; these characteristics assume that roads are already identified and mapped:

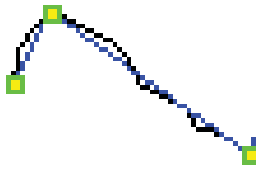
1. *Geometry or topology of features such as roads.* Topology maintains spatial relationships among geographic features. During generalisation processes, GIS tools handle topology through a set of validation rules that define how features are to share a geographic space (e.g. polygons should not overlap and lines should not cross) whilst preserving shared geometry, connectivity and other related topological relationships (Lee, 2004). The shape of a road is an important factor for a contextual classification. Functional requirements, terrain and engineering limitations affect the geometrical and physical form of a road. For example, slope, width, and curvature of a road require an upper boundary. Roads can intersect but rivers join. In the GES, the existing road map was added to the knowledge-based data. Referenced road maps contain clues that guide the spatial analyst to recognise analogous roads according to structural characteristics and other parameters. ArcGIS™ is used to construct the topology of features in order

to obtain input to the GES. For example, ‘undershoot’ errors can be removed by ArcGIS™. This maintains connectivity of road segments in the road coverage.

2. *Land use.* Land use classification typically relies on visual interpretation. However, this model is not sufficient for roads that do not exist on maps or extend over multiple regions because, in general, the density of roads is related to the type of land use. For example, in a high density urban area, the road network is likely to be complex and dense. Different types of land cover can be associated with different types of road network topology. That is, many crossroads are connected to many road segments in an urban area. This contextual information enables the analyst to extract roads.
3. *Drainage pattern.* The drainage pattern is also incorporated, as it has an effect on the appearance of road structures. Roads normally follow contour lines in valleys but are less curved than channels and rivers. This is essential for road construction in order to minimise the number of bridges on road-river crossings. Co-linearity and connectivity are also considered, and drainage networks are used to avoid confusion between water and bridges as a road segment may be bounded by water.
4. *Elevation.* Topography has an effect on the appearance of road structures. A digital elevation model can be used to indicate plausible road tracks in an image. Even in a mountainous area, a road between regions (e.g. towns and countryside having almost the same altitude) follows a similar altitude. In an area with a high slope, a line is unlikely to be a road unless it is approximately parallel to the contours, although there are of course exceptions. For example, line elements (e.g. fire lanes in a forest) are at right angles to the relief. Therefore, an elevation layer is employed in the GES.

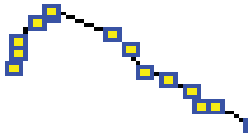
Rule deduction deals with the knowledge acquisition stage that is known as the inference mechanism. In this phase the information generated and collected from the prior phases is aggregated into a rule-based view to maintain consistency and reliability for the utilisation of a cartographers’ experience in feature generalisation. Once the knowledge is collected in an appropriate form, the next step is to convert the collected knowledge into a programming set in order to build a rule-based expert system and to develop procedural code to execute instructions against a database. A rule is a

combination of knowledge that represents an antecedent or condition and its immediate consequence or conclusion. Examples of generic rules used in the construction of the GES are shown in **Figure 7.5**:



Threshold of Line Tolerance for DP algorithm. Default Value is 0.0
If Line_Tolerance <= 0.0, GES will not run DP algorithm even if Apply_DP = Yes
default Value is 0.0Line _Tolerance = 0.005

Rule1



Threshold of Point Tolerance for vertex reduction.
If Line_Tolerance <= 0.0, GES will do nothing
default Value is 0.0Point _Tolerance = 0.15

Rule2

Figure 7.5 Examples of GES rules

Rule deduction is concerned with knowledge representation and map production, which are driven by cartographic knowledge using the spatial database. Many GIS applications have been inductive rather than deductive (e.g. Franklin, 1995). Inductive structures deal with finding general rules and inferring from examples. On the other hand, deductive structures are those in which a known general relationship is employed for particular observations. This allows the users to query the identification of all areas in which a known relationship or desired set of premises are satisfied.

The forward chaining process is considered in order to evaluate all rules for a given feature segment. The computation time linearly increases as the number of vertices and lines increases. The forward chaining (bottom-up) search is time consuming since it leads to a larger number of choices. This is particularly problematic if the database is large, i.e. the number of features and rules is large. For example, if there are 1121 line segments in an elevation layer, then a significant number of vertices have to be processed. This process takes approximately 8 hours using Compaq Presario v2000 Notebook (Windows XP, Intel Pentium M 710 - 1.4 GHz , 256 MB RAM, and 40 GB hard drive). In addition, the more rules, the more computation time is required. In practice, it is tedious and time-consuming to use such a database since it requires a powerful computer with huge RAM and hard drive space. In this study, a small subset

of the data over Australia's capital city, Canberra (covers approximately 23,630 km²) is tested (**Figure 3.1**).

Generalisation operations and algorithms of the GES (Kazemi *et al.*, 2009a) are shown in **Figure 7.4**. Some require further implementation / development (**Table 7.1**).

Table 7.1 Status of GES generalisation operations and algorithms

Generalisation operations and algorithms	Comment
Selection	This operation is operational in GES
Classification	This operation is operational in GES
Typification	This operation is immature in GES
Symbolisation	This operation is immature in GES
Vertex Reduction	This algorithm is operational in GES
Merge	This algorithm is operational in GES
Douglas-Peucker- Kreveld and Peschier (1998)	This algorithm is operational in GES

The three main modules in the GES include: shapes.py, a class of shape objects supported by the GES for Points, Areas (Polygons) and Data Structure. It also includes a set of data structures representing entities such as maps, tables, points and generalisation algorithms (**Appendix 4**).

7.2.1.1 Communication Among Components

GES consists of three main components that are implemented in various programming languages. Communications between these components are carried out in varying ways:

- Direct calling using built-in functions:
  - Java methods call C executable programs
  - Python algorithm methods call C executable programs
- Indirect calling uses Unix/Linux Shell Scripts as a bridge:
  - Java GUI calls Python algorithms using 'gui_run_algorithm.sh'
- The returned data is stored in temporary plain text files.

GES uses the Shapefile C Library [open source software] (http://download.osgeo.org/shapelib or http://shapelib.maptools.org/dl) read and write shape and dbf files (binary). There are four executable programs in the GES that were developed from the Shapefile Library (**Table 7.2**).

Table 7.2 GES executable programs

Executable programs	**Comment**
shpparser.c/shpparser.exe	parse input shape data (binary) and then output as plain text file (ASCII)
dbfparse.c/dbfparse.exe	parse input dbf data (binary) and then output as plain text file (ASCII)
shpwriter/shpwriter.exe	write out final results into shape file (binary)
dbfwriter/dbfwriter.exe	write out final results into dbf file (binary).

7.2.1.2 Java Client for Generalisation Definition

A Java interface through a GUI is provided to set up the parameters for the generalisation processes (**Appendix 4**). **Figure 7.4** shows visualisation of input and output for the interactive completion in case user/cartographer intervention is required. Spatial data generalisation requires a large number of user/cartographer-defined parameters. In the GES (Kazemi *et al.,* 2009a) the majority of those parameters are established by the mapping specification, defining what is to be achieved, coupled with a set of constraints that introduce limits on allowable change (Neuffer *et al.,* 2004). Main features/functions include:

- Choice of user-defined knowledge rules for defining thresholds and algorithms.
- Choice of displaying different parts of GIS data, e.g. lines, vertices, and labels.
- Choice of display colours for different sets of GIS data.
- Displaying both input and output GIS maps in the same window for comparison.
- Highlight selected shape files with prompt text.
- Zoom and/or move the GIS map.

7.2.1.3 C Client for GIS Data Interoperability

This component utilises the open source Shapefile C Library (Version 1.2), developed by Warmerdam (1999), to parse input shape files data and generate output shape files. The Shapefile C Library provides the functions for developing simple C programs for reading, writing and updating (to a limited extent) ESRI shapefiles, and their associated attribute files (.dbf). Four C-based standalone mini-programs have been developed (see **Figure 7.1b**).

7.2.1.4 GES Directories and Coding Components

The following provides a summary of GES descriptions for the directories:

Main Directory

- GES

Sub-Directory

- bin: Executable C program, Unix Shell Scripts
- classes: Compiled Java byte code classes for GUI
- config: Knowledge rules and GesLog.log
- data: Example input shp/dbf files
- doc: Documentation for GES. Currently includes only documents for python components, e.g. algorithm.html, datastructure.html and shapes.html
- lib: The Shapefile C library
- python: Python code, including three modules and driver class
- Src: Java source code for GUI
- Tmp: Temporary directory for storing temporary output files, e.g. tempdata_shp.out and tmpdata_dbf.out
- start_ges.sh – a shell script for running the GES Java GUI

To run the GES the user requires a Unix-like environment.. The Cygwin is free software that provides a Unix-like environment on a Windows platform. The user needs to download and execute setup.exe from the Cygwin website: http://cygwin.com/setup.exe

To start the GES program:

- Run *Cygwin*. This requires command prompts such as the syntax below
- *pwd* to check current location then change the directory to where *ges* folder is
- *cd ges* and run *./start_ges.sh*

7.2.1.5 Input Parameters

The inputs include shape and dbf files (binary), and knowledge rules (e.g. 'ges_rule1.dat' in the 'config' directory). Refer to example below.

EXAMPLE

ges_rule1.dat (in the 'config' directory)

Specifies all rules (algorithms and thresholds) to be applied to the GES program. Lines starting with # are the comment lines and will be ignored by the program.

 # This the knowledge rule file for GES
 # Flag for whether or not to print out processing information to the GesLog.log file (Yes or No)
 # Default value is Yes.
Write_Log_File = Yes
 # Flag for whether or not to mark key points before processing (Yes or No)
 # Default value is No.
Mark_Key_Points = Yes
 # Flag for whether or not to merge polylines which share the same key properties (Yes or No)
 # Default value is No.
Merge_Polylines = Yes
 # Flag for whether or not to allow overlapped line segments before simplification (Yes or No)
 # Default value is Yes.
Line_Overlap = Yes
 # Threshold of Point Tolerance for vertex reduction.
 # If Line_Tolerance <= 0.0, GES will do nothing
 # Default value is 0.0
Point_Tolerance = 0.0015
 # Flag for whether or not to apply Douglas-Peucker (DP) *algorithms (Yes or No)*
 # Default value is No.
Apply_DP = Yes
 # Threshold of Line Tolerance for DP algorithm. Default value is 0.0
 # If Line_Tolerance <= 0.0, GES will not run DP algorithm even if Apply_DP = Yes
 # Default value is 0.0
Line_Tolerance = 0.005
 # Threshold of Minimum Line Segment Length.
 # Any line segments whose length is less than this threshold will be removed.
 # If Min_Line_Segment_Length <= 0.0, GES will do nothing
 # Default value is 0.0
Min_Line_Segment_Length = 0.0

7.2.2 Generalisation Outputs

The outputs consist of temporary ASCII (plain text) shape and dbf files, binary shape and dbf files, and a log file to store runtime processing information ('GesLog.log' in the 'config' directory). See example below.

EXAMPLE

GesLog.log (in the 'config' directory)

Records all running information for each execution of the GES. The contents of this log file can be

used for testing and comparison of results under different input conditions. For example, how long does it take to run a process (CPU time), how many shapes and points (vertices) have been reduced after execution (input shapes against output shapes).

EXAMPLE

1. Inputs: Elevation
   Shape Type = Arc
   Minimum Bounds(148.763, -35.92, 0, 0) Width = 0.635
   Maximum Bounds(149.398, -35.126, 0, 0) Height = 0.794
   Number Of Shapes = 1121
   Number Of Points = 101228
2. Parameter Settings:
   Mark Key Points = Yes
   Merge Polylines = Yes
   Allow Line Overlap = Yes
   Point Tolerance = 0.025
   Apply DP Algorithm = Yes
   Line Tolerance = 0.025
   Minimum Line Segment Length = 0.0
3. Outputs:
   Number of shapes = 1121
   Number Of Points = 9491
4. Total CPU Time (h:m:s):
   8:8:51.267

……

GES is able to generalise both lines and polygons. These differ from each other in terms of algorithmic implementation in the GES. For lines, GES will execute the following tasks:

- Find all crossing vertices for lines (e.g. roads), those lines/roads have higher road classes (>3). Mark these crossing vertices as Key Points which must be preserved.
- Merge lines/roads that have the same road name and can be connected together. This will reduce the number of shapes but the total number of vertices will be the same.
- Apply Douglas-Peucker algorithm to those merged lines (e.g. roads).

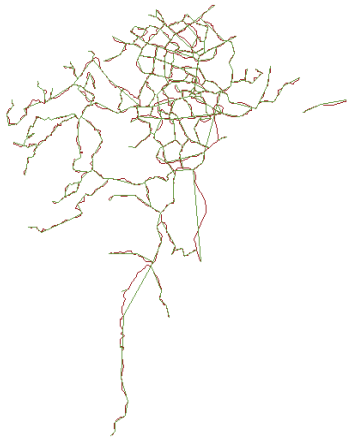
7.2.2.1 Input Dataset for Evaluation

Parts of road, elevation and vegetation layers from the national topographic database of Australia (1:250,000 national topographic data) for Canberra, Australia, were selected as lines and polylines to be generalised and simplified. A case study of real time roads, vegetation and elevation datasets applying different rules and input parameters over this area is shown in **Figures 7.6-7.9**.

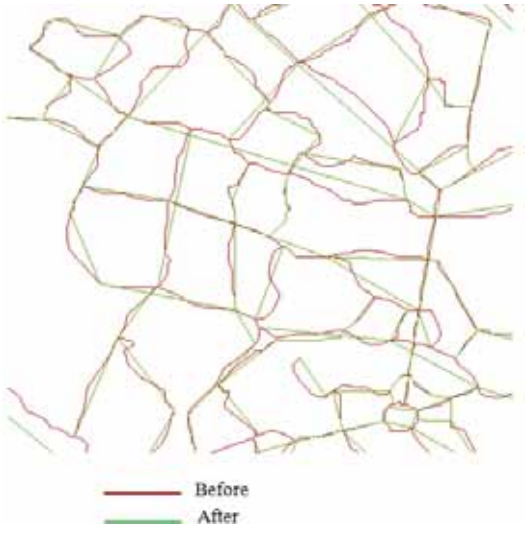
The following operations are required to generalise a road network. The outputs of generalisation workflow applying these operations are shown in **Figure 7.6**.

(a) Classification and Selection

(b) Elimination

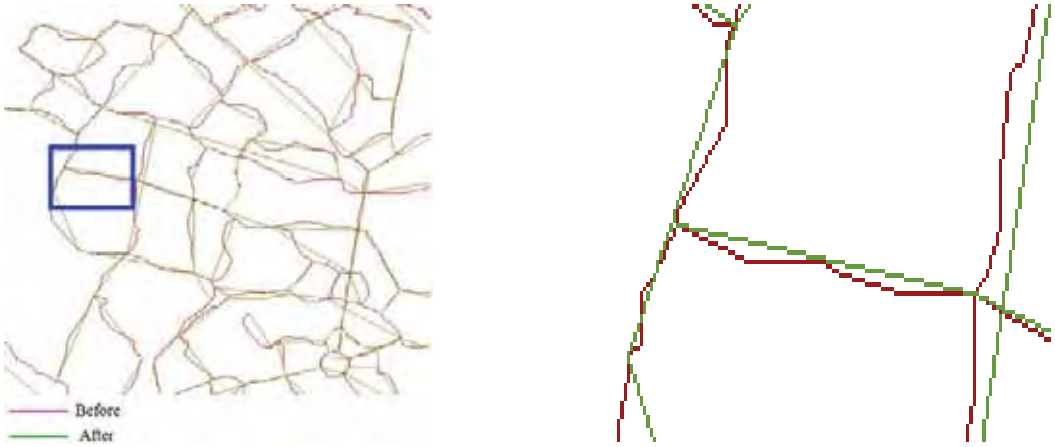


(c) Before and after Simplification



(d) Magnification

Figure 7.6 Road network generalisation of a 1:500,000 map from the original 1:250,000 map, Geoscience Australia © 2001



(a) Bends and junctions

(b) Magnification of bends and junctions

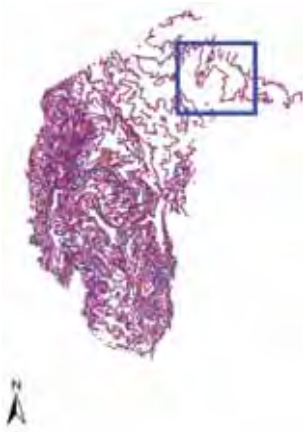
Figure 7.7 Examples of improvements in node junctions, bends and curvatures

7.2.2.2 Contour Generalisation

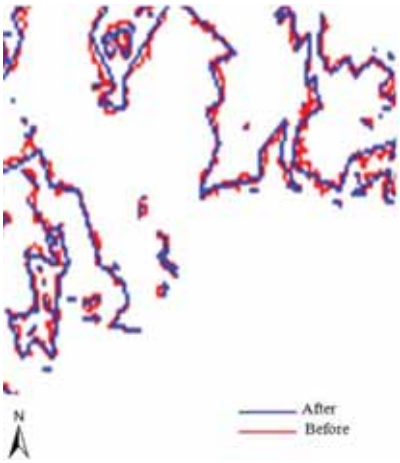
Contour simplification was carried out in order to generate cartographically acceptable graphic shapes for 1:500,000 scale topographic maps. The Douglas-Peucker (DP) simplification algorithm was used; firstly, contours were simplified with a 0.024m tolerance, and then line-crossings with the error band contours were checked and the line segments were marked. A visual output of the simplification results with original contours is depicted on **Figure 7.8.** Contours were simplified within the defined vertical positional accuracy, but maximum horizontal positional accuracy was defined by the simplification tolerance used in the algorithm. The simplification tolerance plays a function in determining the precincts of the simplified contours for flat areas. The simplified contours were obtained within defined spatial accuracy (GA, 2000 and 2001).

(a) Classification and Selection

(b) Elimination



(c) Before and after Simplification



(d) Magnification

Figure 7.8 Contour generalisation of a 1:500,000 map from the original 1:250,000 map with 100m interval, Geoscience Australia © 2001

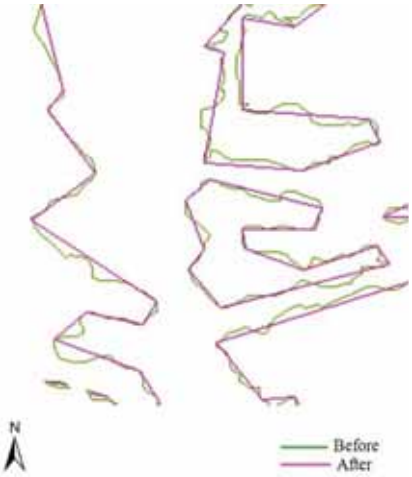
7.2.2.3 Generalisation of Native Vegetation

Users of native vegetation data require different levels of detail that vary from the species of an individual stand, to a class or community of vegetation. These demand different levels of abstraction, i.e. at different scales to satisfy the requirements of various applications. A review of literature necessitates finding an answer to questions such as how to aggregate, what should be aggregated, when to aggregate and how much abstraction is required. This can be achieved by using rule-based generalisation of bio-geographical principles and the spatial distribution of vegetation, which enables automation of the process of multiscale representation of vegetation. The 1:250,000 native vegetation of Canberra is taken as a case study of polygon generalisation (**Figure 7.9**). To produce simplified vegetation patches, the seven land cover categories were identified, starting from the most detailed information at the largest scale possible.

(a) Classification and Selection

(b) Elimination

(c) Before and after Simplification



(d) Magnification

Figure 7.9 Native vegetation generalisation of a 1:500,000 map from the original 1:250,000 map. The area includes vegetation polygons from the range of native vegetation classifications described as part of the 1:250,000 national topographic data specifications, Geoscience Australia © 2001

In addition to the above mentioned features, several examples of applying knowledge-based rules (rule 1 & 2) over various features are shown in **Appendix 3.**

7.3 RESULTS

The results were analysed (**Figure 7.10** and **Tables 7.3 – 7.5)**. A series of generalisation trials were conducted to gauge the sensitivity of simplification results to the different spatial layers. Generalisation performance comparisons are summarised in **Tables 7.3 –**

7.5. According to McMaster (2001)'s cartometric measures, the total length and number of line/polyline segments is used as an index of generalisation to quantify generalisation performance for the target small scale (**Figure 7.10**). For example, there were 101,228 segments and 9,491 segments in the 1:250,000 scale and the 1:500,000 scale contour data respectively over the study area. This means reduction in the complexity and the density of elevation data. Changes in the representation of contour features at 1:250,000 and 1:500,000 scales as a result of generalisation are quantified. The outputs from map derivation have been analysed applying the *Radical Law* (Pfer and Pillewizer, 1966) both employing the GES. The *Radical Law* (Pfer and Pillewizer, 1966; and Kazemi *et al.*, 2009) determines the retained number of objects for a given scale change and the number of objects of the source map. In addition, the outputs compare favourably with GA source datasets such as 1:500,000 and 1:1,000,000.

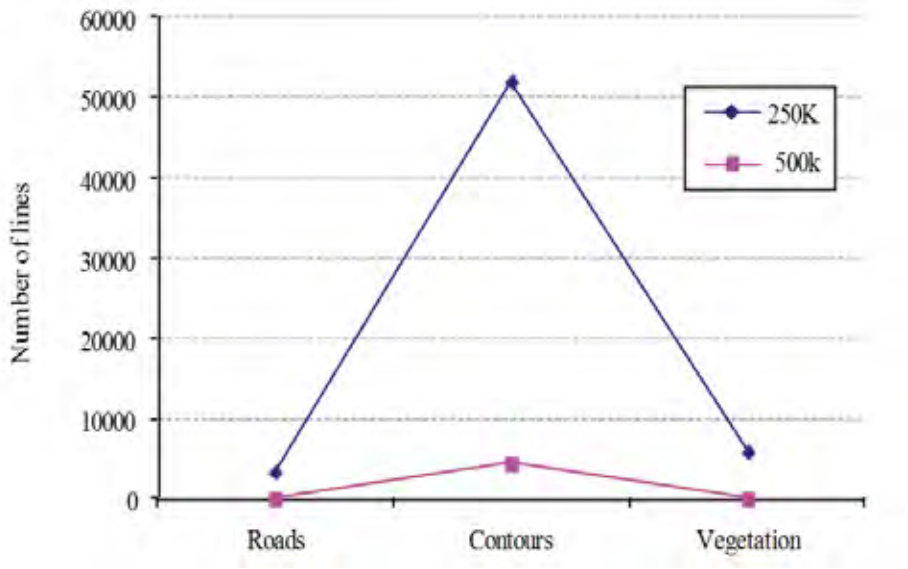


Figure 7.10 Assessment of generalisation results through the GES; the source scale database is 1:250,000 (250K) national topographic data and generalised output is 1:500,000 (500K) scale

The generalisation over roads and vegetation performed better than contours (elevation) data. The outputs of roads, elevation and vegetation generalisation are shown in **Figures 7.6-7.7, 7.8** and **7.9** respectively. The results have been visually assessed through cartographic assessment. Feedback from both the qualitative (expert assessment) and quantitative (cartometric measures) evaluations using measures of goodness-of-fit (Visvalingam and Whyatt, 1990) was incorporated into the development of an enhanced version of GES (v1). A case study of line and polyline generalisation; the original input

GIS maps; and the model-based graphical generalised map were demonstrated earlier (**Figures 7.6 - 7.10**).

Table 7.3 Roads network generalisation input and output parameters for deriving 1:500,000 maps from the source data of the 1:250,000 national topographic data. Some of the key input and setting parameters include Shape Type = Arc, Minimum Bounds (148.763, -35.92, 0, 0), Width = 0.635, Maximum Bounds (149.398, -35.126, 0, 0, Height = 0.794, Number of Shapes = 592, Number Of Points = 51998, Mark Key Points = Yes, Merge Polylines = No, Allow Line Overlap = Yes, Point Tolerance = 0.024, Line Tolerance = 0.024, and Apply DP Algorithm = Yes.

Inputs				**Outputs**		
Number Of Shapes	Number Of Points	Point Tolerance	Line Tolerance	Number of shapes	Number Of Points	Total CPU Time (h:m:s)
456	3457	0.015	0.015	240	693	0:0:1.584

Table 7.4 Elevation generalisation input and output parameters for deriving 1:500,000 maps from the source data of the 1:250,000 national topographic data. Some of the key input and setting parameters include Shape Type = Arc, Minimum Bounds (148.763, -35.92, 0, 0), Width = 0.635, Maximum Bounds (149.398, -35.126, 0, 0, Height = 0.794, Number of Shapes = 592, Number Of Points = 51998, Mark Key Points = Yes, Merge Polylines = No, Allow Line Overlap = Yes, Point Tolerance = 0.024, Line Tolerance = 0.024, and Apply DP Algorithm = Yes.

Inputs				**Outputs**		
Number Of Shapes	Number Of Points	Point Tolerance	Line Tolerance	Number of shapes	Number Of Points	Total CPU Time (h:m:s)
592	51998	0.024	0.024	592	3541	2:2:38.7

Table 7.5 Native vegetation generalisation input and output parameters for deriving 1:500,000 maps from the source data of the 1:250,000 national topographic data. Some of the key input and setting parameters include Shape Type = Arc, Minimum Bounds (148.763, -35.92, 0, 0), Width = 0.635, Maximum Bounds (149.398, -35.126, 0, 0, Height = 0.794, Number of Shapes = 7, Number Of Points = 6085, Mark Key Points = Yes, Merge Polylines = Yes, Allow Line Overlap = Yes, Point Tolerance = 0.035, Line Tolerance = 0.035, and Apply DP Algorithm = Yes.

Inputs				**Outputs**		
Number Of Shapes	Number Of Points	Point Tolerance	Line Tolerance	Number of shapes	Number Of Points	Total CPU Time (h:m:s)
7	6085	0.035	0.035	7	338	0:0:6.916

Selection is considered a pre-processing stage where the content of the map is determined. **Figures 7.6 - 7.9** show the selection process for cartographic features. The features and their corresponding attributes required for the composition of the map are selected and retrieved from the relevant spatial database layer. Scale and map particularities are in use throughout the selection. In the expert system environment the cartographer introduces the category, the scale, and the boundaries of the new map or chart and the system identifies the layers that can be used (original selection). The selection of the features to be portrayed on the map is realised in the GES. The selected features are transferred to and organised in the expert system environment, and those to be considered for portrayal are chosen in accordance with their thematic characteristics (thematic selection).

Quality assessment was carried out firstly by a direct visual comparison between the output (**Figures 7.6 - 7.10**) and the existing maps (1:500,000 and 1:100,000). Notwithstanding the fact that comparison with paper maps is a subjective matter as outlined in other studies (e.g. Chaudhry and Mackaness, 2008; Weibel and Dutton, 1999), it nevertheless provides an indicative success measure for the algorithm. The results of the GES were compared with existing digital datasets and the results were in high agreements when compared with existing generalised mapping products (Geoscience Australia's the digital 1:500,000, 1:100,000 topographic data and

1:1,000,000 Global Map data). It was noted that the algorithm had maintained the overall topological relationships successfully.

It is worthwhile to evaluate generalisation of roads through the use of positional accuracy measures, which is an assessment of the closeness of feature location (e.g. road segments) in the dataset in relation to their true positions on the Earth's surface. The positional accuracy generally includes a horizontal accuracy uncertainty, a vertical accuracy uncertainty, and an explanation of how the accuracy uncertainties were determined (ANZLIC, 2001). The horizontal positional accuracy assessment of roads is conducted once all geometric transformations have taken place. Measures such as root mean square error or standard deviation can be used to represent the variation of vertical accuracy of roads.

Spatial accuracy describes the positional (coordinate) accuracies of spatial data as a result of the generalisation process (multiple-scales). Generalisation quality assessment is a key subject in modern cartography and its automation process influences the ‘fitness for use’ of spatial data. Thus quality assessment is considered indispensable for generalisation of roads in this study. This section reports on a process used for assessing the positional accuracy of generalised roads.

Among several sources of errors identified by researchers (Burrough, 1986; Hunter and Goodchild, 1995 & 1996; Hope *et al.,* 2006), topological and generalisation errors are relevant to this study. The spatial errors of linear features composed of scale-dependent errors as a result of generalisation, and sampling of a line of high geometric accuracy represents the amount of deviations between the interpolated line and the original position of the linear feature. Spatial relationships across road features in a multiple feature type database could offer additional information to support the positioning of road segments. This requires assessing topological and positional relationships to a particular road segment. It is possible to investigate the residuals of displaced road intersections using GPS survey points (Hope *et al.,* 2006). The geometric accuracy of linear features such as roads consists of two key parameters, including positional point accuracy for well defined points on the road network (e.g. intersections), and shape

conformity for a road segment compared to another segment indicating to what extent the curvature of the two segments are similar.

Accuracy of simplified and generalised outputs is determined by comparing the positions of ten well-defined ground control points (GCPs) (here defined as road intersections). The point locations are assessed on the generalised maps (1:500,000 and 1:1,000,000) and corresponding positions from the published 1:250,000 national topographic maps. The source map '1:250,000 digital topographic data' has a basic horizontal accuracy of approximately ±120 metres (GA, 2009). Cartographic generalisation of line and area features introduces errors into the derived map. Generalised output maps have been checked against other data sources including Landsat ETM+ satellite imagery, published 1:250,000 digital topographic data and GPS field information.

The GPS survey datasets were collected by Geoscience Australia (GA) in February 2007. Post-processing for the 70 GPS survey points was performed by Ultimate Positioning Pty Ltd. GA made available a copy of these data sets for this study. It would be useful to include the photographic location of the survey points but unfortunately no picture was taken during the GPS survey. Ten GPS survey points were used for this purpose (**Figure 7.11**) as these points were within the study area. The accuracy of the collected survey data was estimated to be at the sub-metre level.

About 50 GCPs were identified from the 1:25,000 scale topographic data. These GCPs and GPS survey data were overlaid onto the generalised output maps. The tracks and GPS points did not match the road as it was shown on the generalised output maps because of the displacement process, as a result of the simplification operation when originally applied to the 1:250,000 digital topographic data and also during small scale map derivation. For example, to make a map more readable when having multiple adjacent features, one feature may be preserved in its true position and the others may be displaced. Usually hydrographic and transportation features (e.g. railways, roads, tracks) are preserved in their correct positions compared to areal features such as buildings and native vegetation boundaries. Positional accuracies achieved for generalised 1:500,000 and 1:1,000,000 maps are ~120m and ~470m respectively

(**Appendix 2**). The overall agreement among GPS points and road intersections provides an accurate map of a simplified roads database. Measured errors are therefore considered to be within defined mapping standards. Vertical accuracy assessment was beyond the scope of this study, but would be a worthwhile to test in future studies.

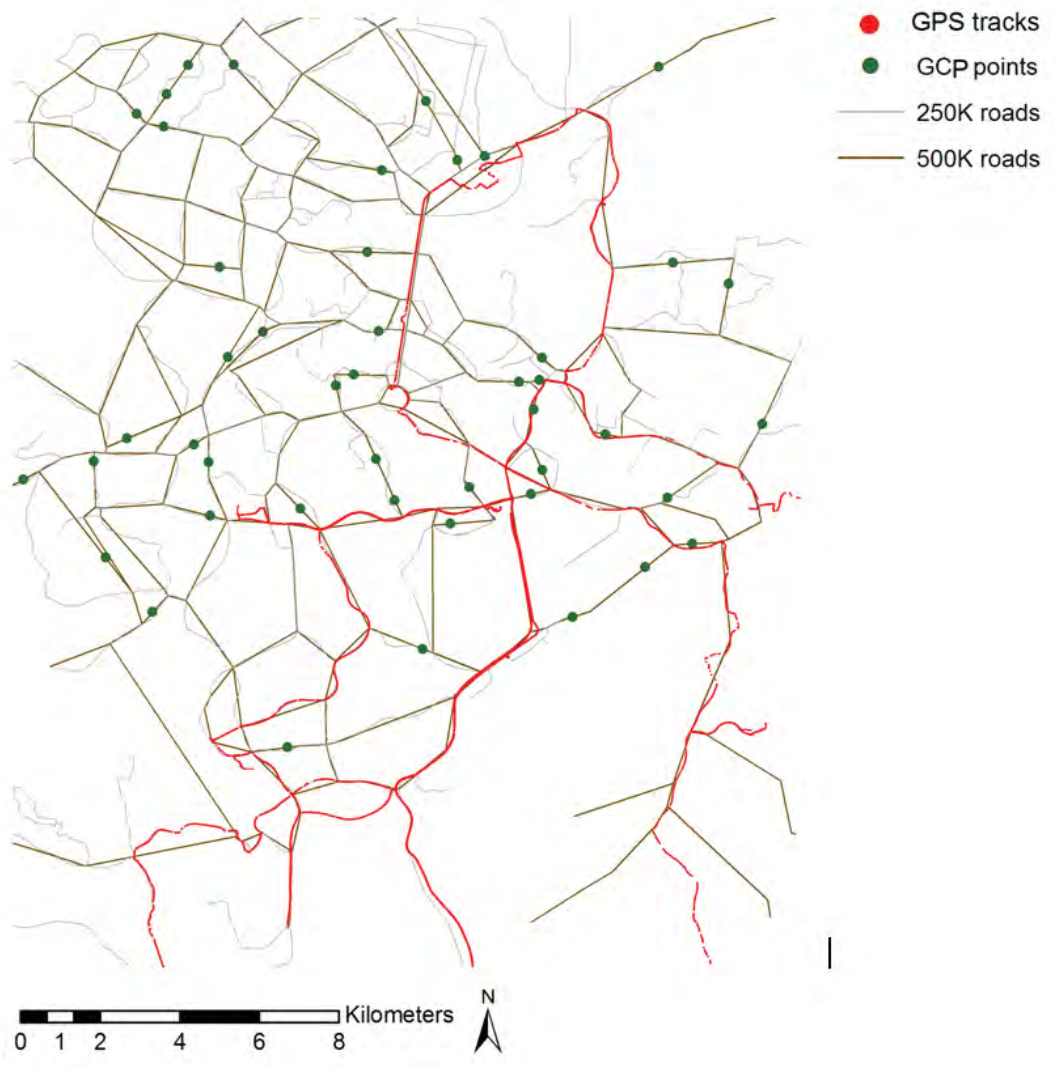
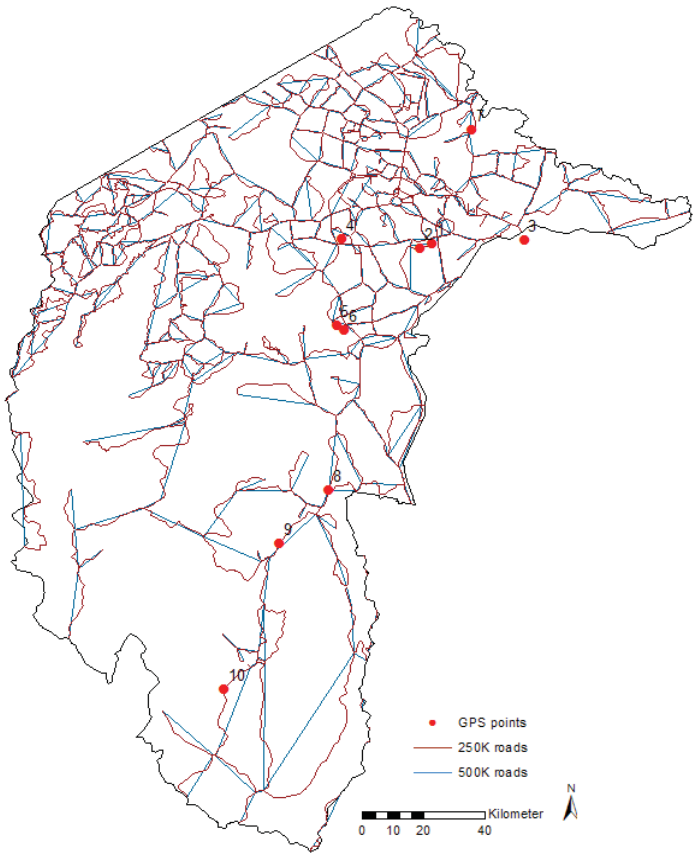
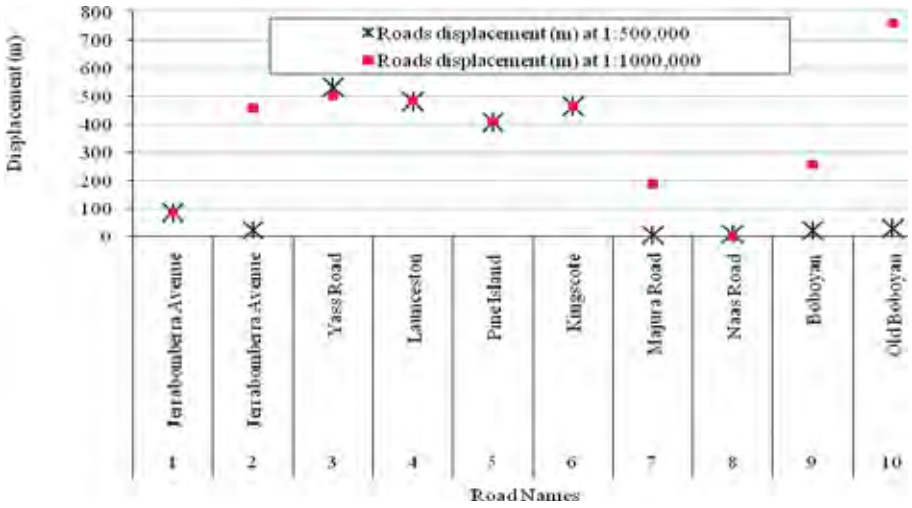


Figure 7.11 The GPS survey tracks and GCP points superimposed to a simplified 1:250,000 and 1:500,000 scale road database

In **Figure 7.12** the results achieved for 1:500,000 and 1:1,000,000 scales are shown for most roads. Displacements of up to 510m among 10 road intersections were surveyed. In the case of Old Boboyan Road (number 10), there appears to be a displacement of 710m.



(a) The GPS survey points superimposed to a simplified 1:500,000 scale road database



(b) Assessment of the average shape changes caused by the simplification operation

Figure 7.12 Assessment of the average shape changes caused by the simplification operation using GPS survey points for a simplified 1:500,000 scale road database

In relation to selected roads intersections, **Figure 7.12** shows the displacement values of the coordinate differences from the original datasets (before generalisation) and derived road map (after generalisation), ranging from 2m to 710m for a generalised road map at 1:500,000 (black line) and 1:1,000,000 (blue line) scales. The residuals of displaced road intersections increase as a function of the scale of roads and the geometric characteristics of road segments.

For example, the Old Boboyan Road reveals the largest displacement distance caused by vertex reduction. The simplification changes road topology (shape, lengths and angles). A set of check points was used in relation to the derived 1:1,000,000 from 1:500,000 data using the simplification operation. **Figure 7.13** indicates the displacement values of the coordinate differences on the derived road map (after generalisation).

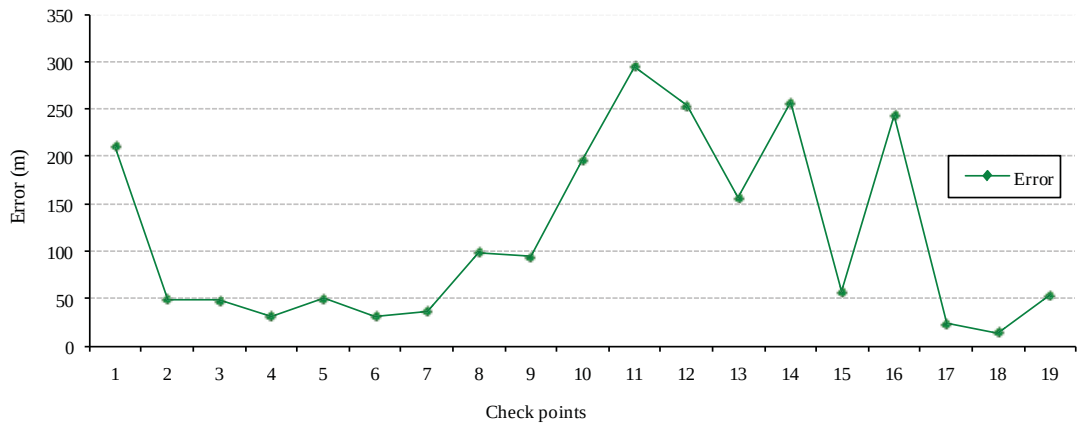


Figure 7.13 Assessment of the average shape changes caused by the simplification operation for 1:500,000 and 1:1,000,000 scales

7.4 CONCLUDING REMARKS

Current map production systems in general provide complex tools and procedural cartographic protocols that still require cartographer interaction. This includes the selection of information, symbolisation of features, maintaining topological relationships among features, and visualisation of graphical conflicts as a result of scale variation, and so on. An efficient generalisation technique aims to improve the graphical quality and legibility of maps.

For many cartographic generalisation procedures there are no algorithmic solutions; instead they often use a combination of manual and semi-automatic approaches. Expert systems use collections of 'rules of thumb' that are mainly heuristic criteria, methods, or principles for deciding which among several alternative courses of action might be taken to solve a particular problem.

The strain of maintaining up-to-date maps at a range of scales for spatial intelligence decision-making is ever increasing. An automated generalisation system meets these requirements for reduced cost and shorter loading time for web mapping applications.

Dynamic database generalisation is required for many applications, such as real time transportation navigation, mobile mapping, emergency management and cartographic map production. A new, semi-automated, spatial data mining and generalisation system was developed for polygon and line related datasets.

The tests established that algorithms implemented in the GES are able to extract characteristic vertices of the original entity lines and polylines (e.g. for roads and native vegetation), while excluding non-characteristic vertices to reduce complexity and improve the efficiency of line/polyline generalisation. This study has demonstrated improvements in vertex reduction, classification and merge, Enhanced Douglas-Peucker (Visvalingam, and Whyatt, 1991) and Douglas-Peucker-Peschier (Kreveld and Peschier, 1998) algorithms.

The test results confirm that the GES generalises line features well and maintains their geometric relations. The results also compared favourably with existing paper maps (e.g. 1:1,000,000). Existing generalisation software requires advanced technical skills from users, however, the GES has a basic GUI that is an advantage for users with limited technical skills and understanding of spatial data management.

Changing geographic parameters should be updated in multiscale maps and spatial databases in “near real time”. The conventional methods focus on a single map at a time and therefore result in inconsistency between databases. The GES is a trial tool for

generalising large-scale maps into smaller scales, and creating maps of different themes across various scales.

CHAPTER 8: RESEARCH SUMMARY AND RECOMMENDATIONS

In this chapter, **Section 8.1** presents the aims and methodology of the research. **Section 8.2** gives a summary of the generalisation expert system development process. **Section 8.3** outlines the research contributions made by this thesis. Finally, **Section 8.4** identifies future research opportunities.

8.1 AIMS AND METHODOLOGY

8.1.1 Aims

The key objective of this study is to develop a framework for generalisation of geographical features. The study defined a series of research questions and developed a methodology that demonstrated the efficacy of a road network generalisation approach. Generalisation techniques were applied over a test area in order to generalise 1:250,000 national topographic data to produce small scale maps through derivative mapping. The study questions included:

- What are the capabilities of existing generalisation systems?
- What does a map/database generalisation framework offer the Geographic Information Systems (GIS) community?
- Why does a map/database generalisation require a systematic framework for simplifying line and polyline databases?
- How does heuristic natural knowledge transfer from cartographers take place in the construction of a generalisation expert system?

These four questions were examined from different perspectives after appraising the state of research in the topic area. Through an examination of the above questions, the objectives of this study were defined as follows:

a) Understand the elements of map and database generalisation from an operational perspective through reviewing worldwide research in both academia and industry.
b) Identify and test the capabilities of GIS that can improve the automation of map generalisation.

c) Apply thematic generalisation techniques that have resulted in the development of a methodological framework for segmentation and generalisation of raster data.
d) Investigate capabilities of expert systems for solving GIS problems with emphasis on automation of map generalisation.
e) Develop a framework for a heuristic natural knowledge elicitation using a Cartographic Generalisation Survey.
f) Design, implement and evaluate knowledge-based solution: a Generalisation Expert System for the delivery of simplified spatial data.
g) Recommend direction for future research in this area.

8.1.2 Methodology

The research methodology consists of three major components:

Generalisation Framework - a detailed generalisation framework for deriving multiscale spatial data has been developed as a test bed based on a functionality assessment of existing generalisation systems. Generalisation techniques were applied over a test area in the city of Canberra, Australian Capital Territory, Australia in order to generalise 1:250,000 national topographic data to produce small scale maps. The assessment concerned a generalisation methodology to derive multiscale data sets using the principles of generalisation. The study paid particular attention to the integration and utilisation of generalisation operators in order to generalise a road network database and produce small scale maps at 1:500,000 and 1:1,000,000 from 1:250,000 national topographic data. There are a number of parameters that could be used for generalisation of roads to maintain legibility, the visual identity of each road segment and the pattern. Generalisation of geographical features (for example, roads, rivers, or other linear features) requires at least six key operations/processes: *Classification, Selection, Elimination, Simplification, Typification, and Symbolisation*. These are used in the generalisation framework in this study through an assessment of commercial software. The resultant maps match well with existing small-scale road maps such as the Global Map at scale of 1:1,000,000. The generalisation operations/algorithms, and their parameters embedded in a modern map generalisation system deliver coherent capabilities to automate the generalisation process for practical production applications.

Nevertheless, it is also apparent that the technology is still developing and requires an integration of generalisation algorithms, with a cartographer's intuition and skills within a GIS. This paves the way to develop a powerful, flexible and robust expert system capable of composing, editing, exhibiting and demonstrating a method for semi-automated generalisation of geographical features.

In addition, a framework for segmentation and generalisation of raster data '*Interactive Automated Segmentation and Raster Generalisation Framework*' (IASRGF) was developed. Test results of the IASGRF shows that all objects derived from the generalisation of land use data over Canberra, Australia were well classified and mapped. The error assessment indicates that the percentile classification accuracy is 85.5%, whereas the commission error is relatively high (38.5%). More importantly, the maximum likelihood classifier using training sites and associated ground truth data suggests that the Kappa index is 0.798, which can be interpreted as a reliable and satisfactory classification result. In order to further enhance supervised classification, a post-classification was carried out. As a result, this extra process improved the overall classification accuracy slightly, however the commission error also increased by 6% (**Appendix 5**).

Cartographic Knowledge-based Generalisation - this is achieved through an International Cartographic Generalisation Survey that collected inputs from several national mapping agencies, state mapping agencies and a number of software vendors. This included cognitive cartographers' knowledge about the principles of cartographic generalisation and experience with existing generalisation platforms. The findings from the survey were incorporated as a series of cartographic rules to propose and implement a knowledge-based generalisation solution.

Generalisation Expert System (GES) - acquired knowledge was then utilised to build a knowledge-based solution developed in the Java-Python-C programming environments for the delivery of generalised geographical features. Its capabilities were demonstrated in a case study through generalising several line and polyline databases over the study area in Canberra, Australia. The cartographic and GIS software communities will

benefit from this study through access to a set of tools, guidelines and protocols that incorporate a standardised cartographic generalisation methodology.

8.2 GENERALISATION EXPERT SYSTEM DEVELOPMENT

The automation of map generalisation requires an expert system approach. An expert system consists of four main components, including knowledge acquisition, inference engine, knowledge representation, and user interface. The 'knowledge acquisition' in the context of this study is a challenging matter. This was undertaken by applying a heuristic natural knowledge transfer from cartographers using a Cartographic Generalisation Survey. The survey takes advantage of national mapping agencies, state mapping agencies and a number of software vendors' inputs in relation to the knowledge acquisition process for cartographic practices, undertaking a cartographic generalisation survey capturing cartographers' knowledge about the principles of cartographic generalisation and experience with existing generalisation platforms. The collected cartographic knowledge and heuristic rules were organised as a series of rules for use in the GES software.

While obtaining representative samples in this survey has proven to be a difficult process, the survey revealed that the knowledge of cartographers is not being consistently communicated to generalisation software developers. Nor is that knowledge documented consistently across different mapping agencies. Out of the 75 agencies that expressed interest in participating in the survey, only a total of 26 responses were received from 15 agencies, representing a ≈ 20% response rate. The majority of respondents were from the NMAs and SMAs.

The aim was to reach a widespread agreement among cartographers in relation to specific knowledge about map and spatial data generalisation. The agreed methodological guidelines and procedures could then be incorporated into future software tools to make generalisation operations and algorithms more reliable.

Following this initial study, a GES was built in Java-Python-C, enabling semi-automated generalisation of lines and polylines. Its capabilities were established in a case study through simplifying roads, native vegetation and elevation datasets. The

results of the trials were analysed. A series of generalisation tests were performed to gauge the sensitivity of simplification results to the different spatial layers.

Cartometric measures, such as total length and number of line/polyline segments, were used as an index of generalisation to quantify generalisation performance for the target small scale. For example, there are 101,228 segments and 9,491 segments in 1:250,000 scale and in 1:500,000 scale contours respectively over the study area. This requires reduction in the complexity and the density of elevation data. Changes in the representation of contour features at 1:250,000 and 1:500,000 scales as a result of generalisation were quantified. The outputs from map derivation have been analysed applying the *Radical Law* that determines the retained number of objects for a given scale change and the number of objects of the source map. Results of map derivation using the *Radical Law* were addressed in the thesis (**Section 4.3**, **Section 5.4** and **Section 7.3**).

The tests demonstrated that the implemented algorithms in the GES were able to extract characteristic vertices on the original entity lines and polylines (e.g. for roads and native vegetation), while excluding non-characteristic ones so as to reduce insignificant computation and improve efficiency of line/polyline generalisation. This study has demonstrated reasonable improvements in the Vertex Reduction, Classification and Merge, Enhanced Douglas-Peucker algorithms; a finding similar to previous research.

The test results show that the GES generalises line features well and maintains their geometric relations. Existing generalisation software requires advanced technical skills from users, however, the GES has a comparatively basic and GUI that is an advantage for users with lower technical skills and understanding of spatial data management.

The geographical transformation that is continuously taking place should be updated in multiscale maps and spatial databases in ‘near real time’. The conventional methods apply to each individual map and therefore result in inconsistency between databases. GES is a trial tool for generalising large-scale maps into smaller scales, and creating maps of different themes across various scales. The cartographic and GIS software

communities could benefit from this study through access to a set of tools, guidelines and protocols that incorporate a standardised cartographic generalisation methodology.

8.3 KEY CONTRIBUTIONS

The thesis contributions are structured into three main areas, including theoretical, methodological and empirical findings resulting from the design, implementation and evaluation of a generalisation system. To support novelty regarding the contributions of this study, the author's work appeared in a number of peer-reviewed conference and journal papers. Considerable parts of those publications are referred to in this thesis. The contributions made by the key chapters are outlined in **Table 8.1**.

Table 8.1 Major contributions of the study

Element (s)	**Chapter (s)**
Development of a detailed generalisation framework to produce small scale maps through derivative mapping applying an assessment of existing generalisation systems - *theoretical and methodological*	**Chapters 2**, **4** and **5**
Building a framework for a heuristic natural knowledge elicitation and transfer from cartographers using an International Cartographic Generalisation Survey - *theoretical and methodological*	**Chapter 6**
Construction of GES for delivering semi-automated generalisation of lines and polylines. Results demonstrated that the implemented algorithms leading to an improvement of efficiency of generalisation - *empirical findings*	**Chapter 7**

8.4 REMARKS AND FUTURE DIRECTIONS

This research attempts to improve spatial data generalisation using a knowledge-based approach for the design of the next generation of semi-automated generalisation tools. Today, national mapping agencies and commercial providers of digital spatial databases encounter a wide range of problems in maintaining multiple databases; this situation is resource-intensive, time consuming and cumbersome (Arnold and Wright, 2005). Consequently there is a need to develop improved map generalisation interfaces to better support this important task. The study focused on development of a generalisation framework for deriving multiscale spatial data. This involved an assessment of existing

generalisation systems (ESRI ArcGIS™, Intergraph DynaGen™, and Laser-Scan Clarity™). As discussed in **Chapter 2**, previous research relating to this area has been limited. Although many studies of map generalisation have been carried out, few have considered integration of cartographic knowledge in the construction of a rule-based expert system.

Cartographic knowledge acquisition has been identified as a key bottleneck problem in relation to generalisation, but suggestions have been made to provide feasible strategies for their solution. Studies by the author show that knowledge acquisition within existing generalisation systems, e.g. generalisation of line and area features, has not been fully implemented. Therefore there was a lack of empirical foundation for bridging generalisation operations and cartographers' experience / intuition within existing generalisation systems. Consequently much of the development in this area has been technologically driven rather than embedding the cartographers' knowledge in generalisation systems. Despite the advancements, many of the innovative generalisation software tools have not been thoroughly evaluated (**Chapter 2**). This study (**Chapters 4** and **5**) empirically assessed the generalisation capabilities to derive a multiscale database from a master database in order to enhance the conceptual framework.

Promising directions for future work require development of theoretical and empirical foundations on multiple scale presentations and real time generalisation. Key recommendations include:

- It is appropriate to reiterate the importance of incorporation of cartographers' knowledge within the expert system design process. Without thorough evaluation of existing generalisation systems, it is not clear whether a new tool (design) is better or different. Regarding empirical studies, the author encourages generalisation system design along with clear protocols and user-friendly design for NMAs. Most of the experiments described in this thesis have taken place in a generalisation trial model. This development needs to be implemented in commercial software so that the tools become available to NMAs for further testing and evaluation; hence, repeatable and quantitative methods to evaluate generalisation results remains as a challenge.

- In order to introduce a comprehensive generalisation system into NMAs production processes an international approach is desirable. This is achievable through strong partnership among NMAs, academic institutions and vendors of GIS solutions since off-the-shelf software products often do not meet the requirements of individual NMAs. This direction for future research (as confirmed by Muller *et al.,* 1995) still appears valid.
- Smoothing of linear features is promising. Development of combined algorithms for line smoothing while preserving shape of linear features is worthwhile to consider. The smoothing algorithms (e.g. B-spline snakes, Gaussian filtering and Wavelets transformations) are recommended for turning the GES into commercial software. Further work would be required to enhance the GES by incorporating other algorithms, such as Snakes (see Guilbert and Saux, 2008; Saux, 2003; Agouris *et al.,* 2001; McKeown and Denlinger, 1988).
- Finally, future research should be directed towards development of web mapping platforms with generalisation functionality at varying scales. This recommendation is also supported by other previous studies (e.g. Lecordix *et al.,* 1997; Jones and Ware, 2005; Sarjakoski *et al.,* 2002; Ross *et al.,* 2007; Neun *et al.,* 2008). Undertaking research on real time generalisation would be worthwhile.

REFERENCES*

AGENT, C., 1999. Selection of Basic Algorithms. Technical Annex, AGENT Consortium, ESPRIT/LTR/24 939, University of Zurich. Zurich, Switzerland.

Agouris, P., A. Stefanidis, and S. Gyftakis, 2001. Differential snakes for change detection in road segments. *Photogrammetric Engineering and Remote Sensing, Vol. 67, No. 12, December 2001, pp. 1391–1399.*

Agrawala, M., and C. Stolte., 2001. Rendering effective route maps: Improving usability through generalisation. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, 28 October 2008, New York, USA pp.241-249.*

Almeida, C.M., J.M. Gleriani, E.F. Castejon, and B.S. Soares-Filho, 2008. Using neural networks and cellular automata for modelling intra-urban land-use dynamics. *International Journal of Geographical Information Science, Vol. 22, No. 9, pp. 943 - 963.*

Alves, M., 2010. Polygonal line simplifying methods applied to GIS. *Proceedings of the FIG Congress 2010. Facing the Challenges – Building the Capacity, 11-16 April 2010, Sydney, Australia, CD-ROM procs, 9pp.*

Anders, K.H., and M. Sester, 2000. Parameter-free cluster detection in spatial databases and its application to typification. *Proceedings of the ISPRS, Vol. XXXIII I, Part B4/1, Commission IV, 16-23 July 2000, Amsterdam, The Netherlands, pp. 75-82.*

* Cited references for books, technical reports, lecture notes, websites and unpublished materials appear as normal texts here. However, literatures (references) from journal publications and conference proceedings are shown in *italic* texts.

ANZLIC, 2001. ANZLIC Metadata Guidelines: Core Metadata Elements for Geographic Data in Australia", Version 2, ANZLIC Spatial Council, Canberra, Australia. Accessed online 25 June 2007: http://anzlic.org.au/asdi/metaelem.htm

Armstrong, M. P., 1991. Knowledge classification and organization. Map generalization. Ed. Barbara P. Buttenfield and Robert B. McMaster. Making Rules for Knowledge Representation, Longman Scientific & Technical, ISBN 0-582-08062-2.

Arnold, L.M., and G.L. Wright, 2005. Analysing product-specific behaviour to support process dependent updates in a dynamic spatial updating model. *Transactions in GIS, Vol. 9, No. 3, pp. 397-419.*

Baelia, B., M. Pla, and D. Lee, 1995. Digital map generalisation in practice. *Proceedings of the 17th International Cartographic Conference and 10th General Assembly: Cartography, Vol. 1, 3-9 September 1995, Barcelona, Spain, CD-ROM procs, 8pp.*

Bader, M., 2001. Energy minimisation methods for feature displacement in map generalisation. PhD Dissertation, Department of Geography, University of Zurich, Zurich, Germany.

Bakker, N. and B. Kolk, 2003. TOP10NL, a new object-oriented topographical database in GML. *Proceedings of the International Cartographic Conference*, 10-16 August, 2003, Durban, South Africa. Accessed online 25 June 2007: http://cartography.tuwien.ac.at/ica/documents/ICC_proceedings/ICC2003/Papers/103.pdf

Bakker, N.J., 1997. Generalisation in practice with topographical databases in the Netherlands. *Proceedings of the 2nd Workshop on Progress in Automated Map Generalisation, 19-21 June 1997, Gavle, Sweden, pp.56-67.*

Barr, S.L., 1992. Object-based reclassification of high resolution digital imagery for urban land-use monitoring. *Proceedings of the XVII Congress of International Archives*

of Photogrammetry and Remote Sensing, Commission VII, Vol. XXIX, 12-14 August 1992, Washington D.C., USA, pp. 969-976.

Barrault, M., 1995. An automated system for linear feature name placement which complies with cartographic quality criteria. *Proceedings of the ACSM/ASPRS Autocarto 12, February 27 - March 2, Charlotte, North Carolina, USA, pp. 321-330.*

Barrault, M., M. Bader and R. Weibel, 2000. Topology preserving conflict removal between symbolised roads in cartographic generalisation: Extending snakes methods. *Proceedings of the GIScience 2000, October 28-31 2000, Savannah, Georgia, USA, pp. 55-63.*

Bartlett, J.E., J.W. Kotrlik and V.C. Higgins, 2001. Organisational research: Determining sample size in survey research. *Information Technology, Learning, and Performance Journal, Vol. 19, No. 1, pp. 43-50.*

Bengston, M., 2001. Design and implementing of automatical generalisation in a New production environment for datasets in scale 1:50,000 and 1:100,000. *Proceedings of the Workshop of ICA Commission on Generalisation and Multiple Representation, 3-11 July 2001, Beijing, China, CD-ROM procs, 14pp.*

Bereuter, P., and R. Weibel., 2010. Generalisation of point data for mobile devices: A problem-oriented approach. *Proceedings of the 13th Workshop of the ICA Commission on Generalisation and Multiple Representation, 12-13 September 2010, Zurich, Switzerland, CD-ROM procs, 8pp.*

Bielawski, L., and R. Lewand, 1988. Expert Systems Development. Building PC-Based Applications. QED Information Sciences, Inc. Wellesley, Massachusetts, USA.

Bolstad, P.V., and M. Lillesand, 1991. Rapid maximum likelihood classification. *Photogrammetric Engineering and Remote Sensing, Vol. 57, No. 1, pp. 67-74.*

Boss, R.W., 1991. What Is an Expert System? ERIC Digest. ERIC Clearing House on Information Resources, Syracuse, New York, USA. Accessed online 20 May 2006: http://www.ericdigests.org/pre-9220/expert.htm

Bowerman, R.G., and D. Clover, 1988. Putting Expert Systems into Practice. Van Nostrand Reinhold Company, New York.

Brassel, K., and R. Weibel, 1988. A review and conceptual framework of automated map generalisation. *International Journal of Geographic Information Systems, Vol. 2, No. 3, pp. 229-244.*

Brophy, D.M., 1973. An automated methodology for linear generalisation in thematic cartography. *Proceeding of the American Congress of Surveying and Mapping Conference, 11 March 1973, Washington D.C., USA, pp. 300-314.*

Brooks, R., 2001. A seat of the pants displacement operator. *Dagstuhl Seminar 01191: Computational Cartography and Spatial Modelling, 6-11 May 2001, Wadern, Germany.* Accessed online 11 May 2006: http://www.rupertbrooks.ca/downloads/pres_displacement_dagstuhl.pdf

Brooks, R., 2000. Two birds with one stone: Constructing national framework data for the global map. *Proceedings of the Global Mapping Forum 2000, 28-30 November 2000, Hiroshima, Japan.* Accessed online 20 February 2006: http://www.iscgm.org/more-iscgm/forum2000/MrRupertBrooks.pdf

Brooks, R., and R. Evans, 1999. Feasibility of using automated generalisation techniques to process the national road network. Technical Report, GeoAccess Division, Canada Centre for Remote Sensing, Natural Resources Canada, Ottawa, Ontario, Canada, pp. 7-38.

Brown, M.H., and J. Hershberger, 1994. Third annual video review of computational geometry. Systems Research Centre's Research Report, California, USA. Accessed

online 20 December 2006: http://www.hpl.hp.com/techreports/Compaq-DEC/SRC-RR-101A.pdf

Burrough, P.A., 1986. Principles of Geographical Information Systems for Resources Assessment, Clarendon Press, Oxford, UK.

Buttenfield, B., 2002. Transmitting vector geospatial data across the internet. *Proceedings of the 2nd International Conference on Geographic Information Science, 25-28 September 2002, Boulder, Colorado, USA, pp. 51- 64.*

Carignan, M., and G. Dumoulin, 2002. Topographic mapping at the 1:100 000 scale in Quebec: Two techniques; One product. *Proceedings of the Geospatial Theory, Processing and Applications ISPRS Commission IV, 9-12 July 2002, Ottawa, Canada.* Accessed online 20 December 2006: http://www.isprs.org/proceedings/XXXIV/part4/pdfpapers/221.pdf

Carrico, M.A, J.C. Girard, and J.P., Jones, 1989. Building Knowledge Systems. McGraw-Hill Book Co., New York, USA.

Cecconi, A., R. Weibel, and M. Barrault, 2002. Improving automated generalisation for on demand web mapping by multiscale databases. *Proceedings of the Joint International Symposium of Geospatial Theory, Processing and Applications, 8-11 July 2002, Ottawa, Canada, CD-ROM procs, 9pp.*

Chaudhry, O., and W.A. Mackanessa, 2008. Automatic identification of urban settlement boundaries for multiple representation databases. *Computers, Environment and Urban Systems, Vol. 32, No. 2, March 2008, pp. 95-109.*

Chen, X., and F. Bai, 2001. Generalisation of three dimensional city maps. *Proceedings of the 20th International Cartography Conference, 6-10 August 2001, Beijing, China, pp. 2083-2091.*

Chen, L. and J. Du, 1999. Graphic matching based old cadastral map generalisation. *Proceedings of the Geoinformatics Beyond 2000: Trends in Geoinformatics Technology and Applications, 9-11 March 1999, Dehradun, India, pp. 40-45.*

Cheng, C., F. Niu, J. Cai, and Y. Zhu, 2008. Extensions of GAP-tree and its implementation based on a non-topological data model. *International Journal of Geographical Information Science, Vol. 22, No. 6, June 2008, pp. 657–673.*

Cherkassky, V., and F. Mulier, 1998. Learning from Data: Concepts, Theory, and Methods, Wiley, New York, USA.

Chybicka, I, A. Iwaniak, and O. Wieslaw, 2004. Generalisation of the topographic database to the vector map Level 2 - the components of the Polish national geographic information system. *Proceedings of the ICA Workshop on Generalisation and Multiple Representation, 20-21 August 2004, Leicester, UK, CD-ROM procs, 14pp.*

Cihlar, J., Q. Xiao, J. Chen, J. Beaubien, K. Fung, and R. Latifovic., 1998. Classification by progressive generalisation: A new automated methodology for remote sensing multichannel data. *International Journal of Remote Sensing, Vol. 19, No. 14, pp. 3685-2704.*

Cochran, W.G., 1977. Sampling Techniques, 3rd Edition, John Wiley and Sons, New York, USA.

Choi, Y., and Z., Li., 2001. Correlation of generalisation effects with thematic attributes of cartographic objects. Department of Land Surveying and Geo-Informatics, the Hong Kong Polytechnic University, Hong Kong, pp. 1-26. Accessed online 1 June 2007: http://www.geo.unizh.ch/ICA/docs/beijing2001/papers/choi_li_v1.pdf

Cromley, R.C., 1992. Digital Cartography. Prentice Hall, Englewood Cliffs, New Jersey, USA.

Daley, N., D.G. Goodenough, A.S. Bhogal, Q. Bradley, and Z. Yin, 1997. Comparing raster and object generalisation. *Proceedings of the IEEE Geoscience and Remote Sensing Symposium, 3-8 August 1997, Singapore, pp. 677-679.*

Davis, A.C., and A.H.F. Laender, 1999. Multiple representations in GIS: Materialisation through map generalisation, geometric and spatial data analysis operations. *Proceedings of 7th ACM International Symposium on Advances in Geographic Information Systems, 2-6 November 1999, Kansas City, Missouri, USA, pp. 60-65.*

De Ville, B., 1990. Applying statistical knowledge to database analysis and knowledge-base construction. *Proceedings of the 6th IEEE Conference on Artificial Intelligence Applications, IEEE Computer Society, 5-9 March 1990, Washington D.C., USA, pp. 30-36.*

Doihara, T., P. Wang and W. Lu, 2002. An adaptive lattice model and its application to map generalisation. *Proceedings of the International Symposium on Geospatial Theory, Processing and Application, 8-12 July 2002, Ottawa, Canada, CD-ROM procs, 10pp.* Accessed online 20 May 2006: www.isprs.org/commission4/proceedings02/pdfpapers/301.pdf

Domenikiotis, C., G.D. Lodwick, and G.L. Wright, 1995. Intelligent interpretation of SPOT data for extraction of a forest road network. *Cartography, Vol. 24, No. 2, pp. 47-55.*

Donald, M.N., 1967. Implications of non-response for the interpretation of mail questionnaire data. *Public Opinion Quarterly, Vol. 24, No. 1, pp. 99-114.*

Douglas, D.H., and T.K. Peucker, 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer, Vol. 10, No. 2, pp. 112-122.*

Duda, R., and P. Hart, 1973. Pattern Classification and Scene Analysis, Bayes Decision Theory. John Wiley and Sons, ISBN-10 0471223611.

Dutton, D., 1999. Scale, sinuosity and point selection in digital line generalisation. *Cartography and Geographic Information Science, Vol. 26, No. 1, pp. 33-53.*

Ebisch, K., 2002. A correction to the Douglas–Peucker line generalisation algorithm. *Computers and Geosciences, Vol. 28, No. 8, October 2002, pp. 995-997.*

Edwards, A.J., and W.A. Mackaness., 2000. Intelligent generalisation of urban road networks. *Proceedings of the GIS Research UK 2000 Conference, 5-7 April 2000, University of York, UK, pp. 81-85.*

Elias, B., 2002. Automatic generation of wayfinding maps. *Journal of Geospatial Engineering, Vol. 4, No. 2, pp. 125-134.*

Environmental Systems Research Institute (ESRI), 1992. Arc/Info Documentation (CD-ROM), ESRI Redlands, ESRI Press, Redlands, California, USA.

Everitt, B., S. Landau, and M. Leese, 2001. Cluster Analysis. Edward Arnold, London, UK.

Fairbairn, D.J., 1993. On the nature of cartographic text. *Cartographic Journal. Vol. 30, No. 2, pp. 104-111.*

Foody, G.M., 2002. Status of land covers classification accuracy assessment. *Remote Sensing and Environment, Vol. 80, pp. 185-201.*

Foote Retzer, K., 2003. Introduction to Survey Sampling. Survey Research Laboratory University of Illinois at Chicago. Accessed online 20 December 2006: http://www.srl.uic.edu/seminars/Spr03_UIUC/samplingS03.PDF

Forghani, A., B. Cechet, and K. Nadimpalli, 2007. Object-based classification of multi-sensor optical imagery to generate terrain surface roughness information for input to wind risk exposure simulation. *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, 23-27 July 2007, Barcelona, Spain,, pp. 3090 – 3095.*

Forghani, A., R. Reddy, and G. Smith, 2003. Evaluating SPOT-5 satellite imagery for national mapping division's topographic mapping programs. *Proceedings of the Spatial Sciences Conference 18–21 September 2003, Canberra, Australia, CD-ROM procs, 13pp.*

Forghani, A., J. Osborn, and M. Roach, 1997. An image interpretation model to support integration of image understanding techniques within a GIS: Delineation of road structures from aerial imagery. *Proceedings of the International GIS/GPS'97 Symposium, 15-19 September 1997, Istanbul, Turkey, CD-ROM procs, 10pp.*

Forghani, A., 2000a. Decision trees for mapping of roads from aerial photography employing a GIS-guided technique. *Proceedings of the 10th Australasian Remote Sensing and Photogrammetry Conference, 21-25 August 2000, Adelaide, Australia, CD-ROM procs, 12pp.*

Forghani, A., 2000b. Semi-automatic detection and enhancement of linear features to update GIS files. *Proceedings of the 6th ISPRS Congress, 16-23 July 2000, Amsterdam, The Netherlands, CD-ROM procs, 10pp.*

Forghani, A., 1997. A Knowledge-Based Approach to Mapping Roads from Aerial Imagery Using a GIS Database. PhD Dissertation, Department of Surveying and Spatial Information Science, the University of Tasmania, Hobart, Tasmania, November 1997, Australia, pp. 1.-299.

Forghani, A., 1994. A new technique for map revision and change detection using merged Landsat TM and SPOT data sets in an urban environment. *Asian-Pacific Remote Sensing Journal, Vol. 7, No. 1, July 1994, pp. 119-131.*

Forghani, A., 1993. Monitoring Change in Urban Areas Using Satellite Remote Sensing Data with Particular Reference to Developing Countries. M.Eng.Sc Thesis, School of Surveying, The University of New South Wales, Sydney, November 1993, Australia, pp. 1-125.

Geoscience Australia (GA), 1999. Topographic Data and Map Specifications (CD-ROM). Technical Specifications, Version 3.4, National Mapping Division, Geoscience Australia. Accessed online 5 December 2004: http://www.ga.gov.au/mapspecs/250k100k/section3

Geoscience Australia (GA), 2001. Helping the global community to resolve global problems through spatial information. *Proceedings of the AURISA 2001 - the 29th Annual Conference of AURISA, 19-23 November 2001, Melbourne, Australia, CD-ROM procs, 12pp.*

Gerke, M., and C. Heipke. 2008. Image-based quality assessment of road databases. *International Journal of Geographical Information Science, Vol. 22, No. 8, pp. 871 - 894.*

Giarrantano, J., and G. Riley, 1989. Expert Systems: Principles and Programming. PWS-Kent, Boston, Massachusetts, USA.

Goodchild, M.F., 1996. Generalisation, uncertainty, and error modelling. *Proceedings of the GIS/LIS 96, 19-21 November 1996, Denver, Colorado, pp. 765-774.*

Goodchild, M.F., and G.J. Hunter, 1997. A simple positional accuracy measure for linear features. *International Journal of Geographical Information Science, Vol. 11, No. 3, January 1997, pp. 299 – 306.*

Guilbert, E., and E. Saux., 2008. Cartographic generalisation of lines based on a B-spline snake model. *International Journal of Geographical Information Science, Vol. 22, No. 8, August 2008, pp. 847–870.*

Gurney, C.M., 1981. The use of contextual information to improve land cover classification of digital remotely sensed data. *International Journal of Remote Sensing, Vol. 2, No. 4, pp. 379-388.*

Gross, J., and J., Yellen, 1999. Graph Theory and its Application, CRC press, London.

Hagbert, E.C., 1968. Validity of questionnaire data: reported and observed attendance in an adult education program. *Public Opinion Quarterly, Vol. 25, pp. 453-456.*

Hardy, P., M.O. Briat, C. Eicher, and T. Kressmann, 2004. Database-driven cartography from a digital landscape model, with multiple representations and human overrides. *Proceedings of the ICA Workshop on Generalisation and Multiple Representation, 20-21 August 2004, Leicester, UK, CD-ROM procs, 12pp.*

Harrie, L., and T. Sarjakoski, 2002. Simultaneous graphic generalisation of vector data sets. *GeoInformatica, Vol. 6, No. 3, pp. 233-261.*

Harrie, L., 2001. An Optimisation Approach to Cartography Generalisation. Doctoral Book, Department of Surveying, Lund Institute of Technology, Lund University, Sweden.

Harrie, L., 2003. Weight – setting and quality assessment in simultaneous graphic generalisation. *The Cartographic Journal, Vol. 40, No. 3, pp. 221-233.*

Harrie, L., 1999. The constraint method for solving spatial conflicts in cartographic generalisation. *Cartography and Geographic Information System, Vol. 26, No. 1, pp. 55-69.*

Harris, P.M., and S.J. Ventura, 1995. The integration of remotely sensed imagery to improve classification in an urban area. *Photogrammetric Engineering and Remote Sensing, Vol. 61, No. 8, pp. 993-998.*

Hartigan, J., 1975. Clustering Algorithms. John Wiley & Sons Inc., New York, USA, ISBN 0-471-35645-X.

He, H.S., S.J. Ventura, and D.J. Mladenoff, 2002. Effects of spatial aggregation approaches on classified satellite imagery. *International Journal of Geographical Information Science, Vol. 16, No. 1, pp. 93-109.*

Hershberger, J., and J. Snoeyink, 1992. Speeding up the Douglas-Peucker line simplification algorithm. *Proceedings of the 5th Spatial Data Handling Symposium, IGU Commission on GIS, 3-7 August 1992, Charleston, South Carolina, USA, University of South Carolina pp. 134--143.*

Hohl, P., 1998. GIS Data Conversion. Strategies, Techniques, and Management. Santa Fe, New Mexico, Published by OnWord Press, USA. ISBN 1566901758.

Holland, P., 2005. Verbal Communication. Geoscience Australia, Canberra, Australia.

Holland, P., I. O'Donnell, and A. Nairn, 2003. The database revolution. *Position Magazine, April 2003, pp. 40-42.*

Holton, E.H., and M.B. Burnett, 1997. Qualitative Research Methods. In R. A. Swanson, and E. F. Holton (Eds.), Human Resource Development Research Handbook: Linking Research and Practice, Berrett-Koehler Publishers, San Francisco, California, USA. ISBN 978-1-57675-314-9.

Hope, S., A. Kealy, and G.J. Hunter, 2006. Improving positional accuracy and preserving topology through spatial data fusion. *Proceedings of the 7th International Symposium on Spatial Accuracy Assessment in Natural Resources and Environmental Sciences, 5-7 July 2006, Lisbon, Portugal, pp. 675-684.*

Hope, S., and G.J. Hunter, 2007. Testing the effects of positional uncertainty on spatial decision-making. *International Journal of Geographical Information Science, Vol. 21, No. 6, July 2007, pp. 645–665.*

Hunt, E.B., 1962. Concept Learning: An Information Processing Problem. John Wiley & Sons, New York, USA, ISBN 0-471-14890-3.

Hunter, G.J., and M.F. Goodchild, 1995. Dealing with error in spatial databases: A simple case study. *Photogrammetric Engineering and Remote Sensing, Vol. 61, No. 5, pp. 529-537.*

Hunter, G.J., and M.F. Goodchild, 1996. Communicating uncertainty in spatial databases. *Transactions in GIS, Vol. 1, No. 1, pp. 13–24.*

International Cartographic Association, (ICA) 1973. Multilingual Dictionary of Technical Terms in Cartography. Frank Steiner Verlag GMBH, Wiesbaden Germany. ISBN-10: 3598107641.

Intergraph, 2004. DynaGen Software Documentation (CD-ROM). P.O. Box 240000, Huntsville, AL 35813, USA.

Iwaniak, A., I. Chybicka, and W. Ostrowski, 2004. Generalisation of the topographic database to the vector map level 2. *Proceedings of the Workshop of ICA Commission on Generalisation and Multiple Representation, 20-21 August 2004, Leicester, UK, CD-ROM procs, 12pp.* Accessed online 11 September 2004: http://ica.ign.fr/Leicester/paper/Chybicka-v2-ICAWorkshop.pdf

Iwaniak, A., and W. Paluszynski, 2001. Generalisation of Topographic Maps of Urban Areas. *Proceedings of the 20th International Cartographic Conferences, 6-10 August 2001, Beijing, China, CD-ROM procs, 12pp.* Accessed online 19 July 2003: http://icaci.org/documents/ICC_proceedings/ICC2001/icc2001/author.htm

Jaakkola, O., 1998. Multiscale categorical databases with automatic generalisation transformations based on map algebra. *Cartography and Geographic Information Systems, Vol. 25, No. 4, pp. 195-207.*

Jenks, G.F., 1989. Geographic logic in line generalisation, *Cartographica, Vol. 26, No. I, pp. 27-42.*

Jiang, B., and C. Claramunt, 2004. Topological analysis of urban street networks. *Environment and Planning B: Planning and Design, Vol. 31, pp. 151-162.*

Joao, E.M., 1998. The Algorithmic Approach in Causes and Consequences of Map Generalisation, Taylor and Francis Ltd, London, ISBN: 9780849339325.

Johnson, P.O., 1994. Development of the sample survey as a scientific methodology. *Journal of Experiential Education, Vol. 27, pp. 167-176.*

Jones, C.B., G.L. Bundy, and J.M. Ware, 1995. Map generalisation with a triangulated data structure. *Cartography and Geographic Information Systems, Vol. 22, No. 4, pp. 317-331.*

Jones, C.B., and J.M. Ware, 2005. Map generalisation in the web age. International *Journal of Geographical Information Science, Vol. 19, Nos. 8-9, pp. 859-870.*

Jones, C.B., A.I. Abdelmoty, M.E. Lonergan, P. Van derpoorten, and S. Zhou, 2000. Multiscale spatial database design for online generalisation. *Proceedings of the Ninth Spatial Data Handling Symposium, Study Group on Geographical Information Science of the International Geographical Union (IGU), 10-12 August 2000, Beijing, China, pp. 34-44.*

Jones, C.R., R. Purves, A. Ruas, M. Sanderson, M. Sester, M.J. van Kreveld, and R. Weibel, 2002. Spatial information retrieval and geographical ontologies an overview of the SPIRIT project - SIGIR 2002. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 11-15 August 2002, Tampere, Finland, pp. 387-388.*

Jones, B., 1997. Geographic Information Systems and Computer Cartography. Addison Wesley Longman Limited, Essex, UK, ISBN-10: 0582044391.

Karimi, H.A., and G.D. Lodwick, 1987. A simple rule-based system for selection of remote sensing imagery. *Proceedings of Eleventh Canadian Symposium on Remote Sensing, 22-25 June 1987, Waterloo, Ontario, Vol. II, pp. 591-598.*

Karaak, M.J., and A. Brown, 2001. Web Cartography: Developments and Prospects. Taylor & Francis, London, ISBN 074840869X.

Kass, M., A. Witkin, and D. Terzopoulos, 1987. Snakes: Active contour models. *Proceedings of the International Conference on Computer Vision (ICCV), IEEE, 16 November 1987. London, UK, pp. 259–268.*

Kazemi, S., S. Lim, and H. Paik, 2009a. Generalisation expert system (GES): A knowledge-based approach for generalisation of line and polyline spatial datasets. *Proceedings of the Surveying & Spatial Sciences Institute Biennial International Conference, 28 September - 2 October 2009, Adelaide, Australia, pp. 717-729.*

Kazemi,' S., S. Lim, and C. Rizos, 2009b. Interactive and automated segmentation and generalisation of raster data. *International Journal of Geoinformatics, Vol 5, No. 3, pp. 65-73.*

Kazemi, S., and S. Lim, 2008. Development of a robust spatial engine for management of environmental data. *Proceedings of the 1st Conference on Environmental Engineering Systems, Planning and Management Engineering, 1-2 January 2008, Tehran, Iran, CD-ROM procs, 16pp.*

Kazemi, S., and S. Lim, 2007a. Deriving multiscale GEODATA from TOPO-250K road network data. *Journal of Spatial Science, Vol. 52, No. 1, pp. 171-182.*

Kazemi, S., and S. Lim, 2007b. Automated spatial data generalisation in support of disasters response. *Proceedings of the 2nd Conference on Geospatial Information Technology and Disaster Management, 25-26 December 2007, Tehran, Iran, , CD-ROM procs, 11pp.*

Kazemi, S., S. Lim, and L. Ge, 2007a. An international survey research: cartographic generalisation practices in mapping agencies. *Proceedings of the SSC 2007 Spatial Intelligence, Innovation and Praxis: The National Biennial Conference of the Spatial Sciences Institute, 4-8 May 2007, Hobart, Australia, pp. 537-556.*

Kazemi, S., S. Lim, and L. Ge, 2005b. An interactive segmentation and generalisation framework for mapping features from Landsat 7 imagery. *Proceeding of the First International Symposium on Cloud-prone & Rainy Areas Remote Sensing (1st CARRS), 4-8 October 2005, Hong Kong, pp. 1285- 1296.*

Kazemi, S., S. Lim, and L. Ge, 2005a. Integration of cartographic knowledge with generalisation algorithms. *Proceedings of the IEEE Geoscience & Remote Sensing Symposium, 25-29 July 2005, Seoul, South Korea, pp. 3502–3505.*

Kazemi, S., and S. Lim, 2005b. Generalisation of road network using Intergraph DynaGen system. *Proceedings of the SSC 2005 Spatial Intelligence, Innovation and Praxis: The National Biennial Conference of the Spatial Sciences Institute, 12-16 September 2005, Melbourne, Australia, pp. 1285-1296.*

Kazemi, S., S. Lim, and C. Rizos, 2004a. A review of map and spatial database generalisation for developing a generalisation framework. *Proceedings of the XIX Congress of 2004 the International Society for Photogrammetry and Remote Sensing, 12-23 July 2004, Istanbul, Turkey, pp. 1221-1226.*

Kazemi, S., S. Lim, and L. Ge, 2004b. Is automated generalisation there yet? *Proceedings of the 12th Australasian Remote Sensing and Photogrammetry Association Conference, 18-22 October 2004, Fremantle, Australia, CD-ROM procs, 12pp.*

Khoshnevis, B., and S. Parisay, 1993. Machine learning and simulation: application in queuing systems. *Simulation, Vol 61, No5, November, pp. 294-302.*

Kilpelainen, T., 1994. Updating multiple representation geodata bases by incremental generalisation. *Proceedings of the Spatial Information from Digital Photogrammetry and Computer Vision, ISPRS, 5-9 September 1994, Munich, Germany, pp. 440-447.*

Kilpelainen, T., 1997. Multiple Representation and Generalisation of Geo-databases for Topographic Maps. PhD Thesis, Publications of the Finnish Geodetic Institute, ISBN 978-951-22-8943-1.

Kilpelainen, T., 2000. Knowledge acquisition for generalisation rules. *Cartography and Geographical Information System, Vol. 27, No. 1, pp. 41-50.*

Kilpelainen, T., 2002. Solving space conflicts in map generalisation using a finite element method. *Cartography and Geographic Information Science, Vol. 27, No. 1, pp. 65-73.*

Koning, E.D., 2010. Polyline simplification: A generic C++ implementation for n-dimensional Douglas-Peucker Approximation. Accessed online 3 January 2011: http://www.codeproject.com/KB/recipes/PolylineSimplification.aspx#heading0002

Kreveld, M.J., and J. Peschier, 1998. On the automated generalisation of road network maps. *Proceedings of the 3rd International Conference on GeoComputation, 17 - 19 September 1998, University of Bristol, United Kingdom.* Accessed online 12 July 2006: *http://www.geocomputation.org/1998/21/gc_21.htm*

Kreveld, M.C., 2001. Smooth generalisation for continuous zooming. *Proceedings of the 20th International Cartographic Conferences,* 5 August 2001, Beijing, China, *pp. 2180–2185*

Krejcie, R.V., and D.W. Morgan, 1970. Determining sample size for research activities. *Educational and Psychological Measurement, Vol. 30, pp. 607-610.*

Lamy, S., A. Ruas, Y. Demazeau, C. Baeijs, M. Jackson, W. Mackaness, and R. Weibel, 1999. The application of agents in automated map generalisation. *Proceedings*

of the 19th International Cartographic Conference, 16-20 August 1999, Ottawa, Canada, pp. 1225–1234.

Landis, J.R., and G.G., Koch, 1977. The measurement of observer agreement for categorical data. *Biometrics, Vol. 33, pp. 159-174.*

Lang, T., 1969. Rules for robot draughtsmen. *Geographical Magazine, Vol. 42, pp. 50-51.*

Laurini, R., and D. Thompson, 1992. Fundamentals of Spatial Information Systems. Academic Press, London, UK, ISBN-13 9780124383807.

Lecordix, F., N. Regnauld, M. Meyer, and A. Fechir, 2005. MAGNET Consortium. *Proceedings of the 8th ICA Workshop on Generalisation and Multiple Representation, 7-8 July2005,, La Coruna , Spain Zurich, CD-ROM procs, 8pp.*

Lecordix, F., N. Plazanet, and J.P. Lagrange, 1997. A platform for research in generalisation: Application to caricature. *Geoinformatica, Vol. 1, No. 2, pp. 161–182.*

Lee, D., 1992. Cartographic Generalisation. Intergraph Corporation Press, USA.

Lee, D., 2002. Personal Communication. ESRI, Redlands, USA.

Lee, D., 2003. Generalisation within a geoprocessing framework. *Proceedings of the GEOPRO 2003 Workshop, 4-5 November 2003, , Mexico City, Mexico, pp. 82-91.*

Lee, D., 2004. Geographic and cartographic contexts in generalisation. *Proceedings of the ICA Workshop on Generalisation and Multiple Representation*, 20-21 August 2004, Leicester, UK. *Accessed online 1 March 2006:* http://ica.ign.fr/Leicester/paper/Lee-v2-ICAWorkshop.pdf

Lee, D., and P. Hardy., 2006. Design and experience of generalization tools. *Proceedings of AutoCarto 2006, June 25* 2006, Vancouver, Canada, *CD-ROM procs, 11pp.*

Leica Geosystems, 2004. ERDAS Field Guide (CD-ROM), December 2004, ERDAS, Inc, 5051 Peachtree Corners Circle, Suite 100 Norcross, GA 30092-2500, USA.

Lemarie, C., 2003, Generalisation process for Top100: Research in generalisation brought to fruition. *Proceedings of the 5th Workshop on Progress in Automated Map Generalisation, 28-30 April 2003, Paris, France, CD-ROM procs, 9pp.*

Li, R., 2000. Data Models for Marine and Coastal Geographic Information Systems. In Marine and Coastal Geographical Information Systems, edited by Wright D., and Barlett D., Taylor and Francis.

Limeng, L., and W., Lixin, 2001. Map generalisation from scale of 1:500,000 to 1:2,500,000. *Proceedings of the 20th International Cartographic Conferences, 5 August 2001, Beijing, China, CD-ROM procs, 8pp.*

Lonergan, M., and B. Jones, 2001. An interactive displacement method for conflict resolution in map generalisation. *Algorithmica, Vol. 30, pp. 287-301.*

Mackaness, W.A., A. Edwardes, and T. Urwin, 1998. Self evaluating generalisation algorithms for the derivation of multiscale products from UKBORDERS data of the census. *Proceedings of the 3rd International Conference on GeoComputation, 17 - 19 September 1998, Bristol, UK, GeoComputation CD-ROM. ISBN 0-9533477-0-2.*

Mackaness, W.A, R. Ruas, and L.T. Sarjakoski, 2007. A synopsis of generalisation operations. In Generalisation of Geographic Information: Cartographic Modelling and Applications. Edited by W. A. Mackaness (University of Edinburgh, Scotland, U.K), A. Ruas (IGN Laboratoire COGIT, St. Mande, France), L. T. Sarjakoski (Finnish Geodetic Institute, Masala, Finland), April 2007, Elsevier Science, ISBN 13: 978-0-08-045374-3.

Marshall, G., 1990. Advanced Students' Guide to Expert Systems. Heinemann Newnes, Oxford, UK.

Mason, S., 1995. Expert system-based design of close-range photogrammetric networks. *ISPRS Journal of Photogrammetry and Remote Sensing, Vol. 50, No. 5, pp. 13-25.*

McMaster, R.B., and K.S. Shea, 1988. Cartographic generalisation in a digital environment: a framework for implementation in a geographic information system. *Proceedings of the GIS/LIS'88, 29 November – 30 December 1988, Antonio, Texas, USA, pp.240-249.*

McMaster, R.B., and K.S. Shea, 1989. Cartographic generalisation in a digital environment: when and how to generalise. *Proceedings of the 9th International Symposium on Computer-Assisted Cartography, 2-7 April 1989, Baltimore, Maryland, USA, pp. 56-67.*

McMaster, R.B., 1986. A statistical analysis of mathematical measures for linear simplification: an empirical study. *The American Cartographer, Vol. 13, No. 2, pp.103-116.*

McMaster, R.B., 1989. The integration of simplification and smoothing algorithms in line generalisation. *Cartographica, Vol. 26, No. 1, pp. 101-121*.

McMaster, R.B., and K.S. Shea, 1992. Vector-based generalisation in digital cartography. Association of American Geographers, 1710 16th St NW, Washington, D.C, USA, pp. 71-98.

McMaster, R.B., 2001. Measurement in generalisation. *Proceedings of the 20th International Cartography Conference, 6-10 August 2001, Beijing, China, pp. 2082-2089.*

McKeown, D.M., W.A. Harvey, and J. McDermott, 1985. Rule-based interpretation of aerial imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 8, pp. 532-542.*

McKeown, D., and J. Denlinger, 1988. Cooperative methods for road tracking in aerial imagery. *Proceedings of Computer Vision and Pattern Recognition, 5–9 June 1998, Ann Arbor, Michigan, USA, pp. 662–672.*

McKeown, D., J. McMahill, and D. Caldwell, 1999. The use of spatial context in linear feature simplification. *GeoComputation, Digital Mapping Laboratory, Carnegie Mellon University Pittsburgh, PA 15213 U.S.A, CD-ROM procs, 8pp.*

Meng, L., 1997. An In-depth Investigation on the Long Lasting Problem of Cartographic Generalisation. Technical Report, May 1997, Swedish Armed Force, pp. 1-75.

Michie, D., D.J. Spiegelhalter, and C.C. Taylor, 1994. Machine Learning, Neural and Statistical Classification. Prentice Hall, Englewood Cliffs, N.J., USA, ISBN-10: 013106360X.

Miller, L.E., and K.L. Smith, 1983. Handling non-response issues. *Journal of Extension, Vol. 21, pp. 45-50.*

Mitchell, T.M., 1997. Machine Learning, McGraw-Hill, New York, USA, ISBN-10: 0070428077.

Moller-Jensen, L., 1990. Knowledge-based classification of an urban area using texture and context information in Landsat-TM imagery. *Photogrammetric Engineering and Remote Sensing, Vol. 56, No. 6, pp. 899-904.*

Monmonier, M.S., and R.B. McMaster, 1991. The sequential effects of the geometric operators in cartographic line generalisation. *Proceedings of the 4th World Conference on ICA*, 17-24 August 1991, Budapest, Hungary, *CD-ROM procs, 9pp.*

Moody, A., 1998. Using landscape spatial relationships to improve estimates of land cover area from coarse resolution remote sensing. *Remote Sensing of Environment, Vol. 64, pp. 202-220.*

Moody, A., and C.E., Woodcock, 1995. The influence of scale and the spatial characteristics of landscapes on land-cover mapping using remote sensing. *Landscape Ecology, Vol. 6, pp. 363*–379.

Moreno, M., M. Torres, S. Levachkine, and I. Fajard, 2002. The automatic generalisation of the multiscale geographic information. *Proceedings of the 5th AGILE Conference on Geographic Information Science (AGILE 2002), 25-27 April 2002, Palma de Mallorca, Spain.* Accessed online March 2005: http://gis.esri.com/library/userconf/proc02/pap0173/p0173.htm

Mroz, M., K. Berger, R. Becherer, and J. Packman, 1996. The Defence Mapping Agency (DMA) world vector shoreline plus (WVSPLUS) revision and conversion project. Accessed online April 2005: http://gis.esri.com/library/userconf/proc96/TO250/PAP237/P237.htm

Muller, J.C., R. Weibel, J.P. Lagrange, and F. Salge, 1995. Generalisation, state of the art and issues. In Muller, J.c., Lagrange, J.P., Weibel, R. (editors) GIS and Generalisation, Methodology and Practice, Taylor and Francis, Bristol, Uk, pp. 91-105.

Muller, J.C., J.P. Lagrange, and R. Weibel, Eds, 1995. GIS and Generalisation: Methodology and Practice. Editor(s): J-P Lagrange; R Weibel; J. Muller, University of Ruhr, GISDATA Series, 20 April 1995, Bochum, Germany, ISBN: 9780748403196..

Muller, J.C. and W. Zeshen, 1992. Area-patch generalisation: A competitive approach. *Cartographic Journal, Vol. 29, No. 2, pp. 137-144.*

Muller, J.C., 1991. Generalisation of spatial databases. *Geographic Information Systems, Vol. 1, pp. 457- 475.*

Muller, J.C., 1995. GIS and Generalisation: Methodology and Practice. Editor(s): J-P Lagrange; R Weibel; J. Muller, University of Ruhr, GISDATA Series. 20 April 1995, Bochum, Germany, ISBN: 9780748403196.

Nakos, B., 1999. Comparison of manual versus digital line generalisation. *Proceedings of the Workshop on Generalisation, 12-14 August 1999, Ottawa, Canada, CD-ROM procs, 10pp.*

Mustiere, S., J. Zucker, and L. Saitta, 2000. An abstraction-based machine learning approach to cartographic generalisation. *Proceedings of the 9th International Symposium on Spatial Data Handling, 10-12 August 2000, Beijing, China, CD-ROM procs, 10pp.*

Mustiere, S., J. Zucker, and L. Saitta, 1999. Cartographic generalisation as a combination of representing and abstracting knowledge. *Proceedings of the 7th ACM International Symposium on Advances in Geographic Information Systems, 5-6 November, Kansas City, Missouri, USA, pp. 162 – 164.*

Mustiere, S., and B. Moulin, 2002. What is spatial context in cartographic generalisation? *Proceedings of the International Symposium on GeoSpatioal Theory, Processing and Applications, 9-12 July 2002, Ottawa, Canada, CD-ROM procs, 12pp.*

Mustiere, S., 2005. Cartographic generalisation of roads in a local and adaptive approach: A knowledge acquisition problem. *International Journal of Geographical Information Science, Vol. 19, No. 29 September – 1 October 2005, pp. 937–955.*

Nakos, B., 1997. Fractal geometry theory in performing automated map generalisation operations. *Proceedings of the 2nd Workshop on Progress in Automated Map Generalisation, 5 June 1997, Gavle, Sweden, CD-ROM procs, 10pp.*

Nakos, B., 1999. Comparison of manual versus digital line generalisation. *Proceedings of the Workshop on Generalisation, 12-14 August 1999, Ottawa, Canada, CD-ROM procs, pp9.*

Neuffer, D., T. Hopewell, and P. Woodsford, 2004. Integration of agent-based generalisation with mainstream technologies and other system components. *Proceedings of the ICA Workshop on Generalisation and Multiple representation, 20-21 August 2004, Leicester, UK, CD-ROM procs, 11pp.*

Neun, M., D. Burghardt, and R. Weibel, 2008. Web service approaches for providing enriched data structures to generalisation operators. *International Journal of Geographical Information Science, Vol. 22, No. 2, pp. 133 - 165.*

Nikolopoulos, C., 1997. Expert Systems: Introduction to First and Second Generation and Hybrid Knowledge Base Systems, Marcel Dekker, Inc., New York, ISBN 10: 0824799275.

Nishijima, M., and T. Watanabe, 1997. A cooperative inference mechanism for extracting road information automatically. *Proceeding of the 3rd Asian Conference on Computer Vision, Volume II, 8-10 January 1998, Hong Kong, pp. 217-224.*

Oosterom, P.V., and V. Schenkelaars, 1995. The development of an interactive multiscale GIS. *International Journal of Geographical Information Systems, Vol. 9, No.5, pp. 489-507.*

Oosterom, P.V., 1995. The GAP-tree, an approach to 'on-the-fly' map generalisation of an area partitioning. The GAP-tree, an approach to "On-the-Fly" Map Generalization of an Area Partitioning, volume GIS and Generalization: Methodology and Practice of GISDATA, Chapter 9, Taylor & Francis, London, UK, pp. 120 - 132.

Oosterom, P.V., 2005, Variable-scale topological data structures suitable for progressive transfer: The GAP-face tree and GAP-edge forest. *Cartography and Geographic Information Science, Vol. 32, pp. 331–346.*

Paluszynski, W., and A. Iwaniak, 2001. Generalisation of topographic maps of urban area. *Proceedings of the 20th International Cartography Conference, 6-10 August 2001, Beijing, China, CD-ROM procs, 9pp.*

Peers, I., 1996. Statistical Analysis for Education and Psychology Researchers. Falmer Press, Bristol, UK.

Peter, B., 2001. Measures for the generalization of polygonal maps with categorical data. *Proceedings of the 4th ICA Workshop on progress in Automated Map Generalization, 2-4 August 2001, Beijing, China, CD-ROM procs, 10pp.*

Peter, B., and R. Weibel, 1999. Using vector and raster techniques in categorical map generalisation. *Proceedings of the 3rd ICA Workshop on Progress in Automated Map Generalisation, 12-14 August, Ottawa, Canada, CD-ROM procs, 8pp.*

Petit, C.C., and E.F. Lambin, 2001. Integration of multi-source remote sensing data for land cover change detection. *International Journal Geographical Information Science. Vol. 15, No.8, pp. 785-803.*

Pfer, T., and W. Pillewizer, 1966. The principles of selection. *The Cartographic Journal, Vol. 3, No. 1, pp. 10-16.*

Provost, F., and V. Kolluri, 1999. A survey of methods for scaling up inductive algorithms. *Data Mining and Knowledge Discovery, Vol. 2, pp. 131–169.*

Quinlan, J.R., 1983. Learning efficient classification procedures and their application to chess and games. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (ed), Machine Learning: An Artificial Intelligence Approach, Tioga Publishing Co. PaIo Alto, California, USA.

Quinlan, J.R., 1986. Induction of decision trees. *Machine Learning, Vol. 1, pp. 81-106.*

Quinlan, J.R., 1990. Learning logical definition from relations. *Machine Learning, Vol. 5, pp. 239-266.*

Radke, J., 2007. Lecture Notes - Geographic Information Systems. Department of Landscape Architecture and Environmental Planning / Department of Geography, University of California Berkeley, California, USA, pp. 1-50.

Radke, J. 1995. Modeling urban/wildland interface fire hazards within a geographic information system. *Geographic Information Sciences, Vol. 1, No 1, pp, 7-20.*

Raisz, E. 1962. Principles of Cartography. McGraw-Hill, New York, USA.

Rdenas, S.N., L. Wang, and F.B. Zhan., 2009. Representing geographical objects with scale-induced indeterminate boundaries: A neural network-based data model. *International Journal of Geographical Information Science, Vol. 23, No. 3, March 2009, pp. 295–318.*

Regnauld, N., 1996. Recognition of building cluster for generalisation. *Proceedings of the 6th Spatial Data Handling, 15 August 1996, Delft, The Netherlands, pp. 185- 198.*

Reumann, K., and A.P.M. Witkam, 1974. Optimizing cure segmentation in computer graphics. *International Computing Symposium. North-Holland Publishing Company, Amsterdam*, The Netherlands, *pp. 467-472.*

Richardson, D., and J. Muller, 1991. Towards rule-based selection of spatial objects for small scale map generalisation. In B. Buttenfield and R. McMaster R, Editors of Map Generalisation: Making Rules for Knowledge Representation, Longman, Essex, UK, pp. 136–149.

Richardson, D., 1989. Rule based generalisation for map production. *Proceeding of the National Conference GIS - Challenge for the 1990s, 2 January 1989, Ottawa, Canada, pp. 718-39.*

Richards, J.A., 1986. Remote Sensing Digital Image Analysis: An Introduction. Springer-Verlag, Berlin, Germany.

Robinson, A.H., R.D. Sale, J.L. Morrison, and P.C. Muehrcke, 1984. Elements of Cartography. John Wiley and Sons, New York, USA.

Ross, S., Purves, R.S, P. Clough, C.B. Jones, A. Arampatzis, B. Bucher, D. Finch, and G. 2007. The design and implementation of SPIRIT: A spatially aware search engine for

information retrieval on the Internet. *International Journal of Geographical Information Science, Vol. 21, No. 7, pp. 717-745.*

Fu, G., H. Joho, A. Khirni Syed, S. Vaid, and B. Yang, 2007. The design and implementation of SPIRIT: a spatially aware search engine for information retrieval on the Internet. *International Journal of Geographical Information Science, Volume 21, Number 7, pp.717-745.*

Ruas, A., and C. Plazanet, 1996. Strategies for automated generalisation. *Proceedings of the 7th Spatial Data Handling Symposium, 2-4 June 1996, Delft, The Netherlands, Vol. 2, pp. 61-67.*

Ruas, A., 1995. Multiple Representation and Generalisation. Lecture Notes, pp41. Accessed online 20 July 1996: http://www.dpi.inpe.br/teses/miro/chapter2.pdf

Ruas, A., 2001. Automating the generalisation of geographical data: The age of maturity? Computer generalisation of spatial data. *Proceedings of the 20th International Cartography Conference, 6-10 August 2001, Beijing, China, pp. 1943-1953.*

Ruas, A., 1998. A method for building displacement in automated map generalisation. *International Journal of Geographical Information Science, Vol. 12, No. 8, pp. 789-803.*

Rusak, E., and H.W. Castner, 1990. Horton's ordering scheme and the generalisation of river networks. *Cartographic Journal, Vol. 27, pp. 104-110.*

Sarjakoski, T., L. Lehto, M. Sester, A. Illert, F. Nissen, B. Rystedt, and R. Ruotsalainen, 2002. Geospatial info-mobility services: A challenge for national mapping agencies. *Proceedings of the International Symposium on Geospatial Theory, Processing and Applications, 9-12 July 2002, Ottawa, Canada, CD-ROM procs, 5pp.*

Saux, E., 2003. B-spline functions and wavelets for cartographic line generalisation. *Cartography and Geographic Information Science, Vol. 30, No. 8, pp. 847-870.*

Sester, M., 2005. Optimisation approaches for generalisation and data abstraction. *International Journal of Geographical Information Science, Vol. 19, No. 8, pp. 871 - 897.*

Sester, M., 2000. Generalisation based on least squares adjustment. *Proceedings of the XIX ISPRS Congress, 16-22 July 2000, Amsterdam, The Netherlands, pp. 931-938.*

Shea, K.S., and R.B. McMaster, 1989. Cartographic generalisation in a digital environment: When and how to generalise. *Proceedings of the Auto-Carto 9, 2-7 April 1989, Baltimore, USA, pp. 56-67.*

Shea, K.S., 1991. Design considerations for an artificially intelligent system. In B.P. Buttenfeild and R.B. McMaster (Eds), Map Generalisation, Making Rules for Knowledge for Representation, London, Longman Scientific and Technical, *pp. 3-20.*

Skopeliti, A., and L. Tsoulos, 2001. A knowledge based approach for the generalisation of liner features. *Proceedings of the 20th International Cartography Conference, 6-12 August 2001, Beijing, China, CD-ROM procs, 10pp.*

Smith, T.R., D Peuquet, S. Menon, and P. Agarwal, 1987. KBGIS-II: A knowledge-based geographical information system. *International Journal of Geographical Information Systems Vol. 1, pp. 149-72.*

Smith, C.J.H., L. Wang, and N. Higgs, 2002. The refinement of Landsat 7 passes of the Australian continent with accurate ground control points. *Proceedings of the 11th ARSPC Conference, 2-6 September 2002, Brisbane, Australia, CD-ROM procs, 9pp.*

Smith, T.W., 1983. The hidden 25 percent: An analysis of non-response on the 1980 General Social Survey. *Public Opinion Quarterly, Vol. 47, pp. 386-404.*

Stoter, J.E., M.J. Karrak, and R.R. Knippers, 2004. Generalisation of framework data: A research agenda. *Proceedings of the ICA Workshop on Generalisation and Multiple Representation, 20-21 August 2004, Leicester, UK.* Accessed online 4 April 2005:

http://ica.ign.fr/bibliography/display_text.php?id_text=50

Sunday, D., 2003. Fast polygon area and Newell normal computation. *Journal of Graphics Tools, Vol. 7, No. 2, pp. 9-12.*

Tapiador, F.J., 2008. Management Tools: Geographical Information Systems (GIS) and Expert Systems, Rural Analysis and Management, Springer Berlin Heidelberg.

Taylor, D.R.F., 2007. Challenge for the industry is brainware! GIS Development. Interview December 2007. Accessed online 10 January 2008:
http://www.gisdevelopment.net/interview/previous/ev0123tayler.htm

Taylor, D.R.F., 2005. Cybercartography: Theory and Practice, Taylor, Fraser, Elsevier, Amsterdam, The Netherlands.

Taylor, D.R.F., 2003. The Concept of Cybercartography. In M. Peterson (Editor), Maps and the Internet, Elsevier Science Ltd, Cambridge, UK.

Thomson, R.C., and D.E. Richardson, 1999. The good continuation – Principle of perceptual organisation applied to the generalisation of road networks. *Proceedings of the ICA 19th International Cartographic Conference, 14-21 August 1999, Ottawa, Canada, pp. 1215–1223.*

Thomson, R.C., and R. Brooks, 2002. Exploiting perceptual grouping for map analysis, understanding and generalisation: The case of road and river networks. Lecture Notes in Computer Science (LNCS), Volume 2390, pp. 148-157.

Tobler, W.R., 1966. Medieval distortions: The projections of ancient maps. *The Association of American Geographers, Vol. 56, No. 2, pp. 51–60.*

Ton, J., J. Sticken, and A. Jain, 1991. Knowledge-based segmentation of Landsat images. *IEEE Transactions on Geoscience and Remote Sensing, Vol. 29, No. 2, pp. 222-232.*

Topfer, F., and W. Pillewizer, 1966. The principles of selection, A means of cartographic generalisation. *Cartographic Journal, Vol. 3, No. 1, pp. 10-16.*

Tryfona, N., and M.J. Egenhofer, 1996. Multi resolution spatial databases: Consistency among networks. *Proceedings of the 6th International Workshop on Foundations of Models and Languages for Data and Objects, 16-20 September 1996, Universitat Magdeburg, Germany, pp. 119-132.*

Vaughan, J.R., and G.R. Brookes, 1989. The Mandelbrot set as a parallel processing benchmark. *Computing, Vol. 11, No. 3, pp. 193 - 197.*

Vaughan, J., D. Whyatt, and G.A. Brooks, 1991. Parallel implementation of Douglas-Pecuker line simplification algorithm. *Software Practice and Experience, Vol. 21, No. 3, pp. 331-336.*

Visvalingam, M., and J.D. Whyatt, 1991. The Douglas-Peucker algorithm for line simplification re-evaluation through visualisation. *Computer Graphics Forum, Vol. 9, No. 3, pp. 213 - 228.*

Visvalingam, M., and J.D. Whyatt, 1993. Line generalisation by repeated elimination of points. *The Cartographic Journal, Vol. 30, No. 1, pp. 46-51.*

Visvalingam, M., and P.J. Williamson, 1995. Simplification and generalisation of large scale data for roads - A comparison of two filtering algorithms. *Cartography and Geographic Information Systems, Vol. 22, No. 4, pp. 264-275.*

Visvalingam, M., and S. Herbert, 1999. A computer science perspective on the bend simplification algorithm. *Cartography and Geographic Information Science, Vol. 26, pp. 253-270.*

Visvalingam, M., 1999. Deconstruction of fractals and its implications for cartographic education. *The Cartographic Journal, Vol. 36, No. 1, pp. 15-29.*

Waele, J., and P. Maeyer, 2001. A quantitative comparison of legends between Belgium and neighbouring countries as a starting point for an objective generalisation. *Proceedings of the 20th International Cartography Conference, 6-10 August 2001, Beijing, China, pp. 2082-2090.*

Walker, P.A., and D.M. Moore, 1988. SIMPLE: An inductive modelling and mapping system for spatially oriented data. *International Journal of Geographical Information Systems, Vol. 2, pp. 347-63.*

Walter, V., 2004. Object-based classification of remote sensing data for change detection. *ISPRS Journal of Photogrammetry and Remote Sensing, Volume 58, Issues 3-4, January 2004, pp 225-238.*

Wang, P.T., T. Doihara, and W. Lu, 2002. Spatial generalisation: An adaptive lattice model based on spatial resolution. *Proceedings of the Geospatial Theory, Processing and Applications ISPRS Commission IV, 9-12 July 2002, Ottawa, Canada.* Accessed online 20 December 2006:
http://www.isprs.org/proceedings/XXXIV/part4/pdfpapers/221.pdf

Wang, F., and R. Newkirk, 1988. A knowledge-based system for highway network extraction. *IEEE Transactions on Geoscience and Remote Sensing, Vol. 26, No. 5, pp. 525-531.*

Wang, Z., and J.C. Muller, 1998. Line generalisation based on analysis of shape. *Cartography and Geographic Information Systems, Vol. 25, pp. 3-15.*

Wang, L., P.S. Wayne, P. Gong, S.B. Gregory, 2004. Comparison of Ikonos and Quickbird images for mapping mangrove species on the Caribbean Coast of Panama. *Remote Sensing of Environment, Vol. 91, pp. 432-440.*

Wang, A., and J.C. Muller, 1993. Complex coastline generalisation. *Cartography and Geographic Information Systems, Vol. 20, No. 2, pp. 96-106.*

Warmerdam, F., 1999. Shapefile C Library V1.2. Accessed online 25 December 2006: www.shapelib.maptools.org

Watson, P., and V. Smith, 2004. Interoperability of agent-based generalisation with open, geospatial clients. *Proceedings of the ICA Workshop on Generalisation and Multiple Representation, 20-21 August 2004, Leicester, UK, CD-ROM procs, 5pp.*

Watson, L., 1997. Applying Case-based Reasoning, Morgan Kaufmann Publishers, San Francisco, California.

Weibel, R., S. Keller, and T. Reicheenbacher, 1995. Overcoming the knowledge acquisition in map generalisation: The role of interactive systems and computational intelligence. In A. Frank & W. Kuhn (Eds.), Spatial Information Theory, Springer, Berlin, Germany , pp. 139-156.

Weibel, R., 1995. Special content: Map generalisation. GIS and Generalization - Methodology and Practice, Guest Editor, Taylor & Francis, pp. 306-316.

Weibel, R., 1991. Amplified intelligence and rule-based system. In B.P. Buttenfield and R.B. McMaster (Eds). Map Generalisation: Making Rules for Knowledge Representation. London, Longman Scientific and Technical, pp. 172-186.

Weibel, R., 1996. Generalisation of spatial data: Principles and selected algorithms. Algorithmic Foundations of Geographic Information Systems. In Kreveld M van, Nievergelt J, Roos T, Widmayer P (eds), Chapter 5, pp. 99-152.

Weibel, R., and C.B. Jones, 1998. Computational perspectives on map generalisation. *GeoInformatica, Vol. 2, No. 4, pp. 307-314.*

Weibel, R., and G. Dutton, 1998. Constraint-based automated map generalisation. *Proceedings of the 8th International Symposium on Spatial Data Handling, 9 July 1998, Vancouver, Canada, pp. 214-224.*

Weibel, E.R., and G. Dutton 1999. Generalising spatial data and dealing with multiple representations. Geographical Information Systems, Vol. I, Principles and Technical Issues, edited by P. A Longley, M. F. Goodchild, D. J. Maguire, and D. W. Rhind, pp. 125-155.

Welch, R., 1985. Cartographic potential of SPOT image data. *Photogrammetric Engineering and Remote Sensing, Vol. 51, No. 8, pp. 1085-1091.*

Wenxiu, A., G. Jianya, and L. Zhilin, 2004. Knowledge-based generalisation on land-use data. *Proceedings of the 5th Generalisation Workshop, 28-30 April 2004, Paris, France, CD-ROM procs, 8pp.*

Wessel, P., and W.H.F. Smith, 1996. A global self-consistent, hierarchical, and high-resolution shoreline database. *Journal of Geophysical Research, Vol. 101, pp. 8741–8743.*

Whyatt, J.D., 1990. The Douglas-Peucker algorithm for line simplification re-evaluation through visualisation. *Computer Graphics Forum, Vol. 9, No. 3, pp. 213-228.*

Whyatt, J.D., and P.R. Wade, 1988. The Douglas-Peucker line simplification algorithm. *Bulletin of the Society of University Cartographers, Vol. 22, No. 1, pp. 17 – 25.*

Worldatlas, 2008. How Many Countries? Accessed online October 2008: http://www.worldatlas.com/nations.htm

Wright, D.J., and M.F. Goodchild, 1997. Data from deepsea: Implications for the GIS community. *International Journal of Geographical Information Systems, Vol. 11, No. 6, pp. 523-528.*

Wright, D.J., 2000. Spatial data infrastructures for coastal environments. Lecture Notes in Geoinformation and Cartography, *Springer Berlin Heidelberg, pp. 91-112.*

Wunsch, D., 1986. Survey research: Determining sample size and representative response. *Business Education Forum, Vol. 40, No. 5, pp. 31-34.*

Xu, R., and D. Wunsch, 2005. Survey of clustering algorithms. *IEEE Transaction on Neural Networks, Vol. 16, No. 3, pp. 645-648.*

Yang, W., and C.M. Gold, 1997. A system approach to automated map generalisation. *Proceedings of the International Workshop on Dynamic and Multi-Dimensional GIS, (Eds, Lee, Y. C. and Li, Z. L.), 25-26. August 1997, Hong Kong, pp. 229-235.*

Yang, W., and C.M. Gold, 2004. A system approach to map generalisation. *Proceedings of the ICA the Workshop on Generalisation and Multiple Representation, 20-21 August 2004, Leicester, UK, CD-ROM procs, 10pp.*

Yang, X., 2007. Integrated use of remote sensing and geographic information systems in riparian vegetation delineation and mapping. *International Journal of Remote Sensing, Vol. 28, pp. 353-370.*

Yaolin, L., M. Moenaar, and A. Tinghua, 2001. Frameworks for generalisation constraints and operations based on object oriented data structure in database generalisation. *Journal of Geo-Spatial Information Science, Vol. 4, No. 3, pp. 42-49.*

Yates, F., 1981. Sampling Methods for Censuses and Surveys, 4th Edition, Griffin, London, pp 1-254.

Zeiler, M., 2001. Exploring ArcObjects, Vol. II – Geographic Data Management ESRI Press, California, USA.

Zeiler, M., 1999. Modelling Our World - The ESRI Guide to Geodatabase Design. ESRI Press, Redlands, California, USA, ISBN 1-879102-06-1.

Zhao, H., X. Li, and L. Jiang, 2001. A modified Douglas-Pecuker simplification algorithm. *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, 9-13 July 2001, Sydney, Australia, CD-ROM procs, 4pp.*

Zhou, X., S.B. Prasher, and M. Kitsuregawa, 2002. Database support for spatial generalisation for WWW and mobile applications. *Proceedings of the 3rd International Conference on Web Information Systems Engineering, 12-14 December 2002, Singapore, pp. 239 - 246.*

Zikmund, W.G, 2000. Mail Questionnaires vs. Telephone Interview. Exploring Marketing Research, Wiley & Sons Ltd, Burgin, KY, USA, ISBN 0030229634.

APPENDIX 1: Knowledge Acquisition Questionnaire

GENERALISATION QUESTIONS

QUESTION 1:

(a) What proportion of your time (in percent) is spent on generalisation of spatial data in your daily work? (tick one)

10-20 percent 20-40 percent 40-60 percent 60-80 percent 80-100 percent

(b) How would you rate your competency in generalisation theory? (tick one)

Good Satisfactory Needs Improvement Mediocre

(c) How would you rate your competency in generalisation tools (e.g. ArcGIS™, ERDAS IMAGINE™, Intergraph™, and Laser-Scan™)? (tick one)

Extensive Reasonable Moderate Limited No/Minimal Requirement

(d) How would you rate your understanding of emerging technologies in generalisation (e.g. on the fly database generalisation, mobile mapping, object oriented technology)? (tick one)

Extensive Reasonable Moderate Limited

QUESTION 2: This question relates to your knowledge/experience in cartographic and database generalisation: Have you had significant formal training in modern generalisation technology?

Do you have a good grounding in the principles of database modelling in the context of generalisation?

What types of training have you completed, including on-the-job training, and academic training (e.g. university)? Have you had significant experience in a specific area of spatial data generalisation?

(b) How many years of experience do you have in using/applying cartographic and database generalisation? (tick one) 2 or Less 3 - 5 6 - 10 10 or More

QUESTION 3: Thinking about the transfer of map and database generalisation knowledge within your agency/company, how would you rate the expertise in generalisation there?

Good\ Satisfactory Needs Improvement Mediocre

QUESTION 4: What generalisation software do you think you will be using in the future and why (e.g. ArcGIS™, ERDAS IMAGINE™, Intergraph™, LaserScan™, CHANGE™, etc)?

SPECIALISED GENERALISATION QUESTIONS

QUESTION 5: The problem of deriving a range of map scales from a common database is considered an interesting and challenging area of research and development in today's map production environment. Technological developments in dynamic cartographic generalisation applications (e.g. mobile mapping, vehicle navigation) call for a robust, practical and cost-effective way of fulfilling automated generalisation needs. Do you think there is a need for intensive experiments with the existing generalisation tools (ESRI ArcGIS™, Intergraph DynaGen™, and Laser-Scan Clarity™) developed in the industry rather than developing new generalisation systems?

QUESTION 6: Researchers believe that combining the cartographers' knowledge with different generalisation algorithms would lead to achieving cartographically acceptable generalised results. From your perspective, how important is it to incorporate cartographers' knowledge into the generalisation process?

QUESTION 7: How important is map/spatial data generalisation to your project (tick one)?

Essential/Critical Very Important Important Slightly Important

QUESTION 8: As there are no standard/universal guidelines or workflows for applying generalisation operations and algorithm selection to maps and geodata, how are these operations undertaken in your agency? (Generalisation operations include Selection, Elimination, Simplification, Aggregation, Collapse, Typification, Conflict Resolution (Displacement), Smoothing, Refinement, Classification, and Symbolisation. For a detailed description of these operations, refer to Kazemi *et al.*, 2004b)

QUESTION 9: Based on your experience, which simplification and smoothing algorithms perform better for generalising linear features (e.g. roads, rivers)? Why? (Some algorithms are: Douglas-Peucker simplification, Brophy smoothing, Lang, Nth Point, Point Relaxation, Reumann-Witkam, VectGen)

QUESTION 10: To generalise a road network, which of the following operations would you use? (Tick all that apply)

Classification Selection Elimination Simplification Typification Symbolisation

QUESTION 11: How would you handle selection of an appropriate/optimal tolerance when simplifying lines using simplification algorithms such as Douglas-Peucker?

QUESTION 12: Often, generalisation algorithms ignore the role of human cognitive knowledge. Discovery of cartographers' knowledge is an integral part of building an expert system to solve generalisation problems such as feature displacement and resolving conflicts among features when generalising line (e.g. roads) and polygon features (e.g. buildings footprint). Do you use cartographic rules, guidelines, workflows when undertaking generalisation? If so, can you please share the rules to apply for generalisation of linear features such as roads?

Examples of generalisation rules are: (a) contours never intersect; (b) water bodies located in the bottom of valleys, (c) roads can crossing each other, and (d) symbology, as well as colouring of land use data, is best displayed with colour-hue for visual perspective.

QUESTION 13: How would you evaluate accuracy (qualitative and quantitative) of the output of the generalisation process; for example, when deriving road features at 1:500,000 and 1:1,000,000 scales from the source scale at 1:250,000? How important is that assessment to you?

QUESTION 14:

(a) In your opinion, what emerging generalisation research and development area will have the greatest impact on the future direction of automated generalisation technology?

(b) Is there any specific topic area that you feel universities and spatial information industry should be pursuing for future research and development?

QUESTION 15:

Please rate how important automated generalisation is to the overall success of your organisation (tick one):

Essential Very important Not important Insignificant No opinion

QUESTION 16: How satisfied are you with current overall performance of generalisation tools (tick one):

Very satisfied Satisfied Not Important Insignificant No opinion

QUESTION 17: What is the most effective generalisation framework to derive various types of maps at different scales, e.g. at scales ranging from 1:250,000 to 1:10,000,000?

QUESTION 18: Please add any additional comments/suggestions relating to map and spatial database generalisation within your agency/company:

Thank you for your assistance in completing this survey.

If you have any queries regarding this survey, please contact Sharon Kazemi via E-mail: s.kazemi@student.unsw.edu.au

CONTACT DETAILS (optional)

Name:

Agency:

Position:

E-mail:

Phone: Fax:

APPENDIX 2: Supplementary Information for GES

GCP ID	Road Class	Error 1:500,000 scale (m)	Error 1:1million scale (m)
195	1	174	164
199	1	53	46
215	1	6	11
221	1	213	226
223	1	232	235
2	2	44	36
60	2	122	124
77	2	150	150
110	2	52	47
113	2	48	59
117	2	28	39
121	2	62	63
122	2	56	64
126	2	76	79
127	2	172	176
130	2	60	68
143	2	37	32
201	2	39	34
211	2	45	40
27	3	676	41
28	3	93	22
92	3	90	88
97	3	107	103
98	3	76	75
100	3	139	137
103	3	353	356
104	3	87	90
107	3	181	192
108	3	51	49
139	3	69	60
140	3	288	277
142	3	35	45
151	3	275	268
153	3	19	39
154	3	62	86
161	3	178	175
182	3	93	94
183	3	99	87
184	3	232	225
185	3	493	490
193	3	105	88
207	3	60	49
217	3	411	409
222	3	349	346
226	3	122	118
228	3	85	91
42	4	100	101
225	4	59	24

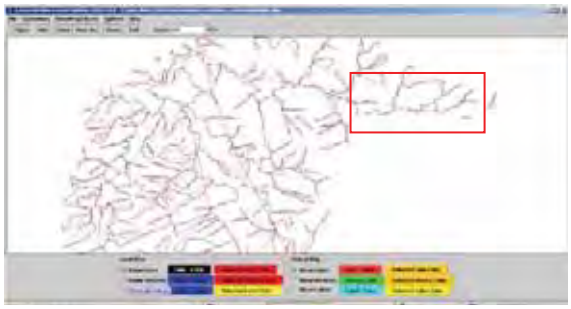
APPENDIX 3: Supplementary Information for GES Rule Sets

Rule 1 watercourse line (vertices)

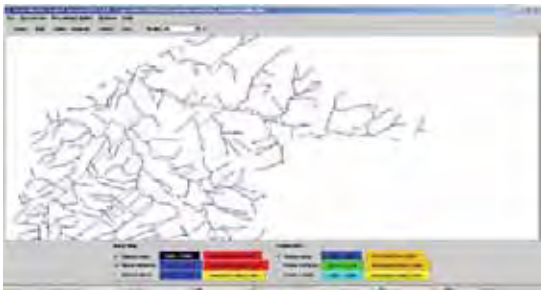
Rule 1 watercourse line (without vertices)

Rule 2 watercourse line (without vertices)

Rule 2 watercourse line (vertices)



Rule 2 watercourse line (without vertices) zoom



Rule1 watercourse line (without vertices) zoom

Rule 1 (Mexico state_sample data)

Rule 1: (Mexico state sample data) vertices

Rule 2 (Mexico state sample data)

Rule 2 (Mexico state sample data) vertices

Figure A.3.1. Supplementary information about testing GES rule sets

APPENDIX 4: Source Codes for GES

APPENDIX 4: Source Codes for GES (Generalisation Expert Systems)


```
/*
**************************************************************************
 *  Command.java: all commands executed via GUI
 *
 *       Saturday, 02 June 2007
 *
**************************************************************************
 */
package jges.gui;

import java.io.*;
import java.awt.*;
import javax.swing.*;
import javax.swing.filechooser.*;
import javax.swing.colorchooser.*;

import jges.util.*;


/** A public class defines all commands called via the button menu **/
public class Command
{
   // public static final String[] GesRules = {"ges_rule1.dat", "ges_rule2.dat",
"ges_rule3.dat", "ges_rule4.dat"};
          protected GesGUI gui;
          private static MapWindow window;
          protected String controlPanePosition;
          protected JPanel controlPane;

          // 4 file choosers have been set so that each has its own current directory
   protected JFileChooser chooserOpen,chooserLoad, chooserImport, chooserSave;
          protected static final String[] fileExtn = {"shp", "dbf"};
          protected static final String fileDesc = "Shape files, xBase files";
          // private GISFileFilter gisFileFilter;

   protected int ruleFileNo;

          /** Constructor **/
          public Command(GesGUI gui, MapWindow window)
          {
                    this.gui = gui;
                    this.window = window;
```

```
                ruleFileNo = 1;  // default is rule 1
                controlPanePosition = BorderLayout.SOUTH;
                // gisFileFilter = null;
        }

        /** Open a file and display its contents for text viewing and editing **/
        public void open()
        {
      if (window.inputMap != null && !window.inputMap.isEmpty())
      {
        String message = "Input map is not empty.\nPlease \"Close\" all opened maps
before opening new one.";
        JOptionPane.showMessageDialog(this.gui, message);
        return ;
      }

                if (chooserOpen == null)
                {
                        chooserOpen = new JFileChooser ();
                        chooserOpen.setCurrentDirectory(new File("../data/"));

        chooserOpen.setFileSelectionMode(JFileChooser.FILES_AND_DIRECTORI
ES);
                        // chooserOpen.setAcceptAllFileFilterUsed(true); // All Files
(*.*)
                        chooserOpen.setFileFilter(new GISFileFilter(fileExtn,
fileDesc));
                }

                int retval = chooserOpen.showOpenDialog(gui);
                if(retval==JOptionPane.NO_OPTION ||
retval==JOptionPane.CANCEL_OPTION)
                {
                        System.out.println ("Opening operation cancelled.");
                        return;
                }

                File openFile = chooserOpen.getSelectedFile();
                if (openFile != null && openFile.isFile())
                {
                        // remember previous directory
                        chooserOpen.setCurrentDirectory(new
File(openFile.getParent()));
                        window.loadData(openFile.getPath());
                }
        }

   public void append()
   {
      if (window.inputMap == null || window.inputMap.isEmpty())
```

```
      {
         String message = "No map has been opened.\nPlease \"Open\" a map before
appending new ones to it.";
         JOptionPane.showMessageDialog(this.gui, message);
         return ;
      }

                  if (chooserOpen == null)
                  {
                        chooserOpen = new JFileChooser ();
                        chooserOpen.setCurrentDirectory(new File("../data/"));

         chooserOpen.setFileSelectionMode(JFileChooser.FILES_AND_DIRECTORI
ES);
                        // chooserOpen.setAcceptAllFileFilterUsed(true); // All Files
(*.*)
                        chooserOpen.setFileFilter(new GISFileFilter(fileExtn,
fileDesc));
                  }

                  int retval = chooserOpen.showOpenDialog(gui);
                  if(retval==JOptionPane.NO_OPTION ||
retval==JOptionPane.CANCEL_OPTION)
                  {
                        System.out.println ("Opening operation cancelled.");
                        return;
                  }

                  File openFile = chooserOpen.getSelectedFile();
                  if (openFile != null && openFile.isFile())
                  {
                        // remember previous directory
                        chooserOpen.setCurrentDirectory(new
File(openFile.getParent()));
                        window.appendData(openFile.getPath());
                  }
   }

   public void run()
   {
      if (window.inputMap == null || window.inputMap.isEmpty())
      {
         String message = "No map has been opened.\nPlease \"Open\" a map before
running the GES processing.";
         JOptionPane.showMessageDialog(this.gui, message);
         return ;
      }
                  if (window != null)
                        window.runAlgorithm (ruleFileNo);
   }
```

```
// ../bin/shpParser.exe C:\Documents and Settings\Family
Folder\ges\data\roads\1million-beforesimplyfy.shp > ../tmp/1million-
beforesimplyfy_shp.out
	/** save text to the original file **/
	public void save ()
	{
		if (window != null)
			window.saveData();
	}

	/** save text or modeling results into a new file with specific type **/
	public void saveAs()
	{
		if (window == null)
			return ;
		else if (chooserSave == null)
		{
			chooserSave = new JFileChooser ();
			chooserSave.setCurrentDirectory(new File("./"));

	chooserSave.setFileSelectionMode(JFileChooser.FILES_AND_DIRECTORI
ES);
			chooserSave.setAcceptAllFileFilterUsed(true); // All Files (*.*)
		}

		chooserSave.setFileFilter(new GISFileFilter(fileExtn, fileDesc));
		chooserSave.setDialogType(JFileChooser.SAVE_DIALOG);
		chooserSave.setDialogTitle("Save As CVS File Format");
		int retval = chooserSave.showDialog(gui, "Save");
		if(retval==JOptionPane.NO_OPTION ||
retval==JOptionPane.CANCEL_OPTION)
		{
			System.out.println("\"Save As\" operation cancelled.");
			return;
		}

		File saveFile = chooserSave.getSelectedFile();
		if (saveFile != null)
		{
			// remember previous directory
			chooserSave.setCurrentDirectory(new
File(saveFile.getParent()));
			window.saveData();
		}
	}

	public void closeInputMap ()
	{
		if (window != null)
```

```
                {
        gui.getControlPanel().resetDefaultButtonColors();
                        window.clear();
                }
        }

        /** Close the GUI window and then exit "GES" **/
        public void exitGES()
        {
                System.out.println ("Conforming exiting of the Generalisation Expert
System ...");
                int answer = JOptionPane.showConfirmDialog (gui, "Do you want to"
                        + " exit the Generalisation Expert System ?", "Exit",
JOptionPane.YES_NO_CANCEL_OPTION);

      if(answer == JOptionPane.YES_OPTION)
                {
                        if (window != null)
                                window.clear();
                        gui.dispose ();
                        gui = null;
                        System.gc ();   // release all allocated memories before exiting

                        // wait 3 seconds before exit, otherwise system may collaps
                        Thread thread = new Thread();
                        try {
                                thread.sleep(3000);
                        }
                        catch (InterruptedException e)
                        {
                                System.out.println("Exception caused while exiting
4DTHERM "+e.toString());
                        }

                        System.exit (0);
                }
                else
                        System.out.println ("Exiting of GES cancelled.");
        }

        public void setRuleFile(int ruleNo)
        {
                ruleFileNo = ruleNo;
        }

        // Part 6: Options commands ---------------------------------
        // change the background color for current model
        public void setBackground ()
        {
                if (window == null)
```

```
                return;
            System.out.println ("Changing background color ...");
            Color currentColor = window.getBackgroundColor();
            Color newColor = JColorChooser.showDialog(gui, "Change
Background Color", currentColor);
            window.setBackgroundColor (newColor);
            System.out.println ("Background color changed.");
        }

        // change the foreground color for current model
        public void setForeground ()
        {
            if (window == null)
                return;
            System.out.println ("Changing foreground color");
            Color currentColor = window.getForegroundColor();
            Color newColor = JColorChooser.showDialog(gui, "Change
Foreground Color", currentColor);
            window.setForegroundColor (newColor);
            System.out.println ("Foreground color changed");
        }

        // change the control panel color for current model
        public void setControlPanelColor ()
        {
            if (window == null)
                return;

            Color currentColor = gui.getControlPanel().getBackground();
            Color newColor = JColorChooser.showDialog(gui, "Change Control
Panel Color", currentColor);
            gui.setControlPanelColor (newColor);
            System.out.println ("Control panel color changed");
        }

        public void showPoints (boolean selected)
        {
            if (window == null)
                return;
/*          window.setPointsShowable (selected);
            if (selected)
                System.out.println ("Show points ...");
            else
                System.out.println ("Close points panel ...");
*/      }

        public void showScales (boolean selected)
        {
            if (window == null)
                return;
```

```
                window.setScaleShowable (selected);
                if (selected)
                        System.out.println ("Show scale ...");
                else
                        System.out.println ("Close scale panel ...");
        }


        public void showDataLocation (boolean selected)
        {
                if (window == null)
                        return;
/*              window.setLocationShowable (selected);
                if (selected)
                        System.out.println ("Show data location ...");
                else
                        System.out.println ("Close data location panel ...");
*/      }

        public void showControlPanel (boolean selected)
        {
                if (gui == null)
                        return;
                else
                        gui.setControlPanelVisible (selected);
        }

        // Part 8: Help commands ---------------------------------
        public void showAbout ( )
        {
                String info = "Generalisation Expert System version 1.0 Release 1\n";
                info += "Copyright 2007 - \n";
                info += "All Rights Reserved\n\n";

                info += "GES 1.0 is a small software application implemented in Java
(for GUI), ";
      info += "Python (for various line simplification/generalization algorithms) ";
      info += "and C Language (for using C Shape/Dbf library.\n";

                info += "System Requirements\n";
                info += "-----------------------------------------------------------------\n";
                info += " Hardware                Minimum         Desired\n";
                info += "-----------------------------------------------------------------\n";
                info += " RAM                       10 MB           256 MB \n";
                info += "       Screen Resolution   1024×768       1400×1050\n";
                info += "       Color                 16 bit            32 bit \n";
                info += "-----------------------------------------------------------------\n";

                JOptionPane.showMessageDialog(gui, info,
                        "About GES",JOptionPane.INFORMATION_MESSAGE);
```

```
                System.out.println ("Show \"About GES 1.0\"");
        }
}

============================================================
package jges.gui;
============================================================

package jges.gui;

import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.border.TitledBorder;

import jges.com.*;
// import jges.util.*;


public class ControlPanel extends JPanel implements ActionListener
{
        private JPanel inPanel, outPanel;
        private JPanel inMapPanel, inColorPanel, inSCPanel;
   private JPanel outMapPanel, outColorPanel, outSCPanel;

        private JCheckBox showInputPointBox, showInputTextBox,
showInputLinesBox;
        private JCheckBox showOutputPointBox, showOutputTextBox,
showOutputLineBox;

   // A set of buttons for changing colors
        private JButton inColorPointButton, inColorTextButton, inColorLineButton;
        private JButton inSCPointButton, inSCTextButton, inSCLineButton;  // SC =
Selected color
        private JButton outColorPointButton, outColorTextButton,
outColorLineButton;
        private JButton outSCPointButton, outSCTextButton, outSCLineButton;  //
SC = Selected color

        public Color panelColor;
        private MapWindow window;


        public ControlPanel (MapWindow window)
        {
                this.window = window;
                panelColor = window.getGESGui().getBackground();          // Control
panel use initial GUI background color
```

```
                createControlPanel();
        }

        // Create the control panel
        private void createControlPanel ()
        {
                // 1. options for drawing lines/points/labels for the input map
                inMapPanel = new JPanel();
                inMapPanel.setLayout (new BoxLayout(inMapPanel,
BoxLayout.Y_AXIS));

                showInputLinesBox = new JCheckBox ("Show Lines", true);
                showInputLinesBox.addActionListener(this);
                showInputLinesBox.setToolTipText("Show all lines");
                inMapPanel.add(showInputLinesBox);

                showInputPointBox = new JCheckBox ("Show Vertices", false);
                showInputPointBox.addActionListener(this);
                showInputPointBox.setToolTipText("Show all vertices");
                inMapPanel.add(showInputPointBox);

                showInputTextBox = new JCheckBox ("Show Labels", false);
                showInputTextBox.addActionListener(this);
                showInputTextBox.setToolTipText("Show the labels for all
locations");
                inMapPanel.add(showInputTextBox);

                // 2. Buttons for changing colors for the input map
                inColorPanel = new JPanel();
                inColorPanel.setLayout (new BoxLayout(inColorPanel,
BoxLayout.Y_AXIS));

                inColorLineButton = new JButton ("Line    Color");
                inColorLineButton.addActionListener(this);
                inColorLineButton.setToolTipText("Click to change the line color");
                inColorPanel.add(inColorLineButton);

                inColorPointButton = new JButton ("Vertex Color");
                inColorPointButton.addActionListener(this);
                inColorPointButton.setToolTipText("Click to change the vertex
color");
                inColorPanel.add(inColorPointButton);

                inColorTextButton = new JButton ("Label   Color");
                inColorTextButton.addActionListener(this);
                inColorTextButton.setToolTipText("Click to change the color of all
labels");
                inColorPanel.add(inColorTextButton);

                // 3. Buttons for changing highlight colors for the input map
```

```
                inSCPanel = new JPanel();
                inSCPanel.setLayout (new BoxLayout(inSCPanel,
BoxLayout.Y_AXIS));

                inSCLineButton = new JButton ("Selected Line Color");
                inSCLineButton.addActionListener(this);
                inSCLineButton.setToolTipText("Click to change the highlight color
for selected lines");
                inSCPanel.add(inSCLineButton);

                inSCPointButton = new JButton ("Selected Vertex Color");
                inSCPointButton.addActionListener(this);
                inSCPointButton.setToolTipText("Click to change the highlight color
for selected vertex");
                inSCPanel.add(inSCPointButton);

                inSCTextButton = new JButton ("Selected Label Color");
                inSCTextButton.addActionListener(this);
                inSCTextButton.setToolTipText("Click to change the highlight color
for selected labels");
                inSCPanel.add(inSCTextButton);

                // 4. options for drawing the output map
                outMapPanel = new JPanel();
                outMapPanel.setLayout (new BoxLayout(outMapPanel,
BoxLayout.Y_AXIS));

                showOutputLineBox = new JCheckBox ("Show Lines", true);
                showOutputLineBox.addActionListener(this);
                showOutputLineBox.setToolTipText("Show all lines");
                outMapPanel.add(showOutputLineBox);

                showOutputPointBox = new JCheckBox ("Show Vertices", false);
                showOutputPointBox.addActionListener(this);
                showOutputPointBox.setToolTipText("Show all vertices");
                outMapPanel.add(showOutputPointBox);

                showOutputTextBox = new JCheckBox ("Show Labels", false);
                showOutputTextBox.addActionListener(this);
                showOutputTextBox.setToolTipText("Show labels for all locations");
                outMapPanel.add(showOutputTextBox);

                // 5. Buttons for changing colors for lines, points and labels of the
output map
                outColorPanel = new JPanel();
                outColorPanel.setLayout (new BoxLayout(outColorPanel,
BoxLayout.Y_AXIS));

                outColorLineButton = new JButton ("Line    Color");
                outColorLineButton.addActionListener(this);
```

```
                outColorLineButton.setToolTipText("Click to change line color");
                outColorPanel.add(outColorLineButton);

                outColorPointButton = new JButton ("Vertex Color");
                outColorPointButton.addActionListener(this);
                outColorPointButton.setToolTipText("Click to change vertex color");
                outColorPanel.add(outColorPointButton);

                outColorTextButton = new JButton ("Label   Color");
                outColorTextButton.addActionListener(this);
                outColorTextButton.setToolTipText("Click to change Label color");
                outColorPanel.add(outColorTextButton);

                // 6. Buttons for changing highlight colors for selected lines, points and
labels of the output map
                outSCPanel = new JPanel();
                outSCPanel.setLayout (new BoxLayout(outSCPanel,
BoxLayout.Y_AXIS));

                outSCLineButton = new JButton ("Selected Line Color");
                outSCLineButton.addActionListener(this);
                outSCLineButton.setToolTipText("Click to change the highlight color
for selected lines");
                outSCPanel.add(outSCLineButton);

                outSCPointButton = new JButton ("Selected Vertex Color");
                outSCPointButton.addActionListener(this);
                outSCPointButton.setToolTipText("Click to change the highlight color
for selected vertices");
                outSCPanel.add(outSCPointButton);

                outSCTextButton = new JButton ("Selected Label Color");
                outSCTextButton.addActionListener(this);
                outSCTextButton.setToolTipText("Click to change the highlight color
for selected labels");
                outSCPanel.add(outSCTextButton);

        // Pack all panels together
        inPanel = new JPanel();
                inPanel.setBorder( new TitledBorder("Input Map") );
                inPanel.setLayout (new BoxLayout(inPanel, BoxLayout.X_AXIS));
                inPanel.add(inMapPanel);
                inPanel.add(inColorPanel);
                inPanel.add(inSCPanel);

        outPanel = new JPanel();
                outPanel.setBorder( new TitledBorder("Output Map") );
                outPanel.setLayout (new BoxLayout(outPanel, BoxLayout.X_AXIS));
                outPanel.add(outMapPanel);
                outPanel.add(outColorPanel);
```

```
            outPanel.add(outSCPanel);

            this.add(inPanel);
            this.add(outPanel);
        }

        // change the background color for control panel and and all other components
    public void setPanelColor (Color color)
        {
            if (color != null && panelColor != color)
            {
                    panelColor = color;
        this.setBackground(panelColor);

        inPanel.setBackground(panelColor);
        outPanel.setBackground(panelColor);

        inMapPanel.setBackground(panelColor);
        inColorPanel.setBackground(panelColor);
        inSCPanel.setBackground(panelColor);

        showInputPointBox.setBackground(panelColor);
        showInputTextBox.setBackground(panelColor);
        showInputLinesBox.setBackground(panelColor);

        // If input map is empty, use panel color for all color buttons, otherwise use
their onw colors
        if (window.inputMap == null || window.inputMap.isEmpty())
        {
           inColorPointButton.setBackground(panelColor);
           inColorTextButton.setBackground(panelColor);
           inColorLineButton.setBackground(panelColor);
           inSCPointButton.setBackground(panelColor);
           inSCTextButton.setBackground(panelColor);
           inSCLineButton.setBackground(panelColor);
        }

        outMapPanel.setBackground(panelColor);
        outColorPanel.setBackground(panelColor);
        outSCPanel.setBackground(panelColor);

        showOutputPointBox.setBackground(panelColor);
        showOutputTextBox.setBackground(panelColor);
        showOutputLineBox.setBackground(panelColor);

        // If output map is empty, use panel color for all color buttons, otherwise use
their onw colors
        if (window.outMap == null || window.outMap.isEmpty())
        {
           outColorPointButton.setBackground(panelColor);
```

```
            outColorTextButton.setBackground(panelColor);
            outColorLineButton.setBackground(panelColor);
            outSCPointButton.setBackground(panelColor);
            outSCTextButton.setBackground(panelColor);
            outSCLineButton.setBackground(panelColor);
        }
        update ();
                }
        }

        // return the whole panel's background color
public Color getBackground ()
        {
                return panelColor;
        }

// Reset all button colors to the default panel color
        public void resetDefaultButtonColors ()
{
    inColorLineButton.setBackground(panelColor);
    if (inColorLineButton.getForeground() != Color.black)
        inColorLineButton.setForeground(Color.black);

    inColorPointButton.setBackground(panelColor);
    if (inColorPointButton.getForeground() != Color.black)
        inColorPointButton.setForeground(Color.black);

    inColorTextButton.setBackground(panelColor);
    if (inColorTextButton.getForeground() != Color.black)
        inColorTextButton.setForeground(Color.black);

    inSCLineButton.setBackground(panelColor);
    if (inSCLineButton.getForeground() != Color.black)
        inSCLineButton.setForeground(Color.black);

    inSCPointButton.setBackground(panelColor);
    if (inSCPointButton.getForeground() != Color.black)
        inSCPointButton.setForeground(Color.black);

    inSCTextButton.setBackground(panelColor);
    if (inSCTextButton.getForeground() != Color.black)
        inSCTextButton.setForeground(Color.black);

    outColorLineButton.setBackground(panelColor);
    if (outColorLineButton.getForeground() != Color.black)
        outColorLineButton.setForeground(Color.black);

    outColorPointButton.setBackground(panelColor);
    if (outColorPointButton.getForeground() != Color.black)
        outColorPointButton.setForeground(Color.black);
```

```
    outColorTextButton.setBackground(panelColor);
    if (outColorTextButton.getForeground() != Color.black)
        outColorTextButton.setForeground(Color.black);

    outSCLineButton.setBackground(panelColor);
    if (outSCLineButton.getForeground() != Color.black)
        outSCLineButton.setForeground(Color.black);

    outSCPointButton.setBackground(panelColor);
    if (outSCPointButton.getForeground() != Color.black)
        outSCPointButton.setForeground(Color.black);

    outSCTextButton.setBackground(panelColor);
    if (outSCTextButton.getForeground() != Color.black)
        outSCTextButton.setForeground(Color.black);
}

// Reset all button colors to the default panel color
    public void setButtonColorsForInputMap (GesMap map)
{
    if (map == null)
        return;

    inColorLineButton.setBackground(map.getLineColor());
    if (isBlackColor(map.getLineColor()))
        inColorLineButton.setForeground(Color.white);
    else
        inColorLineButton.setForeground(Color.black);

    inColorPointButton.setBackground(map.getPointColor());
    if (isBlackColor(map.getPointColor()))
        inColorPointButton.setForeground(Color.white);
    else
        inColorPointButton.setForeground(Color.black);

    inColorTextButton.setBackground(map.getLabelColor());
    if (isBlackColor(map.getLabelColor()))
        inColorTextButton.setForeground(Color.white);
    else
        inColorTextButton.setForeground(Color.black);

    inSCLineButton.setBackground(map.getSelectedLineColor());
    if (isBlackColor(map.getSelectedLineColor()))
        inSCLineButton.setForeground(Color.white);
    else
        inSCLineButton.setForeground(Color.black);

    inSCPointButton.setBackground(map.getSelectedPointOutlineColor());
    if (isBlackColor(map.getSelectedPointOutlineColor()))
```

```
        inSCPointButton.setForeground(Color.white);
    else
        inSCPointButton.setForeground(Color.black);

    inSCTextButton.setBackground(map.getSelectedLabelColor());
    if (isBlackColor(map.getSelectedLabelColor()))
        inSCTextButton.setForeground(Color.white);
    else
        inSCTextButton.setForeground(Color.black);
}

// Reset all button colors to the default panel color
        public void setButtonColorsForOutputMap (GesMap map)
{
    if (map == null)
        return;

    outColorLineButton.setBackground(map.getLineColor());
    if (isBlackColor(map.getLineColor()))
        outColorLineButton.setForeground(Color.white);
    else
        outColorLineButton.setForeground(Color.black);

    outColorPointButton.setBackground(map.getPointColor());
    if (isBlackColor(map.getPointColor()))
        outColorPointButton.setForeground(Color.white);
    else
        outColorPointButton.setForeground(Color.black);

    outColorTextButton.setBackground(map.getLabelColor());
    if (isBlackColor(map.getLabelColor()))
        outColorTextButton.setForeground(Color.white);
    else
        outColorTextButton.setForeground(Color.black);

    outSCLineButton.setBackground(map.getSelectedLineColor());
    if (isBlackColor(map.getSelectedLineColor()))
        outSCLineButton.setForeground(Color.white);
    else
        outSCLineButton.setForeground(Color.black);

    outSCPointButton.setBackground(map.getSelectedPointOutlineColor());
    if (isBlackColor(map.getSelectedPointOutlineColor()))
        outSCPointButton.setForeground(Color.white);
    else
        outSCPointButton.setForeground(Color.black);

    outSCTextButton.setBackground(map.getSelectedLabelColor());
    if (isBlackColor(map.getSelectedLabelColor()))
        outSCTextButton.setForeground(Color.white);
```

```
        else
            outSCTextButton.setForeground(Color.black);
    }

        // ------------------------------------------------------------
        // implement action listerner interface for GUI components
        // ------------------------------------------------------------
        public void actionPerformed(ActionEvent event)
    {
        Object target = event.getSource();

                    if (target == showInputLinesBox)
                    {
                            GesMap inputMap = window.getInMap();
                            if (inputMap != null)
                                    inputMap.setShowLinesFlag
(showInputLinesBox.isSelected ());
                    }
                    else if (target == showInputPointBox)
                    {
                            GesMap inputMap = window.getInMap();
                            if (inputMap != null)
                                    inputMap.setShowPointsFlag
(showInputPointBox.isSelected ());
                    }
                    else if (target == showInputTextBox)
                    {
                            GesMap inputMap = window.getInMap();
                            if (inputMap != null)
                                    inputMap.setShowTextFlag
(showInputTextBox.isSelected ());
                    }
                    else if (target == inColorLineButton)
                    {
                            GesMap inputMap = window.getInMap();
            if (inputMap != null)
            {
               Color oldColor = inputMap.getLineColor();
               Color newColor = JColorChooser.showDialog(null, "Change Line Color for
Input Map", oldColor);
               if (newColor != null)
               {
                  inputMap.setLineColor (newColor);
                  inColorLineButton.setBackground (newColor);

                  // To avoid both background and label colors are all black
                  if (isBlackColor(newColor))
                     inColorLineButton.setForeground(Color.white);
                  else
                     inColorLineButton.setForeground(Color.black);
```

```
            }
        }
                }
                if (target == inColorPointButton)
                {
                        GesMap inputMap = window.getInMap();
        if (inputMap != null)
        {
            Color oldColor = inputMap.getPointColor();
            Color newColor = JColorChooser.showDialog(null, "Change Point Color
for Input Map", oldColor);
            if (newColor != null)
            {
                inputMap.setPointColor (newColor);
                inColorPointButton.setBackground (newColor);

                // To avoid both background and label colors are all black
                if (isBlackColor(newColor))
                    inColorPointButton.setForeground(Color.white);
                else
                    inColorPointButton.setForeground(Color.black);
            }
        }
                }
                else if (target == inColorTextButton)
                {
                        GesMap inputMap = window.getInMap();
        if (inputMap != null)
        {
            Color oldColor = inputMap.getLabelColor();
            Color newColor = JColorChooser.showDialog(null, "Change Label Color
for Input Map", oldColor);
            if (newColor != null)
            {
                inputMap.setLabelColor (newColor);
                inColorTextButton.setBackground (newColor);

                // To avoid both background and label colors are all black
                if (isBlackColor(newColor))
                    inColorTextButton.setForeground(Color.white);
                else
                    inColorTextButton.setForeground(Color.black);
            }
        }
                }
                else if (target == inSCLineButton)
                {
                        GesMap inputMap = window.getInMap();
        if (inputMap != null)
        {
```

```
            Color oldColor = inputMap.getSelectedLineColor();
            Color newColor = JColorChooser.showDialog(null, "Change Color for
Selected Lines of Input Map", oldColor);
            if (newColor != null)
            {
               inputMap.setSelectedLineColor (newColor);
               inSCLineButton.setBackground (newColor);

               // To avoid both background and label colors are all black
               if (isBlackColor(newColor))
                  inSCLineButton.setForeground(Color.white);
               else
                  inSCLineButton.setForeground(Color.black);
            }
         }
                  }
                  if (target == inSCPointButton)
                  {
                           GesMap inputMap = window.getInMap();
         if (inputMap != null)
         {
            Color oldColor = inputMap.getSelectedPointOutlineColor();
            Color newColor = JColorChooser.showDialog(null, "Change Color for
Selected Vertices of Input Map", oldColor);
            if (newColor != null)
            {
               inputMap.setSelectedPointOutlineColor (newColor);
               inSCPointButton.setBackground (newColor);

               // To avoid both background and label colors are all black
               if (isBlackColor(newColor))
                  inSCPointButton.setForeground(Color.white);
               else
                  inSCPointButton.setForeground(Color.black);
            }
         }
                  }
                  else if (target == inSCTextButton)
                  {
                           GesMap inputMap = window.getInMap();
         if (inputMap != null)
         {
            Color oldColor = inputMap.getSelectedLabelColor();
            Color newColor = JColorChooser.showDialog(null, "Change Color for
Selected Labels of Input Map", oldColor);
            if (newColor != null)
            {
               inputMap.setSelectedLabelColor (newColor);
               inSCTextButton.setBackground (newColor);
```

```
                // To avoid both background and label colors are all black
                if (isBlackColor(newColor))
                   inSCTextButton.setForeground(Color.white);
                else
                   inSCTextButton.setForeground(Color.black);
             }
          }
                }
                else if (target == showOutputLineBox)
                {
                        GesMap outMap = window.getOutMap();
                        if (outMap != null)
                                outMap.setShowLinesFlag
(showOutputLineBox.isSelected ());
                }
                else if (target == showOutputPointBox)
                {
                        GesMap outMap = window.getOutMap();
                        if (outMap != null)
                                outMap.setShowPointsFlag
(showOutputPointBox.isSelected ());
                }
                else if (target == showOutputTextBox)
                {
                        GesMap outMap = window.getOutMap();
                        if (outMap != null)
                                outMap.setShowTextFlag
(showOutputTextBox.isSelected ());
                }
                else if (target == outColorLineButton)
                {
                        GesMap outMap = window.getOutMap();
          if (outMap != null)
          {
             Color oldColor = outMap.getLineColor();
             Color newColor = JColorChooser.showDialog(null, "Change Line Color for
Output Map", oldColor);
             if (newColor != null)
             {
                outMap.setLineColor (newColor);
                outColorLineButton.setBackground (newColor);

                // To avoid both background and label colors are all black
                if (isBlackColor(newColor))
                   outColorLineButton.setForeground(Color.white);
                else
                   outColorLineButton.setForeground(Color.black);
             }
          }
                }
```

```
                else if (target == outColorPointButton)
                {
                        GesMap outMap = window.getOutMap();
        if (outMap != null)
        {
           Color oldColor = outMap.getPointColor();
           Color newColor = JColorChooser.showDialog(null, "Change Vertex Color
for Output Map", oldColor);
           if (newColor != null)
           {
              outMap.setPointColor (newColor);
              outColorPointButton.setBackground (newColor);

              // To avoid both background and label colors are all black
              if (isBlackColor(newColor))
                 outColorPointButton.setForeground(Color.white);
              else
                 outColorPointButton.setForeground(Color.black);
           }
        }
                }
                else if (target == outColorTextButton)
                {
                        GesMap outMap = window.getOutMap();
        if (outMap != null)
        {
           Color oldColor = outMap.getLabelColor();
           Color newColor = JColorChooser.showDialog(null, "Change Label Color
for Output Map", oldColor);
           if (newColor != null)
           {
              outMap.setLabelColor (newColor);
              outColorTextButton.setBackground (newColor);

              // To avoid both background and label colors are all black
              if (isBlackColor(newColor))
                 outColorTextButton.setForeground(Color.white);
              else
                 outColorTextButton.setForeground(Color.black);
           }
        }
                }
                else if (target == outSCLineButton)
                {
                        GesMap outMap = window.getOutMap();
        if (outMap != null)
        {
           Color oldColor = outMap.getSelectedLineColor();
           Color newColor = JColorChooser.showDialog(null, "Change Color for
Selected Lines of Output Map", oldColor);
```

```
            if (newColor != null)
            {
               outMap.setSelectedLineColor (newColor);
               outSCLineButton.setBackground (newColor);

               // To avoid both background and label colors are all black
               if (isBlackColor(newColor))
                  outSCLineButton.setForeground(Color.white);
               else
                  outSCLineButton.setForeground(Color.black);
            }
         }
                  }
                  else if (target == outSCPointButton)
                  {
                           GesMap outMap = window.getOutMap();
         if (outMap != null)
         {
            Color oldColor = outMap.getSelectedPointOutlineColor();
            Color newColor = JColorChooser.showDialog(null, "Change Color for
Selected Vertices of Output Map", oldColor);
            if (newColor != null)
            {
               outMap.setSelectedPointOutlineColor (newColor);
               outSCPointButton.setBackground (newColor);

               // To avoid both background and label colors are all black
               if (isBlackColor(newColor))
                  outSCPointButton.setForeground(Color.white);
               else
                  outSCPointButton.setForeground(Color.black);
            }
         }
                  }
                  else if (target == outSCTextButton)
                  {
                           GesMap outMap = window.getOutMap();
         if (outMap != null)
         {
            Color oldColor = outMap.getSelectedLabelColor();
            Color newColor = JColorChooser.showDialog(null, "Change Color for
Selected Labels of Output Map", oldColor);
            if (newColor != null)
            {
               outMap.setSelectedLabelColor (newColor);
               outSCTextButton.setBackground (newColor);

               // To avoid both background and label colors are all black
               if (isBlackColor(newColor))
                  outSCTextButton.setForeground(Color.white);
```

```
                else
                    outSCTextButton.setForeground(Color.black);
            }
        }
                }

                window.update ();
        }

    private boolean isBlackColor(Color color)
    {
       return (color.getRed() <= 51 && color.getGreen() <= 51 && color.getBlue() <=
51);
    }

        public void update ()
        {
                this.repaint();
        }

}

============================================================
/*
*************************************************************************
*****
 * Created on: 26 May, 2007
 *
 *  GesGUI.java: visualize the graphical user interface (GUI) for GES
 *
*************************************************************************
*****
 */

package jges.gui;

import java.util.Vector;
import java.util.Enumeration;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;



public class GesGUI extends JFrame
{
    protected static final String metal = "javax.swing.plaf.metal.MetalLookAndFeel";
    protected static final String windows =
"com.sun.java.swing.plaf.windows.WindowsLookAndFeel";
```

```
   protected static final String motif =
"com.sun.java.swing.plaf.motif.MotifLookAndFeel";
   protected static final String mac =
"com.sun.java.swing.plaf.mac.MacLookAndFeel";

        private GUIBuilder guiBuilder;
        private ControlPanel controlPanel;
        private MapWindow window;

        public GesGUI ()
        {
                super ("Generalization Expert System v1.0");


                getAccessibleContext().setAccessibleDescription("A Java application
for generalization of road map.");
      int WIDTH = 400, HEIGHT = 200;
      setSize(WIDTH, HEIGHT);
      Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
      setLocation(d.width/2 - WIDTH/2, d.height/2 - HEIGHT/2);
      setCursor(Cursor.getPredefinedCursor(Cursor.WAIT_CURSOR));

                // ensure the nice window decoration
                /*JFrame.setDefaultLookAndFeelDecorated(true);*/
                 try {
                        UIManager.setLookAndFeel(metal);
                } catch (Exception ex) {
                        System.out.println(ex);
                }

                // mix light and heavy weight components
                JPopupMenu.setDefaultLightWeightPopupEnabled (false);

                window = new MapWindow(this);
                guiBuilder = new GUIBuilder(this, window);

                // set default close operation
      setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
                addWindowListener (new WindowAdapter()
                {
                        public void windowClosing(WindowEvent e)
                        {
                                guiBuilder.getGUICommand ().exitGES();
                        }
                });

                // set this frame to be the root of all components in 4DTHERM
                JOptionPane.setRootFrame(this);

                // add menubar and toolbar at the top of window
```

```
                setJMenuBar( guiBuilder.createMenuBar() );
        getContentPane().add(guiBuilder.createToolBar(), BorderLayout.NORTH);

                getContentPane().add(window, BorderLayout.CENTER);

                controlPanel = new ControlPanel(window);
                getContentPane().add(controlPanel, BorderLayout.SOUTH);

                pack(); // pack() must before setSize(), otherwise setSize() wouldn't
work
        setLocation(0, 0);
        setSize(d.width, d.height-30);  // windows taskbar is 30 pixels high
        setCursor(Cursor.getPredefinedCursor(Cursor.DEFAULT_CURSOR));

        setVisible (true);
                System.out.println ("GES GUI set up completed.");
           }

           // Return a reference of the Control Panel
           public ControlPanel getControlPanel ()
           {
                return controlPanel;
           }

           //   the control Panel
           public void setControlPanelColor (Color color )
           {
                if (color != null)
           controlPanel.setPanelColor(color);
           }

           // show/Hide the control Panel based on the "isVisible" flag
           public void setControlPanelVisible (boolean isVisible)
           {
                controlPanel.setVisible(isVisible);
                window.update ();
           }

           public void updateScaleComboBox(String value)
           {
                guiBuilder.scaleComboBox.setSelectedItem (value);
           }
}
```

```
/*
*********************************************************************
*********Created on 03 June 2007
 *
 *  GUIBuilder.java: creates GUI components, such as menubars & toolbars
 *
 *
*********************************************************************
*********
 */
package jges.gui;

import java.util.Vector;
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;


public class GUIBuilder implements ActionListener
{
	// buttons used in toolbar
	private JButton openButton, appendButton, runButton, saveButton,
saveAsButton, closeButton, exitButton;
	public JComboBox scaleComboBox;

	// 'File' Menu
	private JMenuItem openMenuItem, appendMenuItem, runMenuItem,
saveMenuItem, saveAsMenuItem, closeMenuItem, exitMenuItem;

	// 'Operations' Menu
	private JMenuItem selectionMenuItem, classificationMenuItem,
typificationMenuItem, symbolizationMenuItem;

	// 'Options' Menu
	private JMenuItem backgroundMenuItem, foregroundMenuItem,
controlPanelColorMenuItem;

	// 'Knowledge Rules' Menu
	private JRadioButton rule1RadioButton, rule2RadioButton, rule3RadioButton,
rule4RadioButton, rule5RadioButton;

	// 'Help' Menu
	private JMenuItem aboutMenuItem, topicMenuItem, homepageMenuItem;

	// ------ Control Panel --------------
	private JCheckBoxMenuItem pointCheckBox, legendCheckBox,
labelCheckBox,
					scaleCheckBox, locationCheckBox,
controlPanelCheckBox;
```

```
        private Font font = new Font("serif", Font.PLAIN, 12);

        final String[] scales =
        {
                "1000", "800", "600", "500","400","300","200","150",
                "100","75", "50", "25", "10", "Fit Height", "Fit Width"
        };


//      private static boolean showOutline, showLegend, showAxis,
showControlPanel;
        private  GesGUI gui;
        private  Command command;
        private MapWindow window;

        public GUIBuilder(GesGUI gui, MapWindow window)
        {
                this.gui = gui;
                this.window = window;
                command = new Command(gui, window);
        }

        // return the command object
        public Command getGUICommand ()
        {
                return command;
        }

        // create floatable toolbar with icons
        public JToolBar createToolBar()
        {
        JToolBar toolbar = new JToolBar();

      // Make toolbar floatable
        toolbar.setFloatable (true);

        // Create an 'Open' button
        toolbar.add (openButton = new JButton ("Open"));
        openButton.setToolTipText ("Open and visualise Input Shape/CSV Files");
        openButton.addActionListener (this);

        // Create an 'P-B Algorithm' button
        //toolbar.add (appendButton = new JButton ("Append"));
        //appendButton.setToolTipText ("Append shapes to previous opened shapes");
        //appendButton.addActionListener (this);

                toolbar.addSeparator();         // Create a separator

        // Create an 'P-B Algorithm' button
        toolbar.add (runButton = new JButton ("Run"));
```

```
        runButton.setToolTipText ("Run line generalization/simplification
processing");
        runButton.addActionListener (this);

                toolbar.addSeparator();         // Create a separator

      // Create a 'Save DP' button
        toolbar.add (saveButton = new JButton ("Save"));
        saveButton.setToolTipText ("Save the output shapes with the same file
name");
        saveButton.addActionListener (this);

      // Create a 'Save DP' button
        toolbar.add (saveAsButton = new JButton ("Save As"));
        saveAsButton.setToolTipText ("Save the output shapes as a new file name");
        saveAsButton.addActionListener (this);

                toolbar.addSeparator();         // Create a separator

      // Close the input map
        toolbar.add (closeButton = new JButton ("Close"));
        closeButton.setToolTipText ("Close the input and output maps");
        closeButton.addActionListener (this);

                toolbar.addSeparator();         // Create a separator

      // Create a 'Exit' button
        toolbar.add (exitButton = new JButton ("Exit"));
        exitButton.setToolTipText ("Exit the Generalization Expert System (GES, v.
1.0)");
        exitButton.addActionListener (this);

                toolbar.addSeparator();         // Create a separator
                toolbar.addSeparator();         // Create a separator
                toolbar.addSeparator();         // Create a separator

                toolbar.add(new JLabel ("Scale"));
                toolbar.add(scaleComboBox = new JComboBox (scales));
                toolbar.add(new JLabel ("%"));
                scaleComboBox.setEditable(true);
                scaleComboBox.setSelectedItem("100");
                scaleComboBox.setPreferredSize(new Dimension(80,20));
      scaleComboBox.setMinimumSize(new Dimension(50,20));
      scaleComboBox.setMaximumSize(new Dimension(100,20));
                scaleComboBox.addActionListener(this);

        return toolbar;
    }

        // create a menubar with a set of main menus and their items
```

```
	public JMenuBar createMenuBar()
	{
		JMenuBar menuBar = new JMenuBar();
		//add all menus into menubar
		menuBar.add (createFileMenu());
		menuBar.add (createOperationMenu());
		menuBar.add (createRuleMenu());
		menuBar.add (createOptionMenu());
		menuBar.add (createHelpMenu());

		return menuBar;
	}

	/* ----------------------------------------------------------
	 * Part 1: Create 'File' menu and its menu items
	 * ----------------------------------------------------------
	 */
	public JMenu createFileMenu ()
	{
		JMenu fileMenu = new JMenu ("File");
		fileMenu.setMnemonic (KeyEvent.VK_F);

		// Open a data file for modeling
		fileMenu.add (openMenuItem = new JMenuItem ("Open"));
		openMenuItem.setMnemonic (KeyEvent.VK_O);
		openMenuItem.addActionListener (this);

		// Create a menu MenuItem Delete
		//fileMenu.add (appendMenuItem = new JMenuItem ("Append"));
	//appendMenuItem.setToolTipText ("Append shapes to previous opened
shapes");
		//appendMenuItem.setMnemonic (KeyEvent.VK_P);
		//appendMenuItem.addActionListener (this);

		fileMenu.addSeparator();   // Create a separator

		// Create a menu MenuItem Delete
		fileMenu.add (runMenuItem=new JMenuItem("Run"));
	runMenuItem.setToolTipText ("Run line generalization/simplification
processing");
		runMenuItem.setMnemonic (KeyEvent.VK_R);
		runMenuItem.addActionListener (this);

		fileMenu.addSeparator();   // Create a separator

		// Save results into a file with a new file name
		fileMenu.add (saveMenuItem = new JMenuItem ("Save"));
		saveMenuItem.setMnemonic (KeyEvent.VK_S);
		saveMenuItem.addActionListener (this);
```

```
                fileMenu.add (saveAsMenuItem = new JMenuItem ("Save As"));
                saveAsMenuItem.setMnemonic (KeyEvent.VK_A);
                saveAsMenuItem.addActionListener (this);

                fileMenu.addSeparator();   // Create a separator

                // Close input map
                fileMenu.add (closeMenuItem = new JMenuItem ("Close Map"));
                closeMenuItem.setMnemonic (KeyEvent.VK_C);
                closeMenuItem.addActionListener (this);

                fileMenu.addSeparator();   // Create a separator

                // Exit this software afetr closing all windows and GUI componem
                fileMenu.add (exitMenuItem = new JMenuItem ("Exit"));
                exitMenuItem.setMnemonic (KeyEvent.VK_E);
                exitMenuItem.addActionListener (this);

                return fileMenu;
        }

        /* ----------------------------------------------------------
         * Part 3: Create Operations menu and its menuItems
         * ----------------------------------------------------------
         */
        public JMenu createOperationMenu ()
        {
                JMenu operationsMenu = new JMenu ("Operations");
                operationsMenu.setMnemonic (KeyEvent.VK_O);

                operationsMenu.add (selectionMenuItem = new JMenuItem
("Selection"));
                selectionMenuItem.setMnemonic (KeyEvent.VK_S);
                selectionMenuItem.addActionListener (this);

                operationsMenu.add (classificationMenuItem = new JMenuItem
("Classification"));
                classificationMenuItem.setMnemonic (KeyEvent.VK_C);
                classificationMenuItem.addActionListener (this);

                operationsMenu.add (typificationMenuItem = new JMenuItem
("Typification"));
                typificationMenuItem.setMnemonic (KeyEvent.VK_Y);
                typificationMenuItem.addActionListener (this);

                operationsMenu.add (symbolizationMenuItem = new JMenuItem
("Symbolization"));
                symbolizationMenuItem.setMnemonic (KeyEvent.VK_Y);
                symbolizationMenuItem.addActionListener (this);
```

```
            return operationsMenu;
        }


        /* ----------------------------------------------------------
         * Part 4: Create option menu and its menuItems
         * ----------------------------------------------------------
         */
        public JMenu createOptionMenu ()
        {
            JMenu optionsMenu = new JMenu ("Options");
            optionsMenu.setMnemonic (KeyEvent.VK_O);

            optionsMenu.add (backgroundMenuItem = new JMenuItem
("Background Color"));
            backgroundMenuItem.setMnemonic (KeyEvent.VK_B);
            backgroundMenuItem.addActionListener (this);

            optionsMenu.add (controlPanelColorMenuItem = new JMenuItem
("Control Panel Color"));
            controlPanelColorMenuItem.setMnemonic (KeyEvent.VK_C);
            controlPanelColorMenuItem.addActionListener (this);

            optionsMenu.addSeparator();   // Create a separator

            optionsMenu.add (pointCheckBox = new JCheckBoxMenuItem
("Show Data Points", true));
            pointCheckBox.setMnemonic (KeyEvent.VK_O);
            pointCheckBox.addActionListener (this);

            optionsMenu.add (scaleCheckBox = new JCheckBoxMenuItem
("Show Scales", true));
            scaleCheckBox.setMnemonic (KeyEvent.VK_S);
            scaleCheckBox.addActionListener (this);

            optionsMenu.add (locationCheckBox = new JCheckBoxMenuItem
("Show Location", true));
            locationCheckBox.setMnemonic (KeyEvent.VK_D);
            locationCheckBox.addActionListener (this);

            optionsMenu.addSeparator();   // Create a separator

            optionsMenu.add (controlPanelCheckBox = new JCheckBoxMenuItem
("Show Control Panel", true));
            controlPanelCheckBox.setMnemonic (KeyEvent.VK_P);
            controlPanelCheckBox.addActionListener (this);

            return optionsMenu;
        }
```

```
	/* ----------------------------------------------------------
	 * Part 5: Create 'Knowledge Rules' menu and its menuItems
	 * ----------------------------------------------------------
	 */
	public JMenu createRuleMenu ()
	{
		JMenu ruleMenu = new JMenu ("Knowledge Rules");
		ruleMenu.setMnemonic (KeyEvent.VK_R);

		ruleMenu.add (rule1RadioButton = new JRadioButton ("Rule 1",
true));
		rule1RadioButton.setMnemonic (KeyEvent.VK_1);
		rule1RadioButton.addActionListener (this);

		ruleMenu.add (rule2RadioButton = new JRadioButton ("Rule 2"));
		rule2RadioButton.setMnemonic (KeyEvent.VK_2);
		rule2RadioButton.addActionListener (this);

		ruleMenu.add (rule3RadioButton = new JRadioButton ("Rule 3"));
		rule3RadioButton.setMnemonic (KeyEvent.VK_3);
		rule3RadioButton.addActionListener (this);

		ruleMenu.add (rule4RadioButton = new JRadioButton ("Rule 4"));
		rule4RadioButton.setMnemonic (KeyEvent.VK_4);
		rule4RadioButton.addActionListener (this);

		ruleMenu.add (rule5RadioButton = new JRadioButton ("Rule 5"));
		rule5RadioButton.setMnemonic (KeyEvent.VK_5);
		rule5RadioButton.addActionListener (this);

	ButtonGroup ruleGroup = new ButtonGroup();
	ruleGroup.add(rule1RadioButton);
	ruleGroup.add(rule2RadioButton);
	ruleGroup.add(rule3RadioButton);
	ruleGroup.add(rule4RadioButton);
	ruleGroup.add(rule5RadioButton);


		return ruleMenu;
	}

	/* ----------------------------------------------------------
	 * Part 6: Create 'Help' menu and its menuItems
	 * ----------------------------------------------------------
	 */
	public JMenu createHelpMenu ()
	{
		JMenu helpMenu = new JMenu ("Help");
		helpMenu.setMnemonic (KeyEvent.VK_H);
```

```
                helpMenu.add (aboutMenuItem = new JMenuItem ("About GES"));
                aboutMenuItem.setMnemonic (KeyEvent.VK_A);
                aboutMenuItem.addActionListener (this);

                helpMenu.add (topicMenuItem = new JMenuItem ("Help Topics"));
                topicMenuItem.setMnemonic (KeyEvent.VK_T);
                topicMenuItem.addActionListener (this);

                helpMenu.add (homepageMenuItem = new JMenuItem ("Home
Page"));
                homepageMenuItem.setMnemonic (KeyEvent.VK_P);
                homepageMenuItem.addActionListener (this);

                return helpMenu;
        }

        public void actionPerformed(ActionEvent e)
   {
      Object target = e.getSource();

                // File Menu -------------------------------------------------
                if (target == openButton || target == openMenuItem)
                {
                        command.open();
                }
                //else if (target == appendMenuItem || target == appendButton)
                //{
                //        command.append();
                //}
                else if (target == runMenuItem || target == runButton)
                {
                        command.run ();
                }
                else if (target == saveMenuItem || target == saveButton)
                {
                        command.save ();
                }
                else if (target == saveAsMenuItem || target == saveAsButton)
                {
                        command.saveAs ();
                }
                else if (target == closeMenuItem || target == closeButton)
                {
                        command.closeInputMap ();
                }
                else if (target == exitMenuItem || target == exitButton)
                {
                        command.exitGES ();
                }
                // Operations menu ----------------------------------------------
```

```
		else if (target == selectionMenuItem)
		{
			// command.setSelection ();
		}
		else if (target == classificationMenuItem)
		{
			// command.setClassification ();
		}
		else if (target == typificationMenuItem)
		{
			// command.setTypification ();
		}
		else if (target == symbolizationMenuItem)
		{
			// command.setSymbolization ();
		}
		// Options menu ----------------------------------------------
		else if (target == backgroundMenuItem)
		{
			command.setBackground ();
		}
		else if (target == foregroundMenuItem)
		{
			command.setForeground ();
		}
		else if (target == controlPanelColorMenuItem)
		{
			command.setControlPanelColor ();
		}
		else if (target == pointCheckBox)
		{
			command.showPoints (pointCheckBox.isSelected());
		}
		else if (target == scaleCheckBox)
		{
			command.showScales (scaleCheckBox.isSelected());
		}
		else if (target == locationCheckBox)
		{
			command.showDataLocation (locationCheckBox.isSelected());
		}
		else if (target == controlPanelCheckBox)
		{
			command.showControlPanel
(controlPanelCheckBox.isSelected());
		}
		else if (target == rule1RadioButton)
		{
			command.setRuleFile(1);
		}
```

```
		else if (target == rule2RadioButton)
		{
			command.setRuleFile(2);
		}
		else if (target == rule3RadioButton)
		{
			command.setRuleFile(3);
		}
		else if (target == rule4RadioButton)
		{
			command.setRuleFile(4);
		}
		else if (target == rule5RadioButton)
		{
			command.setRuleFile(5);
		}
		else if (target == aboutMenuItem)
		{
			// command.showAboutGES ();
		}
		else if (target == topicMenuItem)
		{
			// command.showHelpTopics ();
		}
		else if (target == homepageMenuItem)
		{
			// command.showHomepage ();
		}
		else if (target == scaleComboBox)
		{
			String scaleValue = (String)scaleComboBox.getSelectedItem
();
			if (scaleValue != null)
			{
				scaleValue = scaleValue.trim();
				if (scaleValue.endsWith("%"))
				{
					scaleValue = scaleValue.substring (0,
scaleValue.length()-1);
					scaleComboBox.setSelectedItem(scaleValue);
				}

				if (scaleValue.equalsIgnoreCase("Fit Height"))
					window.setScale(-1);
				else if (scaleValue.equalsIgnoreCase("Fit Width"))
					window.setScale(-2);
				else
				{
					try {
						int value = Integer.parseInt(scaleValue);
```

```
                                        window.setScale(value);
                                    } catch (Exception ex) {
                                        System.out.println("Scale value entered
is not an integer value: "+scaleValue);
                                    }
                                }
                            }
                        }
                        // window.update();
                }
        }
==========================================================
package jges.gui;

import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.border.TitledBorder;

import jges.com.*;
// import jges.util.*;


public class MapWindow extends JPanel implements MouseListener
{
        static final long serialVersionUID = 123456L;
        private boolean isDosSystem = true;    // false;

        // Fixed temporary output file names for both input shape and dbf files
        private String tmpInShpFile = "../tmp/tmpdata_shp.out";
        private String tmpInDbfFile = "../tmp/tmpdata_dbf.out";

        private String tmpOutShpFile = "../tmp/tmpdata_shp_dp.out";
        private String tmpOutDbfFile = "../tmp/tmpdata_dbf_dp.out";

        public Color bgColor, fgColor;
        protected final GesGUI gui;

        // input shape and dbf file names
        private String inShpFile, inDbfFile;

        // private Color[] colors = {Color.red, Color.blue,Color.green, Color.yellow};
        // private final Font font = new Font("serif", Font.PLAIN, 12);

        private Point firstClickedPoint, windowCenter;  // center point of the window
        private int width, height;
        private double globalScale;
    private int selectedShapeNo = -1;
```

```
        public double maxX, minX, maxY, minY, maxValue, minValue;
        public String xUnit, yUnit, valueUnit;

        // Input and output maps
        public GesMap inputMap;
        public GesMap outMap;

        public MapWindow (GesGUI gui)
    {
                this.gui = gui;
                setLayout(new BorderLayout());  // default layout
                inShpFile = "";
                inDbfFile = "";

                inputMap = new GesMap();
                addMouseListener (this);  // used for moving and scaling inputMap
image
                windowCenter = null;

                bgColor = Color.white;
                fgColor = Color.black;

                // scale whole image to fit the window and then enlarge it if need
                globalScale = 1.0;
    }

        public GesGUI getGESGui ()
        {
                return gui;
        }

        public void update ()
        {
                repaint();
        }

        // set the background color for the modelling window
    public void setBackgroundColor (Color color)
        {
                if (color != null)
                {
                        bgColor = color;
                        update();
                }
        }

        // set the foreground color for the letters and lines
    public void setForegroundColor (Color color)
        {
```

```
            if (color != null)
            {
                this.fgColor = color;
                update();
            }
        }

        // return the background color
public Color getBackgroundColor ()
        {
            return bgColor;
        }

        // return the foreground color
public Color getForegroundColor ()
        {
            return fgColor;
        }

        /* ------------------------------------------------------------
         *  a set of methods for reading and saving data
         * ------------------------------------------------------------
         */
        // return the shaoe file name of the input map
        public String getShpFileName ()
        {
            return inShpFile;
        }

        // set the shape file name to a new file name
        public void setShpFileName (String fileName)
        {
            this.inShpFile = fileName;
        }

        // return the dbf file  nameof the input map
        public String getDbfFileName ()
        {
            return inDbfFile;
        }

        // set the dbf file name to a new file name
        public void setDbfFileName (String fileName)
        {
            this.inDbfFile = fileName;
        }

        // load data from specific file and then parse them
        public void loadData(String inputFile)
        {
```

```
                // Set the GUI Title with the input file name to be  opened
                gui.setTitle("Line Generalization Expert System (v1.0) - "+inputFile);

                // Input file is allowed to be either ".shp" or ".dbf"
                if (inputFile.endsWith(".shp"))
                {
                        inShpFile = inputFile;
                        inDbfFile = inputFile.substring(0, inputFile.length()-4)+".dbf";
        loadBinaryShapeDbfFile(inShpFile, inDbfFile);
                }
                else if (inputFile.endsWith(".dbf"))
                {
                        inShpFile = inputFile.substring(0, inputFile.length()-4)+".shp";
                        inDbfFile = inputFile;
        loadBinaryShapeDbfFile(inShpFile, inDbfFile);
                }
                else if (inputFile.endsWith("_shp.out"))
                {
        String shpOutFile = inputFile;
                        String dbfOutFile = inputFile.substring(0, inputFile.length()-
8)+"_dbf.out";
        loadAsciiShapeFile(shpOutFile, dbfOutFile);
                }
                else if (inputFile.endsWith("_dbf.out"))
                {
        String shpOutFile = inputFile.substring(0, inputFile.length()-8)+"_shp.out";
                        String dbfOutFile = inputFile;
        loadAsciiShapeFile(shpOutFile, dbfOutFile);
                }
                else if (inputFile.endsWith(".out")) // no matched dbf out file
                {
        String shpOutFile = inputFile;
                        String dbfOutFile = null;
        loadAsciiShapeFile(shpOutFile, dbfOutFile);
                }
                else
                {
                        System.out.println("Unknown input file type: "+inputFile);
                        return ;
                }
   }

   // Open the input binary files by calling the c library functions
   public void loadBinaryShapeDbfFile(String inShpFile, String inDbfFile)
   {
                String command_shp = "../bin/shpparser.exe \""+inShpFile+"\"";
                String command_dbf = "../bin/dbfparser.exe -r -h -m
\""+inDbfFile+"\"";
```

```
                // Using "\" seperator for DOS system, e.g. when running by window
Command Prompt
                if (isDosSystem)
                {
                        command_shp = command_shp.replace('/', '\\');
                        command_dbf = command_dbf.replace('/', '\\');
                }

                // Execute the commands to extract data from both shape and dbf files
                if (executeCommand(command_shp))
                {
                        if (!executeCommand(command_dbf))
                                System.out.println("Failed to run
\""+command_dbf+"\"");
                }
                else
                        System.out.println("Failed to run \""+command_shp+"\"");

                if (inputMap == null)
                        inputMap = new GesMap();

                // load data
                inputMap.readDataFromTextFile(tmpInShpFile, tmpInDbfFile);
        gui.getControlPanel().setButtonColorsForInputMap (inputMap);
                update();
          }

    // Open the input ASCII files directly (plain text files)
    public void loadAsciiShapeFile(String shpOutFile, String dbfOutFile)
    {
                if (inputMap == null)
                        inputMap = new GesMap();

                // load plain text file
                inputMap.readDataFromTextFile(shpOutFile, dbfOutFile);
        gui.getControlPanel().setButtonColorsForInputMap (inputMap);
                update();
          }

          // load data from specific file and then parse them
          public void appendData(String inputFile)
          {
                // Set the GUI Title with the input file name to be  opened
                gui.setTitle("Line Generalization Expert System (v1.0) - "+inputFile);

                // Input file is allowed to be either ".shp" or ".dbf"
                if (inputFile.endsWith(".shp"))
                {
                        inShpFile = inputFile;
                        inDbfFile = inputFile.substring(0, inputFile.length()-4)+".dbf";
```

```
        appendBinaryShapeDbfFile(inShpFile, inDbfFile);
                }
                else if (inputFile.endsWith(".dbf"))
                {
                        inShpFile = inputFile.substring(0, inputFile.length()-4)+".shp";
                        inDbfFile = inputFile;
        appendBinaryShapeDbfFile(inShpFile, inDbfFile);
                }
                else if (inputFile.endsWith("_shp.out"))
                {
        String shpOutFile = inputFile;
                        String dbfOutFile = inputFile.substring(0, inputFile.length()-
8)+"_dbf.out";
        appendAsciiShapeFile(shpOutFile, dbfOutFile);
                }
                else if (inputFile.endsWith("_dbf.out"))
                {
        String shpOutFile = inputFile.substring(0, inputFile.length()-8)+"_shp.out";
                        String dbfOutFile = inputFile;
        appendAsciiShapeFile(shpOutFile, dbfOutFile);
                }
                else if (inputFile.endsWith(".out")) // no matched dbf out file
                {
        String shpOutFile = inputFile;
                        String dbfOutFile = null;
        appendAsciiShapeFile(shpOutFile, dbfOutFile);
                }
                else
                {
                        System.out.println("Unknown input file type: "+inputFile);
                        return ;
                }
    }

    // Open the input binary files by calling the c library functions
    public void appendBinaryShapeDbfFile(String inShpFile, String inDbfFile)
    {
                String command_shp = "../bin/shpparser.exe \""+inShpFile+"\"";
                String command_dbf = "../bin/dbfparser.exe -r -h -m
\""+inDbfFile+"\"";

                // Using "\" seperator for DOS system, e.g. when running by window
Command Prompt
                if (isDosSystem)
                {
                        command_shp = command_shp.replace('/', '\\');
                        command_dbf = command_dbf.replace('/', '\\');
                }

                // Execute the commands to extract data from both shape and dbf files
```

```
                if (executeCommand(command_shp))
                {
                        if (!executeCommand(command_dbf))
                                System.out.println("Failed to run
\""+command_dbf+"\"");
                }
                else
                        System.out.println("Failed to run \""+command_shp+"\"");

                if (inputMap == null)
                        inputMap = new GesMap();

                // load data
                inputMap.appendDataFromTextFile(tmpInShpFile, tmpInDbfFile);
                update();
        }

   // Open the input ASCII files directly (plain text files)
   public void appendAsciiShapeFile(String shpOutFile, String dbfOutFile)
   {
                if (inputMap == null)
                        inputMap = new GesMap();

                // load plain text file
                inputMap.appendDataFromTextFile(shpOutFile, dbfOutFile);
                update();
        }

        // load data from specific file and then parse them
        public void runAlgorithm(int ruleNo)
        {
                if (inputMap == null || inputMap.isEmpty())
                        return ;

      double scale = getScale (width, height)*globalScale;
      double pointTolerance = 1.0/scale;
      System.out.println("Execute \"Run\" command: pixel size="+pointTolerance);

                String command = "../bin/gui_run_algorithm.sh "+ruleNo;
      //+" \""+inShpFile+"\' \""+inDbfFile+"\"";
      System.out.println(command);

                // Execute the commans to extract data from both shape and dbf files
                if (!executeUnixCommand(command))
                        System.out.println("Failed to run \""+command+"\".");
                else
                {
                        if (outMap == null)
                                outMap = new GesMap();
                        outMap.setLineColor (Color.blue);
```

```
                        outMap.setPointColor (Color.green);
                        outMap.setLabelColor (Color.cyan);
                        outMap.setSelectedLineColor (Color.orange);
                        outMap.setSelectedPointOutlineColor (Color.orange);
                        outMap.setSelectedPointFillColor (Color.yellow);

                        // load data
                        try {
                                outMap.readDataFromTextFile(tmpOutShpFile,
tmpOutDbfFile);
            gui.getControlPanel().setButtonColorsForOutputMap (outMap);
            update();
                        }
                        catch (Exception ex) {
            System.out.println("Error in readDataFromTextFile()");
                                return;
                        }
                }
        }

        // return the configured results/data for saving
    public String getSavingData (String type)
        {
                return "";
        }

        // Clear the input map and its resulting maps
        public void clear ()
        {
                if (inputMap != null)
                {
                        inputMap.reset();
                        inputMap = null;
                }

                if (outMap != null)
                {
                        outMap.reset();
                        outMap = null;
                }

                inShpFile = null;
                inDbfFile = null;
                windowCenter = null;
                gui.setTitle("Generalization Expert System (v1.0)");
                update();
                //gui.updateGUI();
        }

    public void paint (Graphics page)
```

```
{
		Dimension d = getSize();
		if (width != d.width)
			width = d.width;

		if (height != d.height)
			height = d.height;

		page.setColor(bgColor);  // draw background color
		page.clearRect (0, 0, width, height);
		page.fillRect (0, 0, width, height);

		page.setColor(fgColor);
		page.drawRect (0, 0, width-1, height-1);

		// scale whole image to fit the window and then enlarge it if need
		double scale = getScale (width, height)*globalScale;
		if (windowCenter == null)
			windowCenter = new Point(width/2, height/2);

		if (inputMap != null && !inputMap.isEmpty())
			inputMap.drawMap (page, scale, windowCenter);

		if (outMap != null && !outMap.isEmpty())
			outMap.drawMap (page, scale, windowCenter);

   drawSelectedShapeInfo(page);
      }

private void drawSelectedShapeInfo(Graphics page)
{
   if (firstClickedPoint == null || selectedShapeNo == -1 || inputMap == null)
      return ;
   String shpInfo = inputMap.shapeList[selectedShapeNo].getSummaryInfo();
   shpInfo = "Shape "+(selectedShapeNo+1)+": "+shpInfo;
   page.setColor(Color.cyan);
   page.fillRect(firstClickedPoint.x+5, firstClickedPoint.y-10, 220, 20);
   page.setColor(Color.black);
   page.drawString(shpInfo, firstClickedPoint.x+10, firstClickedPoint.y+5);
}

      // scale data to fit the whole window
      private double getScale (int wid, int high)
      {
		if (inputMap == null || (wid == 0 && high == 0))
			return 1.0;

		GesPoint[] bounds = inputMap.getBounds ();
		if (bounds == null || bounds[0] == null || bounds[1] == null)
			return 1.0;
```

```
                double xRange = bounds[1].x - bounds[0].x;
                double yRange = bounds[1].y - bounds[0].y;

                if (xRange <= 0 || yRange <= 0)
                        return 1.0;

                double xScale = wid/xRange;
                double yScale = high/yRange;

                // choose smaller one to ensure same scale in both directions
                return (xScale<=0 ? yScale : (yScale<=0 ? xScale : (xScale<yScale ?
xScale : yScale)));
        }

        public void setScale(int scale)
        {
                if (inputMap == null)
                        return ;

                if (scale == -1) // fit hieght
                {
                        double value1 = getScale (width, height);
                        double value2 = getScale (0, height-5);
                        globalScale = (value1==0) ? 1.0 : value2/value1;
                        windowCenter = new Point(width/2, height/2);
                }
                else if (scale == -2.0) // fit width
                {
                        double value1 = getScale (width, height);
                        double value2 = getScale (width-5, 0);
                        globalScale = (value1==0) ? 1.0 : value2/value1;
                        windowCenter = new Point(width/2, height/2);
                }
                else
                        globalScale = scale/100.0;

                update();
        }

        // ------------------------------------------------------------------------
        // implement mouse listerner interface for moving and scaling inputMap image
        // ------------------------------------------------------------------------
        public void mouseReleased (MouseEvent event)
        {
                if (windowCenter == null || firstClickedPoint == null || inputMap ==
null)
                        return;

                Point point2 = event.getPoint();
```

```
                if (point2 == null)
                        return;
                int x = point2.x - firstClickedPoint.x;
                int y = point2.y - firstClickedPoint.y;

                if (event.getButton() == MouseEvent.BUTTON3) //right button for
moving image
                {
                        windowCenter = new Point (windowCenter.x+x,
windowCenter.y+y);
                }
                else if (event.getButton() == MouseEvent.BUTTON2)  //middle button
for scaling image
                {
                        globalScale += (double)y/height + (double)x/width;
                        if (globalScale < 0.01)  // minimum scale is 1%
                                globalScale = 0.01;
                        gui.updateScaleComboBox
(String.valueOf((int)(globalScale*100)));
                }
                /* else // left button for select shape
                {
                        double scale = getScale (width, height)*globalScale;
                        int shapeNo = inputMap.findSelectedShapeNo (point2, scale,
windowCenter);
          if (outMap != null)
             outMap.setSelectedShapeNo(shapeNo);
                }*/

                firstClickedPoint = null;
                repaint ();
          }

          // get mouse's position when it pressed
          public void mousePressed (MouseEvent event)
          {
       if (inputMap == null)
          return;
       selectedShapeNo = -1;
                firstClickedPoint = event.getPoint();

       // left button for selecting shape
       if (firstClickedPoint != null && event.getButton() == MouseEvent.BUTTON1)
                {
                        double scale = getScale (width, height)*globalScale;
                        selectedShapeNo = inputMap.findSelectedShapeNo
(firstClickedPoint, scale, windowCenter);
          if (outMap != null)
             outMap.setSelectedShapeNo(selectedShapeNo);
          repaint ();
```

```
        }
}

// other methods in mouseListener interface
public void mouseClicked (MouseEvent edges){ }
public void mouseEntered (MouseEvent edges){ }
public void mouseExited  (MouseEvent edges){ }

// set scales showable state (called by a GUI command)
public void setScaleShowable (boolean isShowing)
{

}

// Return the input map
public GesMap getInMap ()
{
        return inputMap;
}

// Return DP map whic is the generalized version of the input map
public GesMap getOutMap ()
{
        return outMap;
}

public void saveData ()
{
        String data = "";
        saveData (inShpFile, data);
}

public void saveData (String fileName, String data)
{
        inputMap.saveData (fileName, data);
}

// Return the file name without the file extension from the full file path
public String getFileNameWithoutExtn(String fullFilePath)
{
        int index = fullFilePath.lastIndexOf("/");
        if (index == -1)
                index = fullFilePath.lastIndexOf("\\");
        if (index == -1)
                return fullFilePath.substring(0, fullFilePath.length()-4);
        String fileName = fullFilePath.substring(index+1);
        return fileName.substring(0, fileName.length()-4);
}
```

```
	public boolean executeCommand(String command)
	{
		try
		{
			String osName = System.getProperty("os.name");
			System.out.println("OS Name=" + osName) ;

	Runtime rt = Runtime.getRuntime();
	Process proc = null;

				// Run by Windows OS
				if (osName.startsWith("Windows"))
				{
					if (isDosSystem)
					{
						String[] cmds = new String[3];
						if(osName.equals("Windows NT") ||
osName.equals("Windows XP"))
						{
							cmds[0] = "cmd.exe";
							cmds[1] = "/C";
							cmds[2] = command;
						}
						else // if(osName.equals("Windows 95")  ||
osName.equals("Windows 98") || osName.equals("Windows 00"))
						{
							cmds[0] = "command.com";
							cmds[1] = "/C";
							cmds[2] = command;
						}
						proc = rt.exec(cmds);
					}
					else  // Linux-like command line program running on
WIndows OS
						proc = rt.exec (command);
				}
				// Run by Unix/Linux OS
				else
					proc = rt.exec (command);

	// any error message?
	ExecuteInfo errorInfo = new ExecuteInfo(proc.getErrorStream(), "ERROR");

	// any output?
	ExecuteInfo outputInfo = new ExecuteInfo(proc.getInputStream(),
"OUTPUT");

	// kick them off
	errorInfo.start();
	outputInfo.start();
```

```
        // any error???
        int exitVal = proc.waitFor();
        System.out.println("Exit status of executing "+command+"="+exitVal);

        // Stop the two threads by set them to null objects (!!!do not use stop() as it is
unsafe !!!)
        //errorInfo = null;
        //outputInfo = null;

                        return (exitVal==0);
    }
            catch (Throwable t)
            {
        t.printStackTrace();
                        return false;
            }
        }


        // Execute a Unix- or Linux-like command
        public boolean executeUnixCommand(String command)
        {
            try
            {
                        Runtime rt = Runtime.getRuntime();
        String[] cmd = new String[3];
        cmd[0] = "sh";
        cmd[1] = "-c";
        cmd[2] = command;
                        Process proc = rt.exec(cmd);

        // any error message?
        ExecuteInfo errorInfo = new ExecuteInfo(proc.getErrorStream(), "ERROR");

        // any output?
        ExecuteInfo outputInfo = new ExecuteInfo(proc.getInputStream(),
"OUTPUT");

        // kick them off
        errorInfo.start();
        outputInfo.start();

                        int exitStatus = proc.waitFor();
                        System.out.println("Exit status of executing
"+command+"="+exitStatus);
                        return (exitStatus == 0);
            }
            catch (Exception e)
            {
```

```
                    e.printStackTrace();
                    return false;
                }
        }

        // Return true of the OS is Microsoft Windows, otherwise return false.
        public boolean isWindows()
        {
                return System.getProperty("os.name").startsWith("Windows");
        }

        public String getFileSeperator()
        {
                if(isWindows())
                        return "\\";
                else
                        return "/";
        }
}


class ExecuteInfo extends Thread
{
        InputStream is;
        String type;

        ExecuteInfo (InputStream is, String type)
        {
                this.is = is;
                this.type = type;
        }

        public void run ()
        {
                try
                {
                        InputStreamReader isr = new InputStreamReader(is);
                        BufferedReader br = new BufferedReader(isr);
                        String line=null;
                        while ((line = br.readLine()) != null)
                                System.out.println(type + ">" + line);
                }
                catch (IOException ioe)
                {
                        ioe.printStackTrace();
                }
        }
}
```

```
/*
*********************************************************************
*********
 *  GUIDriver.java  Created on 27 May 2003
 *
 *  The driver class to start the GUI of Generalisation Expert System
 *
 *  *
*********************************************************************
*********
 */


package jges;

import javax.swing.*;
import jges.gui.GesGUI;


public class GUIDriver
{
	// the main method starting 4dtherm
	public static void main (String[] args)
	{
		try
		{
			GesGUI guiWindow = new GesGUI();
		}
		catch (Exception ex)
		{
			System.out.println("Unexpected exception caught while
starting Generalisation Expert System (GES)");
			System.out.println("Please report the following bugs");
			System.out.println("\t"+ex);

			String ask = "Unexpected exception caught! Do you want to
exit \"GES\" ?\n";
			int answer=JOptionPane.showConfirmDialog (null, ask,"Exit",

	JOptionPane.YES_NO_OPTION);
			if(answer==JOptionPane.NO_OPTION)
				return;
			else
				System.exit (0);
		}
	}
}
```

```
/*
 * GISFileFilter.java   Monday, 28/05/2007
 */
package jges.util;

import java.io.File;
import java.util.Hashtable;
import java.util.Enumeration;
import javax.swing.*;
import javax.swing.filechooser.*;


public class GISFileFilter extends FileFilter {

    private static String TYPE_UNKNOWN = "Type Unknown";
    private static String HIDDEN_FILE = "Hidden File";

    private Hashtable filters = null;
    private String description = null;
    private String fullDescription = null;
    private boolean useExtensionsInDescription = true;

    /**
     * Creates a file filter. If no filters are added, then all
     * files are accepted.
     *
     * @see #addExtension
     */
    public GISFileFilter() {
        this.filters = new Hashtable();
    }

    /**
     * Creates a file filter that accepts files with the given extension.
     * Example: new GISFileFilter("jpg");
     *
     * @see #addExtension
     */
    public GISFileFilter(String extension) {
        this(extension,null);
    }

    /**
     * Creates a file filter that accepts the given file type.
     * Example: new GISFileFilter("jpg", "JPEG Image Images");
     *
     * Note that the "." before the extension is not needed. If
     * provided, it will be ignored.
     *
     * @see #addExtension
```

```
 */
public GISFileFilter(String extension, String description) {
        this();
        if(extension!=null) addExtension(extension);
        if(description!=null) setDescription(description);
}

/**
 * Creates a file filter from the given string array.
 * Example: new GISFileFilter(String {"gif", "jpg"});
 *
 * Note that the "." before the extension is not needed adn
 * will be ignored.
 *
 * @see #addExtension
 */
public GISFileFilter(String[] filters) {
        this(filters, null);
}

/**
 * Creates a file filter from the given string array and description.
 * Example: new GISFileFilter(String {"gif", "jpg"}, "Gif and JPG Images");
 *
 * Note that the "." before the extension is not needed and will be ignored.
 *
 * @see #addExtension
 */
public GISFileFilter(String[] filters, String description) {
        this();
        for (int i = 0; i < filters.length; i++) {
            // add filters one by one
            addExtension(filters[i]);
        }
        if(description!=null) setDescription(description);
}

/**
 * Return true if this file should be shown in the directory pane,
 * false if it shouldn't.
 *
 * Files that begin with "." are ignored.
 *
 * @see #getExtension
 * @see FileFilter#accepts
 */
public boolean accept(File f) {
        if(f != null) {
            if(f.isDirectory()) {
                    return true;
```

```
            }
            String extension = getExtension(f);
            if(extension != null && filters.get(getExtension(f)) != null) {
                  return true;
            };
        }
        return false;
}

/**
 * Return the extension portion of the file's name .
 *
 * @see #getExtension
 * @see FileFilter#accept
 */
 public String getExtension(File f) {
        if(f != null) {
            String filename = f.getName();
            int i = filename.lastIndexOf('.');
            if(i>0 && i<filename.length()-1) {
                  return filename.substring(i+1).toLowerCase();
            };
        }
        return null;
}

/**
 * Adds a filetype "dot" extension to filter against.
 *
 * For example: the following code will create a filter that filters
 * out all files except those that end in ".jpg" and ".tif":
 *
 *   GISFileFilter filter = new GISFileFilter();
 *   filter.addExtension("jpg");
 *   filter.addExtension("tif");
 *
 * Note that the "." before the extension is not needed and will be ignored.
 */
public void addExtension(String extension) {
        if(filters == null) {
            filters = new Hashtable(5);
        }
        filters.put(extension.toLowerCase(), this);
        fullDescription = null;
}


/**
 * Returns the human readable description of this filter. For
 * example: "JPEG and GIF Image Files (*.jpg, *.gif)"
```

```
     *
     * @see setDescription
     * @see setExtensionListInDescription
     * @see isExtensionListInDescription
     * @see FileFilter#getDescription
     */
    public String getDescription() {
        if(fullDescription == null) {
            if(description == null || isExtensionListInDescription()) {
                fullDescription = description==null ? "(" : description + " (";
                // build the description from the extension list
                Enumeration extensions = filters.keys();
                if(extensions != null) {
                    fullDescription += "." + (String) extensions.nextElement();
                    while (extensions.hasMoreElements()) {
                        fullDescription += ", ." + (String) extensions.nextElement();
                    }
                }
                fullDescription += ")";
            } else {
                fullDescription = description;
            }
        }
        return fullDescription;
    }

    /**
     * Sets the human readable description of this filter. For
     * example: filter.setDescription("Gif and JPG Images");
     *
     * @see setDescription
     * @see setExtensionListInDescription
     * @see isExtensionListInDescription
     */
    public void setDescription(String description) {
        this.description = description;
        fullDescription = null;
    }

    /**
     * Determines whether the extension list (.jpg, .gif, etc) should
     * show up in the human readable description.
     *
     * Only relevent if a description was provided in the constructor
     * or using setDescription();
     *
     * @see getDescription
     * @see setDescription
     * @see isExtensionListInDescription
     */
```

```
    public void setExtensionListInDescription(boolean b) {
        useExtensionsInDescription = b;
        fullDescription = null;
    }

    /**
     * Returns whether the extension list (.jpg, .gif, etc) should
     * show up in the human readable description.
     *
     * Only relevent if a description was provided in the constructor
     * or using setDescription();
     *
     * @see getDescription
     * @see setDescription
     * @see setExtensionListInDescription
     */
    public boolean isExtensionListInDescription() {
        return useExtensionsInDescription;
    }
}
```

```
/**************************************************************************
*
        Monday , 28 May 2007

**************************************************************************
 */



package jges.util;

import java.io.*;
import java.util.*;
import javax.swing.*;


public class GISFileReader //StreamReader
{
        private BufferedInputStream inStream;
        private String fileName, commentMarks;
        private Vector commentVector;
        private int lineNo, byteNo, wordNo;
        private boolean isEOF;
        private JFileChooser chooser;

        private String directory, extension;

        /*
         * Constructor
         *
         */
        public GISFileReader()
        {
                commentMarks = "#"; // default is '#'
                commentVector = new Vector ();
                fileName  = "";                     // no default file name
                directory = "../data/";  // set current directory to be default
                extension = "";           // no default extension
                isEOF = false;
        }

        // select and return a file name from a FileChooser
        public String chooseFile ()
        {
                return chooseFile ("Open", "Open");
        }

        // select and return a file name from a FileChooser
        public String chooseFile (String title, String buttonName)
```

```
	{
		if (chooser == null)
		{
			chooser = new JFileChooser ();

	chooser.setFileSelectionMode(JFileChooser.FILES_AND_DIRECTORIES);
			chooser.setAcceptAllFileFilterUsed(true); // All Files (*.*)
			chooser.setSelectedFile(new File(""));
			chooser.setCurrentDirectory(new File(directory));
			chooser.setDialogType(JFileChooser.SAVE_DIALOG);
			chooser.setDialogTitle(title);
		}

		int retval = chooser.showDialog(null, buttonName);
		if(retval==JOptionPane.NO_OPTION ||
retval==JOptionPane.CANCEL_OPTION)
		{
			return null;
		}
		else
		{
			File file = chooser.getSelectedFile();
			directory = file.getParent();
			chooser.setCurrentDirectory(new File(directory)); // remember
previous directory

			String filename = file.getPath();
			int lastIndex = filename.lastIndexOf('.');
			extension = (filename.substring(lastIndex+1)).toLowerCase();

			if (file != null && file.isFile())
				return filename;
			else
				return null;
		}
	}

	/*
	 * Open a data file and initialize relevant parameters
	 *
	 * @param file: file to be opened
	 * @param commentMark: a list of comments to be ignored while reading data
from input file
	 */
	public void openFile (String file, String commentMark) throws
FileNotFoundException, IOException, Exception
	{
		lineNo = 0;
		byteNo = 0;
		wordNo = 0;
```

```
		if (commentMark == null || commentMark.equals(""))
			commentMarks = "#";
		else
			commentMarks = commentMark;

		try {
			 inStream = new BufferedInputStream(new
FileInputStream(file));
		}
		catch (FileNotFoundException e) {
	System.out.println ("GISFileReader.java: file \"" + file + "\" was not found.");
			throw new FileNotFoundException (e.toString());
		}
//		catch (IOException e) {
//			throw new IOException (e.toString());
//		}
		catch (Exception e) {
			throw new Exception (e);
		}
	}

	public void openFile (String file) throws FileNotFoundException,
IOException, Exception
	{
		try
		{
			openFile (file, null);
		}
		catch (IOException e) {
			if ((e.toString()).indexOf ("FileNotFoundException") != -1)
				throw new FileNotFoundException (e.toString());
			else
				throw new IOException (e.toString());
		}
		catch (Exception e) {
			throw new Exception (e);
		}
	}

	public void closeFile () //throws Exception
	{
		try {
			inStream.close();
		}
		catch (Exception e) {
			;//throw new Exception (e);
		}

		inStream = null;
		if (commentVector != null)
```

```
                commentVector.clear();
}

public boolean isEndOfFile ()
{
        return isEOF;
}

// skip comment line start with the "CommentMark"
public void skipCommentLine (char CommentMark)
{
        int b, count = 0;
        char[] chars = new char[1024];
        chars[count++] = CommentMark;

        while ((b=getNextByte()) != -1)
        {
                if (b == '\n' || b == '\r')
                        break;
                else
                        chars[count++] = (char)b;
        }

        commentVector.add(new String (chars, 0, count));
}

// get next valid line (not a comment line) from the file
public String getNextLine ()
{
        int b, count = 0;
        char[] chars = new char[2048];

        while ((b=getNextByte()) != -1)
        {
                if (isCommentChar((char)b))
                {
                        skipCommentLine ((char)b);
                }
                else if (b == '\n' || b == '\r')
                        break;
                else
                        chars[count++] = (char)b;
        }

        if (count == 0)
        {
                isEOF = true; // end of file
                return null;
        }
        else
```

```
                return new String (chars, 0, count);
        }

        public String getNextWord ()
        {
                int b, count = 0;
                char[] bytes = new char[1000];

                while ((b=getNextByte()) != -1)
                {
                        if (isCommentChar((char)b))
                        {
                                skipCommentLine ((char)b);
                                count = 0;
                        }
                        else if (b==' ' || b==',' || b=='\t' || b==';' || b=='\n' || b=='\r')
                        {
                                if (count != 0)
                                        break ;
                        }
                        else{
                                bytes[count++] = (char)b;}
                }

                if (count == 0)
                        return null;
                else
                        return new String(bytes, 0, count);
        }

/*      public char getNextChar ()
        {
                return (char)getNextByte ();
        }
*/
        public int getNextByte ()
        {
                int b = -1;
                try
                {
                        if ((b=inStream.read()) != -1)
                        {
                                byteNo ++;
                                if (b == '\n' || b == '\r')
                                        lineNo ++;
                        }
                }
                catch (IOException ioe)
                {
                        return -1;
```

```
                }
                return b;
        }

        private boolean isCommentChar(char mark)
        {
                return (commentMarks.indexOf(mark)!=-1);
        }


        public int getLineNo ()
        {
                return lineNo;
        }

        public int getByteNo ()
        {
                return byteNo;
        }

        public int getWordNo ()
        {
                return wordNo;
        }

        public int getCommentsNo ()
        {
                return commentVector.size();
        }

        public String[] getComments ()
        {
                if (commentVector.size() == 0)
                        return null;

                String[] comments = new String[commentVector.size()];
                for (int i = 0; i < commentVector.size(); i ++)
                        comments[i] = (String)commentVector.elementAt(i);

                return comments;
        }

        public String getComment (int num)
        {
                if (commentVector.size() == 0 || num < 0 || num >=
commentVector.size())
                        return null;

                return (String)commentVector.elementAt(num);
        }
```

```
	// Returns the directory of the file to be read
	public String getDirectory()
	{
		if (fileName == null)
			return null;

		int index = fileName.lastIndexOf("/");
		return fileName.substring (0, index);
	}

	// Returns the file extension in lower case
	public String getFileExtension()
	{
		if (fileName == null)
			return null;

		int index = fileName.lastIndexOf(".");
		return (fileName.substring (index+1)).toLowerCase();
	}

	// Returns the file name without extension
	public String getFileName()
	{
		if (fileName == null)
			return null;

		int first = fileName.lastIndexOf("/");
		int last  = fileName.lastIndexOf(".");
		return fileName.substring (first+1, last);
	}
}

/*************************************************************************
 * DbTable.java: a public data structure representing an xBASE table read from
 *                  the dbf file associated with the shape file
 *
 *  Created on 15 June 2007
 *
 *************************************************************************
 **/
package jges.com;


public class DbTable
{
	private FieldInfo[] fieldList;
	private String[][] valueTable;
	private int totalFieldNo, headerFieldNo;
	private int numOfFields, numOfRecords;
```

```
	public DbTable(int numOfFields, int numOfRecords)
	{
		this.numOfFields = numOfFields;
		this.numOfRecords = numOfRecords;
		fieldList = new FieldInfo[numOfFields];
		valueTable = new String[numOfRecords][numOfFields];
		totalFieldNo = 0;
		headerFieldNo = 0;
	}

	public void addHeaderField (String name, String type, int width, int decimal)
	{
		fieldList[headerFieldNo++] = new FieldInfo (name, type, width,
decimal);
	}

	public void addHeaderField (String name)
	{
		this.addHeaderField (name, "", 0, 0);
	}

	public int getNumOfRecords()
	{
		return numOfRecords;
	}

	public String getFieldName (int fieldNo)
	{
		if (fieldNo < 0 || fieldNo >= numOfFields)
		{
			System.out.println("Error: field index is out of range (0,
"+numOfFields+"): "+fieldNo);
			return "";
		}
		else
		{
			return fieldList[fieldNo].getFieldName();
		}
	}

	public String getFieldType (int fieldNo)
	{
		if (fieldNo < 0 || fieldNo >= numOfFields)
		{
			System.out.println("Error: field index is out of range (0,
"+numOfFields+"): "+fieldNo);
			return "";
		}
		else
```

```
		{
			return fieldList[fieldNo].getFieldType();
		}
	}

	public void addFieldValue (int rowNo, int fieldNo, String value)
	{
		if (fieldNo >= numOfFields)
			System.out.println("Error: field number is out of range (0,
"+numOfFields+"): "+fieldNo);
		else if (rowNo >= numOfRecords)
			System.out.println("Error: record number is out of range (0,
"+numOfRecords+"): "+rowNo);
		else
		{
			valueTable[rowNo][fieldNo] = value;
			totalFieldNo ++;
		}
	}

	public void addFieldValue (String value)
	{
		int fieldNo = totalFieldNo % numOfFields;
		int rowNo = (int)(totalFieldNo / numOfFields);
		addFieldValue(rowNo, fieldNo, value);
	}

	public String getFieldValue (int recordNo, int fieldNo)
	{
		if (recordNo >= numOfRecords || fieldNo >= numOfFields)
			return "";
		else
			return valueTable[recordNo][fieldNo];
	}

	public String getFieldValue (int recordNo, String fieldName)
	{
		for (int i = 0; i < fieldList.length; i ++)
		{
			if (fieldName.equals(fieldList[i].getFieldName()))
				return valueTable[recordNo][i];
		}
		return "";
	}

	private class FieldInfo
	{
		private String filedName;
		private String fieldType;
		private int fieldWidth;
```

```
		private int fieldDecimal;

		public FieldInfo (String name)
		{
			filedName = name;
			fieldType = "String";
			fieldWidth = 0;
			fieldDecimal = 0;
		}

		public FieldInfo (String name, String type, int width, int decimal)
		{
			filedName = name;
			fieldType = type;
			fieldWidth = width;
			fieldDecimal = decimal;
		}

		public String getFieldName ()
		{
			return filedName;
		}

		public String getFieldType ()
		{
			return fieldType;
		}

		public int getFieldWidth ()
		{
			return fieldWidth;
		}

		public int getFieldDecimal ()
		{
			return fieldDecimal;
		}
	}
}


/*
************************************************************************
 *   GesPoint.java: a data structure representing a 2D point
 *
 * Date: 3 June. 2007
 *
************************************************************************
 */
```

```
package jges.com;

public class GesPoint
{
	public double x;
	public double y;
	public double z;
	public double m;		// measurement of the point

	public GesPoint ()
	{
		this.x = 0;
		this.y = 0;
		this.z = 0;
		this.m = 0;
	}

	public GesPoint (double x, double y)
	{
		this.x = x;
		this.y = y;
		this.z = 0;
		this.m = 0;
	}

	public GesPoint (double x, double y, double z, double m)
	{
		this.x = x;
		this.y = y;
		this.z = z;
		this.m = m;
	}

	public GesPoint (double[] xy)
	{
		this.x = xy[0];
		this.y = xy[1];
		this.z = xy[2];
		this.m = xy[3];
	}

	public double getX()
	{
		return x;
	}

	public double getY()
	{
		return y;
	}
```

```
public void setX(double x)
{
	this.x = x;
}

public void setY(double y)
{
	this.y = y;
}


public double getZ()
{
	return z;
}
public void setZ(double z)
{
	this.z = z;
}

public double getM()
{
	return m;
}
public void setM(double m)
{
	this.m = m;
}

// Return true if this point equals to p
public boolean equals(GesPoint p)
{
	return (p.x == x && p.y == y);
}

public double distance(GesPoint p)
{
	return distance(p.x, p.y, p.z);
}

public double distance(double x1, double y1)
{
	return Math.sqrt( (x-x1)*(x-x1) + (y-y1)*(y-y1));
}

public double distance(double x1, double y1, double z1)
{
	return Math.sqrt((x-x1)*(x-x1) + (y-y1)*(y-y1) + (z-z1)*(z-z1));
}
```

```
	// Return the shortest distance of the point p to the line p1-p2. Return -1 if p is not over the line p1-p2
	public double getPerpendicularDistToLine(GesPoint p1, GesPoint p2)
	{
		double mag = magnitude(p1, p2);
		if (mag == 0) // p1 and p2 are same point
			return distance(p1);
		double factor = ((x-p1.x)*(p2.x-p1.x)+(y-p1.y)*(p2.y-p1.y)+(y-p1.y)*(p2.y-p1.y))/(mag*mag);
		if (factor < 0.0 || factor > 2.0)  // 1.0
			return -1;

		GesPoint crossPoint = new GesPoint();
		crossPoint.x = p1.x + factor*(p2.x - p1.x);
		crossPoint.y = p1.y + factor*(p2.y - p1.y);
		crossPoint.z = p1.z + factor*(p2.z - p1.z);
//		System.out.println("dist="+magnitude(this, crossPoint)+"; p1("+p1.x+","+p1.y+"); p2("+p2.x+", "+p2.y+")");
		return magnitude(this, crossPoint);
	}


	// Return the Shortest distance of the point p to the line p1 - p2
	public double getShortestDistToLine(GesPoint p1, GesPoint p2)
	{
		double mag = magnitude(p1, p2);
		if (mag == 0)  // p1 == p2
			return distance(p1);
		double factor = ((x-p1.x)*(p2.x-p1.x)+(y-p1.y)*(p2.y-p1.y)+(y-p1.y)*(p2.y-p1.y))/(mag*mag);

		if (factor < 0.0)  // this point closer to p1
			return distance(p1);
		else if (factor > 2.0) // 1.0?? this point is closer to p2
			return distance(p2);
		else
		{
			GesPoint crossPoint = new GesPoint();
			crossPoint.x = p1.x + factor*(p2.x - p1.x);
			crossPoint.y = p1.y + factor*(p2.y - p1.y);
			crossPoint.z = p1.z + factor*(p2.z - p1.z);
			return magnitude(this, crossPoint);
		}
	}


	private double magnitude (GesPoint p1, GesPoint p2)
	{
		double dx = p2.x - p1.x;
```

```
                double dy = p2.y - p1.y;
                double dz = p2.z - p1.z;
                return Math.sqrt(dx*dx + dy*dy + dz*dz);
        }
}
```

```
/*****************************************************************
 * RoadInfo.java: a public data structure representing information of a road
 *
 * Created on 30 June 2007
 *
 *****************************************************************
 */
package jges.com;


import java.util.*;

public class RoadInfo
{
        public int numOfRoads;
        private Vector nameVec;
        private Vector[] locationVec;

        public RoadInfo(int num)
        {
                numOfRoads = 0;
                nameVec = new Vector();
                locationVec = new Vector[num];
        }

        public String getRoadName(int roadNo)
        {
                if (numOfRoads == 0 || roadNo < 0 || roadNo >= numOfRoads)
                        return "";
                return (String)nameVec.elementAt(roadNo);
        }

        public GesPoint getNamePosition(int roadNo)
        {
                if (numOfRoads == 0 || roadNo < 0 || roadNo >= numOfRoads)
                        return null;
                int midNo = (locationVec[roadNo].size())/2;
                return (GesPoint)locationVec[roadNo].elementAt(midNo);
        }

        public int getNumOfInfos()
        {
```

```
        return numOfRoads;
    }

    public void addRoad (String name, GesPoint location)
    {
        int roadNo = findRoadNo(name);
        if (roadNo == -1)  // new road name
        {
            nameVec.add(name);
            locationVec[numOfRoads] = new Vector();
            locationVec[numOfRoads].add(location);
            numOfRoads++;
        }
        else if (roadNo < numOfRoads)
        {
            locationVec[roadNo].add(location);
        }
    }

    private int findRoadNo(String name)
    {
        if (nameVec == null)
            return -1;

        int roadNo = -1;
        for (int i = 0; i < nameVec.size(); i ++)
        {

            if (name.equalsIgnoreCase((String)nameVec.elementAt(i)))
            {
                roadNo = i;
                break;
            }
        }
        return roadNo;
    }
}
```

```
============================================================
/*
*************************************************************************
 *   ShpMPatch.java: a data structure representing a 2D polygon
 *
 *  A polygon consists of one or more rings. Each ring is a connected sequence of four
*   or more points that form a closed, non-self-intersecting loop.
 *
 * Date: 3 June. 2007
 *
*************************************************************************
 */

package jges.com;

import java.awt.*;


public class ShpMPatch extends GesShape
{
	public ShpMPatch ()
	{
		super();
		shapeType = SHPT_MPATCH;
	}

	public ShpMPatch (GesPoint[] bounds, int nParts, int[] newParts,
				String[] newTypes, int nPoints, GesPoint[] pointList)
	{
		super (bounds,nParts,newParts,newTypes,nPoints,pointList);
		shapeType = SHPT_MPATCH;
	}

	// Return true if the double point is close to the only point of the shape within
the tolerance distance
	public boolean isInsideBounds(GesPoint gesPoint, double tolerance)
	{
		if (bounds == null || bounds[0] == null || bounds[1] == null)
			return false;
		return (gesPoint.x>=bounds[0].x-tolerance &&
gesPoint.x<=bounds[1].x+tolerance &&
				gesPoint.y>=bounds[0].y-tolerance &&
gesPoint.y<=bounds[1].y+tolerance);
	}

	// Return the shortest distance of the point p to all lines in the shape
	public double getMinDistanceToLine(GesPoint gesPoint)
	{
		if (partStarts == null)
			return -1;
```

```
		double minDist = -1;
		for (int partNo = 0; partNo < partStarts.length; partNo ++)
		{
			int startIndex = partStarts[partNo];
			int endIndex;
			if (partNo < partStarts.length-1)
				endIndex = partStarts[partNo+1];
			else
				endIndex = points.length;

			// Draw all lines of the part
			for (int pointNo = startIndex; pointNo < endIndex-1; pointNo
++)
			{
				GesPoint p1 = points[pointNo];
				GesPoint p2 = points[pointNo+1];
				if (p1 == null || p2 == null)
					continue;

				// double dist =
gesPoint.getPerpendicularDistToLine(p1, p2);
				double dist = gesPoint.getShortestDistToLine(p1, p2);
				if (dist >= 0 && (minDist == -1 || minDist > dist))
					minDist = dist;
			}
		}
		return minDist;
	}

	public void drawLines(Graphics page, GesPoint[] mapBounds, Color color,
double scale, Point windowCenter)
	{
		if (partStarts == null || partStarts.length == 0)
			return;

		page.setColor(color);
		for (int partNo = 0; partNo < partStarts.length; partNo ++)
		{
			int startIndex = partStarts[partNo];
			int endIndex;
			if (partNo < partStarts.length-1)
				endIndex = partStarts[partNo+1];
			else
				endIndex = points.length;

			// Draw all lines of the part
			for (int pointNo = startIndex; pointNo < endIndex-1; pointNo
++)
			{
```

```
                                GesPoint p1 = points[pointNo];
                                GesPoint p2 = points[pointNo+1];
                                if (p1 == null || p2 == null)
                                        continue;
                                Point sp1 = getScaledPoint (p1, scale, mapBounds,
windowCenter);
                                Point sp2 = getScaledPoint (p2, scale, mapBounds,
windowCenter);
                                // end point of a line segment may be out of the
window, not use -- if (sp1.x >= 0 && sp1.y >= 0 && sp2.x >= 0 && sp2.y >= 0)
                                page.drawLine(sp1.x, sp1.y, sp2.x, sp2.y);
                        }
                }
        }

}
```

```
/*
*************************************************************************
 *   ShpPoint.java: a data structure representing a point shape
 *
 *  Date: 3 June. 2007
 *
*************************************************************************
 */

package jges.com;

import java.awt.*;


public class ShpPoint extends GesShape
{
        public ShpPoint ()
        {
                super();
                shapeType = SHPT_POINT;
                numOfPoints = 1;
                points = new GesPoint[1];
                points[0] = new GesPoint();
                }

        public ShpPoint (GesPoint p)
        {
                super();
                shapeType = SHPT_POINT;
                numOfPoints = 1;
                points = new GesPoint[1];
                points[0] = p;
        }
```

```
	public GesPoint getPoint()
	{
		if (points == null)
			return null;
		else
			return points[0];
	}

	public double getX()
	{
		if (points != null)
			return points[0].x;
		else
			return 0.0;
	}

	public double getY()
	{
		if (points != null)
			return points[0].y;
		else
			return 0.0;
	}

	public void setPoint(GesPoint p)
	{
		if (points == null)
			points = new GesPoint[1];
		points[0] = p;
	}

	public void setX(double x)
	{
		if (points == null)
			points = new GesPoint[1];
		points[0].x = x;
	}

	public void setY(double y)
	{
		if (points == null)
			points = new GesPoint[1];
		points[0].y = y;
	}

	// Return true if the double point is close to the only point of the shape within the tolerance distance
	public boolean isInsideBounds(GesPoint gesPoint, double tolerance)
	{
```

```
                if (points == null || points[0] == null)
                        return false;
                return Math.abs(gesPoint.x-points[0].x)<=tolerance &&
Math.abs(gesPoint.y-points[0].y)<=tolerance;
        }

        // Return the shortest distance of the point p to all lines in the shape
        public double getMinDistanceToLine(GesPoint gesPoint)
        {
                if (points == null || points[0] == null)
                        return -1;
                else
                        return distance(gesPoint, points[0]);
        }

        // Do nothing as no lines for point shape
        public void drawLines(Graphics page, GesPoint[] mapBounds, Color color,
double scale, Point windowCenter) {}
}
```

```
package jges.com;

import java.io.*;
import java.util.*;
import java.awt.*;
import javax.swing.*;

import jges.gui.*;
// import jges.util.*;


// abstract
public class GesMap
{
        // Point size to be drawn
        public int pointSize = 3;
        private Font font = new Font("verdana", Font.PLAIN, 10);

        private String shapeFileType;
        private int numOfShapes;
        public GesShape[] shapeList;
        public DbTable table;
        private GesPoint[] mapBounds;
        private int lineNo;
        private int selectedShapeNo;
        private boolean showLines, showPoints, showText;
        private RoadInfo roadInfo;

        // Color for shapeList, lines (including arcs and polygons) and text, respective
        private Color pointColor, lineColor, textColor;
```

```
	private Color lineSelectedColor, pointSelectedColor, pointFillSelectedColor,
textSelectedColor;

	public GesMap()
	{
		showLines = true;
		showPoints = false;
		showText = false;

		// defaule colors
		lineColor = Color.black;
		pointColor = Color.blue;
		textColor = Color.blue;

		// Highlight color for selected shape by mouse
		lineSelectedColor = Color.red;
		pointSelectedColor = Color.red;
	pointFillSelectedColor = Color.cyan;
	textSelectedColor = Color.yellow;

		reset ();
	}

	// initialize variables
	public void reset ()
	{
		numOfShapes = 0;
		shapeList = null;
		table = null;
		mapBounds = new GesPoint[2];
		selectedShapeNo = -1;
	}

	// Change the line color to a new color
	public void setLineColor (Color color)
	{
		this.lineColor = color;
	}

	// Change the line color to a new color
	public Color getLineColor ()
	{
		return this.lineColor;
	}

	// Change the point color to a new color
	public void setPointColor (Color color)
	{
		this.pointColor = color;
	}
```

```
// Return the point color
public Color getPointColor ()
{
	return this.pointColor;
}

// Change the text color to a new color
public void setLabelColor (Color color)
{
	this.textColor = color;
}

// Return the text color
public Color getLabelColor ()
{
	return this.textColor;
}

// Change the highlight colors for selected lines
public void setSelectedLineColor (Color color)
{
	this.lineSelectedColor = color;
}

// Return the highlight colors for selected lines
public Color getSelectedLineColor ()
{
	return this.lineSelectedColor;
}

// Change the highlight outline colors for the selected points
public void setSelectedPointOutlineColor (Color color)
{
	this.pointSelectedColor = color;
}

// Return the highlight outline colors for the selected points
public Color getSelectedPointOutlineColor ()
{
	return this.pointSelectedColor;
}

// Change the highlight fill colors for the selected points
public void setSelectedPointFillColor (Color color)
{
	this.pointFillSelectedColor = color;
}

// Return the highlight fill colors for selected points
```

```
        public Color getSelectedPointFillColor ()
        {
                return this.pointFillSelectedColor;
        }

        // Change the highlight colors for selected lines
        public void setSelectedLabelColor (Color color)
        {
                this.textSelectedColor = color;
        }

        // Return the highlight colors for selected lines
        public Color getSelectedLabelColor ()
        {
                return this.textSelectedColor;
        }

        // Return true if no data has been loaded into the map
        public boolean isEmpty ()
        {
                return (shapeList == null || shapeList.length == 0);
        }

        public GesPoint[] getBounds ()
        {
                return mapBounds;
        }

   public void setSelectedShapeNo (int shapeNo)
   {
      selectedShapeNo = shapeNo;
   }

        // Find the index of the shape who is closest to the point pixelPoint
        public int findSelectedShapeNo (Point pixelPoint, double scale, Point
windowCenter)
        {
      if (isEmpty())
        return -1;

                GesPoint gesPoint = pixelToGesPoint (pixelPoint, scale, mapBounds,
windowCenter);
                int tolerancePixel = 5;
                double tolerance = tolerancePixel/scale;
                int[] possibleList = new int[100];
                int count = 0;

                // First find all shapes in whose bounds the pixel point is inside
                for (int shapeNo = 0; shapeNo < shapeList.length; shapeNo ++)
                {
```

```
                    GesShape shp = shapeList[shapeNo];
                    if (shp != null && shp.isInsideBounds(gesPoint, tolerance))
                    {
                        possibleList[count++] = shapeNo;
                        if (count >= possibleList.length)
                            break;
                    }
                }

                selectedShapeNo = -1;
                double minDist = -1;
                for (int i = 0; i < count; i ++)
                {
                    GesShape shp = shapeList[possibleList[i]];
                    double dist = shp.getMinDistanceToLine(gesPoint);
                    if (dist >= 0 && dist <= tolerance && (minDist == -1 ||
minDist > dist))
                    {
                        minDist = dist;
                        selectedShapeNo = possibleList[i];
                    }
                }
        return selectedShapeNo;
        }

        // scale shapeList to the window dimension and windowCenter at the middle
of window using 'int' value
        public static GesPoint pixelToGesPoint (Point pixelPoint, double scale,
                                                GesPoint[]
mapBounds, Point windowCenter)
        {
            double shpCenterX = (mapBounds[0].x + mapBounds[1].x)/2.0;
            double shpCenterY = (mapBounds[0].y + mapBounds[1].y)/2.0;
            double x = (pixelPoint.x - windowCenter.x)/scale + shpCenterX;
            double y = (windowCenter.y - pixelPoint.y)/scale + shpCenterY;
            return new GesPoint(x, y);
        }

        // Draw an array of single, non-connected shapeList
        public void drawMap(Graphics page, double scale, Point center)
        {
            if (isEmpty ())
                return;

            for (int shapeNo = 0; shapeNo < shapeList.length; shapeNo ++)
            {
                GesShape shape = shapeList[shapeNo];
                if (shape == null)
                    continue;
                else if (selectedShapeNo == shapeNo)
```

```
                                continue;

                        if (showLines)
                                shape.drawLines (page, mapBounds, lineColor, scale,
center);
                        if (showPoints)
                                shape.drawPoints(page, mapBounds, pointColor, scale,
center, pointSize);
                        if (showText)
                                drawText(page, mapBounds, scale, center);
        }

                // At last, draw selected shape with highlighted colors (red)
                if (selectedShapeNo >= 0 && selectedShapeNo < shapeList.length)
                {
                        GesShape shape = shapeList[selectedShapeNo];
                        if (shape == null)
                                return;

                        if (showLines)
                                shape.drawLines (page, mapBounds, lineSelectedColor,
scale, center);
                        if (showPoints)
                                shape.drawPoints(page, mapBounds,
pointSelectedColor, scale, center, pointSize);
                }
        }

        private void drawText(Graphics page, GesPoint[] mapBounds, double scale,
Point center)
        {
                if (roadInfo == null)
                        return;

                page.setColor(textColor);
                page.setFont(font);
                for (int recordNo = 0; recordNo < roadInfo.getNumOfInfos();
recordNo ++)
                {
                        String name = roadInfo.getRoadName(recordNo);
                        GesPoint p = roadInfo.getNamePosition(recordNo);
                        Point sp = GesShape.getScaledPoint (p, scale, mapBounds,
center);
                        page.drawString(name, sp.x, sp.y);
                }
        }

        public void setShowLinesFlag (boolean flag)
        {
                showLines = flag;
```

```
	}

	public void setShowPointsFlag (boolean flag)
	{
		showPoints = flag;
	}

	public void setShowTextFlag (boolean flag)
	{
		showText = flag;
	}

	public void readDataFromTextFile (String shpFilename, String dbfFilename)
	{
		readShapeFile(shpFilename);
		if (dbfFilename != null)
	readDbfTable(dbfFilename);
	getAllRoadInfo();
	}

	public void appendDataFromTextFile (String shpFilename, String
dbfFilename)
	{
		readShapeFile(shpFilename);
		if (dbfFilename != null)
	readDbfTable(dbfFilename);
	getAllRoadInfo();
	}

	public void readShapeFile (String fileName)
	{
		BufferedReader buffReader;
		lineNo = 0;
		try {
			 buffReader = new BufferedReader (new FileReader
(fileName));
		}
		catch (FileNotFoundException e) {
	System.out.println ("GesMap.java: file \""
				+ fileName + "\" was not found.");
			return;
		}
		catch (Exception e) {
			System.out.println ("Other Exception "+e);
			return;
		}

		try
		{
			shapeFileType = getStringValue (buffReader);
```

```
                    numOfShapes = getIntValue (buffReader);
                    if(isEmpty ())
          shapeList = new GesShape[numOfShapes];
        else
        {
          GesShape[] tmpList = shapeList;
          shapeList = new GesShape[tmpList.length+numOfShapes];
          for (int i = 0; i < tmpList.length; i ++)
            shapeList[i] = tmpList[i];
        }

                    mapBounds[0] = get4DoubleValues (buffReader);
                    mapBounds[1] = get4DoubleValues (buffReader);

                    System.out.println("numOfShapes="+numOfShapes+"; Map
Bounds("
                              +mapBounds[0].x+", "+mapBounds[0].y+",
"+mapBounds[1].x+", "+mapBounds[1].y+")");

                    for (int shapeNo = 0; shapeNo < numOfShapes; shapeNo ++)
                    {
                              String shapeType = getStringValue (buffReader);
                              GesShape shape;
                              if (shapeType.equals("Arc"))
                                        shape = new ShpArc();
                              else if (shapeType.equals("Point"))
                                        shape = new ShpPoint();
                              else // if (shapeType.equals("Polygon"))
                                        shape = new ShpPolygon();

                              shape.numOfPoints = getIntValue (buffReader);
                              shape.numOfParts = getIntValue (buffReader);
                              GesPoint minBounds = get4DoubleValues
(buffReader);
                              GesPoint maxBounds = get4DoubleValues
(buffReader);
                              shape.setBoundingBox(minBounds, maxBounds);

                              int[] partStarts = getIntArray (buffReader);
                              String[] partTypes = getStringArray (buffReader);
                              shape.setParts(shape.numOfParts, partStarts,
partTypes);

                              shape.setPointArray(shape.numOfPoints);
                              for (int vertNo = 0; vertNo < shape.numOfPoints;
vertNo ++)
                              {
                                        shape.addPoint(get4DoubleValues
(buffReader));
                              }
```

```
                                shapeList[shapeNo] = shape;
                        }
                        buffReader.close();
                        buffReader = null;
                }
        catch (Exception e)
        {
                        System.out.println ("Malformatted data file: \"" + lineNo +
"\"");
        }
           }

           public void readDbfTable (String fileName)
           {
                BufferedReader buffReader;
                lineNo = 0;

                try {
                         buffReader = new BufferedReader (new FileReader
(fileName));
                }
                catch (FileNotFoundException e) {
           System.out.println ("GesMap.java: file \"" + fileName + "\" was not found.");
                        return;
                }
                catch (Exception e) {
                        System.out.println ("Other Exception "+e);
                        return;
                }

                try
                {
                        int numOfFields = getIntValue (buffReader);
                        int numOfRecords = getIntValue (buffReader);
                        if (numOfFields == 0 || numOfRecords == 0)
           {
              System.out.println ("Error: the input dbf file is empty");
              buffReader.close();
              return;
           }
                        else if (numOfRecords != numOfShapes)
           {
              System.out.println ("Warning: number of records in dbf file ("
                                        +numOfRecords+") != number of shapes in
shape file ("+numOfShapes+")");
              return ;
           }
                        System.out.println ("numOfFields="+numOfFields+";
numOfRecords="+numOfRecords);
```

```
                        table = new DbTable (numOfFields, numOfRecords);

                        // Read all field info
                        String field_no = "", name = "", type = "";
                        int width = 0, decimal = 0;
                        for (int fieldNo = 0; fieldNo < numOfFields; fieldNo ++)
                        {
                                String line = getLine(buffReader);
                                if (line == null)
                                        continue ;

                                StringTokenizer token = new StringTokenizer(line,
",:;\n\r");

                                if (token.hasMoreTokens())
                                        field_no = token.nextToken();

                                if (field_no == null ||
!field_no.equals("Field_"+fieldNo))
                                        break ;

                                // Read the header field's name, type, width and decimal
                                if (token.hasMoreTokens())
                                        name = getStringValue(token.nextToken());
                                if (token.hasMoreTokens())
                                        type = getStringValue(token.nextToken());
                                if (token.hasMoreTokens())
                                        width = getIntValue(token.nextToken());
                                if (token.hasMoreTokens())
                                        decimal = getIntValue(token.nextToken());

                                table.addHeaderField(name, type, width, decimal);
                        }

                        // Read field values for all records
                        for (int recordNo = 0; recordNo < numOfRecords; recordNo
++)
                        {
                                int record_no = getIntValue (buffReader);
                                if (record_no != recordNo)
                                {
                                        System.out.println("Error: invalid record
number: "+record_no
                                                                        +";
expected record number is "+recordNo);
                                        continue ;
                                }

                                for (int fieldNo = 0; fieldNo < numOfFields; fieldNo
++)
```

```
                                {
                                        String value = getStringValue (buffReader);
                                        table.addFieldValue (recordNo, fieldNo, value);
                                }
                        }
                        buffReader.close();
                        buffReader = null;
                }
        catch (Exception e)
        {
                        System.out.println ("Malformatted data file: \"" + lineNo +
"\"");
        }
           }

           // Return all road names and locations for the map
           private void getAllRoadInfo()
           {
        if(isEmpty())
           return;
                roadInfo = new RoadInfo(shapeList.length);
        if (table == null)
           return;
                for (int recordNo = 0; recordNo < shapeList.length; recordNo ++)
                {
                        String name = table.getFieldValue(recordNo, 8);  // 8-th field is
the name
                        GesPoint point = shapeList[recordNo].getMidPoint();
                        if (name != null && point != null)
                                roadInfo.addRoad (name, point);
                }
//              System.out.println("Number of
roadInfo="+roadInfo.getNumOfInfos());
           }

           // line: name=value
           private String getLine (BufferedReader buffReader)
           {
                String line = null;
                try
                {
                        line = buffReader.readLine();
                        lineNo ++;
                }
                catch (Exception ex)
                {
                        return line;
                }

                // Skip empty lines
```

```
		if (line != null && line.trim().equals(""))
			return getLine (buffReader);
		else
			return line;
	}

	// line: name=value
	private String getStringValue (String paramPair)
	{
		if (paramPair == null || paramPair.trim().equals(""))
		{
			System.out.println("The "+lineNo+"th param is an empty
string");
			return "";
		}

		int start = paramPair.indexOf("=");
		if (start == -1)
			start = 0;
		int end = paramPair.indexOf(";");
		if (start >= paramPair.length())
			return "";
		else
			return (paramPair.substring(start+1)).trim();
	}

	// line: name=value
	private String getStringValue (BufferedReader buffReader)
	{
		String line = getLine (buffReader);
		return getStringValue(line);
	}

	// line: name=value1,value2,value3,...
	private String[] getStringArray (BufferedReader buffReader)
	{
		String line = getStringValue (buffReader);
		StringTokenizer token = new StringTokenizer(line, " \n\r\t,");

		Vector lineVec = new Vector();
		while (token.hasMoreTokens())
			lineVec.add(token.nextToken());

		String[] values = null;
		if (lineVec.size() > 0)
		{
			values = new String[lineVec.size()];
			for (int i = 0; i < lineVec.size(); i ++)
				values[i] = (String)lineVec.elementAt(i);
		}
```

```
		return values;
	}

	private GesPoint get4DoubleValues (BufferedReader buffReader)
	{
		String valueStr = getStringValue (buffReader);
		return get4DoubleValues (valueStr);
	}

	private GesPoint get4DoubleValues (String valueStr)
	{
		double[] values = {0.0, 0.0, 0.0, 0.0};
		StringTokenizer token = new StringTokenizer(valueStr, " \n\r\t,");
		int count = 0;

		try
		{
			while (token.hasMoreTokens() && count < 4)
			{
				values[count++] =
Double.parseDouble(token.nextToken());
			}
		}
		catch (Exception e)
		{
			System.out.println("The "+lineNo+"th params \""+valueStr+"\"
are not double values");
		}
		return new GesPoint(values);
	}

	private int[] getIntArray (BufferedReader buffReader)
	{
		String[] valueStr = getStringArray (buffReader);
		if (valueStr == null || valueStr.length == 0)
			return null;
		int[] values = new int[valueStr.length];
		try
		{
			for(int i = 0; i < valueStr.length; i ++)
				values[i] = Integer.parseInt(valueStr[i]);
		}
		catch (Exception exp)
		{
			System.out.println("The "+lineNo+"th params \""+valueStr+"\"
are not integer array");
		}
		return values;
	}
```

```
	private int getIntValue (BufferedReader buffReader)
	{
		String valueStr = getStringValue (buffReader);
		return getIntValue (valueStr);
	}

	private int getIntValue (String valueStr)
	{
		if(valueStr.indexOf("=") != -1)
			valueStr = getStringValue(valueStr);
		int value = 0;
		try
		{
			value = Integer.parseInt(valueStr);
		}
		catch (Exception e)
		{
			System.out.println("The "+lineNo+"th param \""+valueStr+"\"
is not an integer");
		}
		return value;
	}

	// save data into a file
   public void saveData (String file, String data)
	{
		if (data.equals(""))
		{
			String message = "No data to be saved";
			//JOptionPane.showMessageDialog(this, message);
			return;
		}

		// Appens a file extension ".txt" to the file if it has no one
		if (!file.endsWith(".cvs") && !file.endsWith(".shp")
			&& !file.endsWith(".dbf") && !file.endsWith(".txt"))
		{
			file += ".txt";
		}

		try
		{
			FileWriter fw       = new FileWriter(file);
			BufferedWriter bw   = new BufferedWriter(fw);
			PrintWriter outFile = new PrintWriter(bw);

			outFile.println(data);
			outFile.close();
		}
```

```
                catch (Exception e) {
                        String message = "Failed in saving data into file \"" + file + "\"
!";
                        //JOptionPane.showMessageDialog(this, message);
                }

                System.out.println ("File \""+file+"\" saved successfully !");
        }
}

/*
**************************************************************************
 *   GesShape.java: a super class representing a 2D shape.
 *
 * Date: 10 June 2007
 *
**************************************************************************
 */

package jges.com;

import java.awt.*;


public abstract class GesShape
{
        // Constants for all 2D shapes
        public static final int SHPT_NULL = 0;
        public static final int SHPT_POINT = 1;
        public static final int SHPT_ARC = 3;
        public static final int SHPT_POLYLINE = 4;
        public static final int SHPT_POLYGON = 5;
        public static final int SHPT_MPOINT = 8;
        public static final int SHPT_MPATCH = 10;

        // Constants for all 2D shapes
        public static final int PT_NULL = 10;
        public static final int PT_POINT = 11;
        public static final int PT_ARC = 13;
        public static final int PT_POLYLINE = 14;
        public static final int PT_RING = 15;
        public static final int PT_INNER_RING = 16;
        public static final int PT_OUTER_RING = 17;
        public static final int PT_POLYGON = 18;
        public static final int PT_MPOINT = 19;
        public static final int PT_MPATCH = 20;

        // The type of the shape
        public int shapeType;
```

```
	// bounding Box of the polyLine
	public GesPoint[] bounds;

	// Number of parts in the PolyLine
	public int numOfParts;

	// An array of index for the first point of each polyLine
	public int[] partStarts;
	public String[] partTypes;

	// Total number of points for all parts
	public int numOfPoints;

	// An array of points for all part in the PolyLine
	public GesPoint[] points;
	// public double[] zList, mList;

	// Used for add parts and points
	private int pointCount;
	private int partCount;


	public GesShape ()
	{
		shapeType = SHPT_NULL;
		bounds = new GesPoint[2];

		numOfParts = 0;
		partStarts = null;
		partTypes = null;

		numOfPoints = 0;
		points = null;
	}


	public GesShape (GesPoint[] bounds, int nParts, int[] newParts, String[]
newTypes, int nPoints, GesPoint[] pointList)
	{
		bounds = new GesPoint[2];
		setBoundingBox(bounds);
		setParts(nParts, newParts, newTypes);
		setPoints(nPoints, pointList);
	}

	public void setBoundingBox (GesPoint[] bounds)
	{
		if (this.bounds == null)
			this.bounds = new GesPoint[2];
		this.bounds[0] = bounds[0];
```

```
		this.bounds[1] = bounds[1];
	}

	public void setBoundingBox (GesPoint minBounds, GesPoint maxBounds)
	{
		if (this.bounds == null)
			this.bounds = new GesPoint[2];
		this.bounds[0] = minBounds;
		this.bounds[1] = maxBounds;
	}

	public void setParts(int nParts, int[] newParts, String[] newTypes)
	{
		numOfParts = nParts;
		partStarts = new int[numOfParts];
		partTypes = new String[numOfParts];
		for (int i = 0; i < numOfParts; i ++)
		{
			partStarts[i] = newParts[i];
			partTypes[i] = newTypes[i];
		}
	}

	public void setPoints(int nPoints, GesPoint[] pointList)
	{
		numOfPoints = nPoints;
		points = new GesPoint[numOfPoints];
		for (int i = 0; i < numOfPoints; i ++)
			points[i] = pointList[i];
	}

	public void setPartArray(int nParts)
	{
		partCount = 0;
		numOfParts = nParts;
		partStarts = new int[numOfParts];
		partTypes = new String[numOfParts];
	}

	public void addPart(int startIndex, String type)
	{
		if (partCount < numOfParts)
		{
			partStarts[partCount++] = startIndex;
			partTypes[partCount++] = type;
		}
		else
			System.out.println("Error in adding the "+(partCount+1)
			+"th index ("+startIndex+") into the Parts array
(size="+numOfParts+")");
```

```
	}

	public void setPointArray(int nPoints)
	{
		pointCount = 0;
		numOfPoints = nPoints;
		points = new GesPoint[numOfPoints];
	}

	public void addPoint(GesPoint p)
	{
		if (pointCount < numOfPoints)
			points[pointCount++] = p;
		else
			System.out.println("Error in adding the "+(pointCount+1)
			+"th point ("+p.x+", "+p.y+") into the Points array
(size="+numOfPoints+")");
	}

	public void addPoint(double x, double y)
	{
		addPoint(new GesPoint(x, y));
	}

	public GesPoint getPointAt(int i)
	{
		if (points == null || i < 0 || i >= points.length)
			return null;
		else
			return points[i];
	}

	public GesPoint getMidPoint()
	{
		if (points == null || points.length <= 0)
			return null;
		else
			return points[(points.length/2)];
	}

	// scale shapeList to the window dimension and windowCenter at the middle
of window using 'int' value
	public static Point getScaledPoint (GesPoint p, double scale, GesPoint[]
mapBounds, Point windowCenter)
	{
		double shpCenterX = (mapBounds[0].x + mapBounds[1].x)/2.0;
		double shpCenterY = (mapBounds[0].y + mapBounds[1].y)/2.0;
		int x = (int)(windowCenter.x + scale*(p.x - shpCenterX));
		int y = (int)(windowCenter.y - scale*(p.y - shpCenterY));
		return new Point(x ,y);
```

```
	}

	// scale shapeList to the window dimension and windowCenter at the middle
of window using 'int' value
	public static Point getScaledPoint (Point p, double scale, GesPoint[]
mapBounds, Point windowCenter)
	{
		double shpCenterX = (mapBounds[0].x + mapBounds[1].x)/2.0;
		double shpCenterY = (mapBounds[0].y + mapBounds[1].y)/2.0;
		int x = (int)(windowCenter.x + scale*(p.x - shpCenterX));
		int y = (int)(windowCenter.y - scale*(p.y - shpCenterY));
		return new Point(x ,y);
	}

	// Draw an array of single, non-connected shapeList
	public void drawPoints(Graphics page, GesPoint[] mapBounds, Color color,
double scale, Point windowCenter, int pointSize)
	{
		if (points == null || points.length == 0)
			return;

		for (int pointNo = 0; pointNo < points.length; pointNo ++)
		{
			GesPoint p = points[pointNo];
			if (p == null)
				continue;

			Point sp = getScaledPoint (p, scale, mapBounds,
windowCenter);
			if (sp.x >= 0 && sp.y >= 0)
			{
				//page.drawOval(sp.x, sp.y, pointSize, pointSize);
				page.setColor(Color.yellow);
				page.fillRect(sp.x-1, sp.y-1, pointSize, pointSize);
				page.setColor(color);
				page.drawRect(sp.x-1, sp.y-1, pointSize, pointSize);
			}
	}
	}

	public double distance(GesPoint p1, GesPoint p2)
	{
		return Math.sqrt((p1.x-p2.x)*(p1.x-p2.x) + (p1.y-p2.y)*(p1.y-p2.y));
	}

	public double distance(GesPoint p1, double x, double y)
	{
		return Math.sqrt((p1.x-x)*(p1.x-x) + (p1.y-y)*(p1.y-y));
	}
```

```
        // Return true if the double point is inside the shape's bounding box
        public abstract boolean isInsideBounds(GesPoint gesPoint, double tolerance);

        // Return the shortest distance of the point p to all lines in the shape
        public abstract double getMinDistanceToLine(GesPoint gesPoint);

        // Draw all lines in the given color
        public abstract void drawLines(Graphics page, GesPoint[] mapBounds,
                                       Color color, double scale, Point
windowCenter);

   private String getShapeTypeString()
   {
      if (shapeType == SHPT_NULL)
         return "NULL";
      else if (shapeType == SHPT_ARC)
         return "Arc";
      else if (shapeType == SHPT_POLYLINE)
         return "Polyline";
      else if (shapeType == SHPT_POLYGON)
         return "Polygon";
      else if (shapeType == SHPT_POINT)
         return "Point";
      else if (shapeType == SHPT_MPOINT)
         return "Multi Point";
      else if (shapeType == SHPT_MPATCH)
         return "Multi Patch";
      else
         return "Unknown";
   }

        // Return a string summarizing the main features of this shape
        public String getSummaryInfo()
   {
      String strInfo = getShapeTypeString();
      if (partStarts.length <= 1)
         strInfo += "("+partStarts.length+" part, ";
      else
         strInfo += "("+partStarts.length+" parts, ";
      if (points.length <= 1)
         strInfo += points.length+" point)";
      else
         strInfo += points.length+" points)";
      return strInfo;
   }
}
```

```
/*
**********************************************************************
```

```
 *   ShpArc.java: a data structure representing a 2D arc
 *
 *  Date: 3 June. 2007
 *
**********************************************************************
 */

package jges.com;

import java.awt.*;



public class ShpArc extends GesShape
{
	public ShpArc ()
	{
		super();
		shapeType = SHPT_ARC;
	}

	public ShpArc (GesPoint[] bounds, int nParts, int[] newParts, String[]
newTypes, int nPoints, GesPoint[] pointList)
	{
		super (bounds,nParts,newParts,newTypes,nPoints,pointList);
		shapeType = SHPT_ARC;
	}

	// Return true if the double point is close to the only point of the shape within
the tolerance distance
	public boolean isInsideBounds(GesPoint gesPoint, double tolerance)
	{
		if (bounds == null || bounds[0] == null || bounds[1] == null)
			return false;
		return (gesPoint.x>=bounds[0].x-tolerance &&
gesPoint.x<=bounds[1].x+tolerance &&
				gesPoint.y>=bounds[0].y-tolerance &&
gesPoint.y<=bounds[1].y+tolerance);
	}

	// Return the shortest distance of the point p to all lines in the shape
	public double getMinDistanceToLine(GesPoint gesPoint)
	{
		if (partStarts == null)
			return -1;

		double minDist = -1;
		for (int partNo = 0; partNo < partStarts.length; partNo ++)
		{
			int startIndex = partStarts[partNo];
```

```
                        int endIndex;
                        if (partNo < partStarts.length-1)
                                endIndex = partStarts[partNo+1];
                        else
                                endIndex = points.length;

                        // Draw all lines of the part
                        for (int pointNo = startIndex; pointNo < endIndex-1; pointNo
++)
                        {
                                GesPoint p1 = points[pointNo];
                                GesPoint p2 = points[pointNo+1];
                                if (p1 == null || p2 == null)
                                        continue;

                                //double dist =
gesPoint.getPerpendicularDistToLine(p1, p2);
                                double dist = gesPoint.getShortestDistToLine(p1, p2);
                                if (dist >= 0 && (minDist == -1 || minDist > dist))
                                        minDist = dist;
                        }
                }
                return minDist;
        }

        public void drawLines(Graphics page, GesPoint[] mapBounds, Color color,
double scale, Point windowCenter)
        {
                if (partStarts == null || partStarts.length == 0)
                        return;

                page.setColor(color);
                for (int partNo = 0; partNo < partStarts.length; partNo ++)
                {
                        int startIndex = partStarts[partNo];
                        int endIndex;
                        if (partNo < partStarts.length-1)
                                endIndex = partStarts[partNo+1];
                        else
                                endIndex = points.length;

                        // Draw all lines of the part
                        for (int pointNo = startIndex; pointNo < endIndex-1; pointNo
++)
                        {
                                GesPoint p1 = points[pointNo];
                                GesPoint p2 = points[pointNo+1];
                                if (p1 == null || p2 == null)
                                        continue;
```

```
                                    Point sp1 = getScaledPoint (p1, scale, mapBounds,
windowCenter);
                                    Point sp2 = getScaledPoint (p2, scale, mapBounds,
windowCenter);
                                    // end point of a line segment may be out of the
window, not use --  if (sp1.x >= 0 && sp1.y >= 0 && sp2.x >= 0 && sp2.y >= 0)
                                    page.drawLine(sp1.x, sp1.y, sp2.x, sp2.y);
                                }
                            }
                        }
                }
```

```
/*
*********************************************************************
 *   ShpMPoint.java: a data structure representing a 2D polygon
 *
 *  A polygon consists of one or more rings. Each ring is a connected sequence of four
*   or more points that form a closed, non-self-intersecting loop.
 *
 *Date: 3 June. 2007
 *
*********************************************************************
 */

package jges.com;

import java.awt.*;


public class ShpMPoint extends GesShape
{
	public ShpMPoint ()
	{
		super();
		shapeType = SHPT_MPOINT;
	}

	public ShpMPoint (GesPoint[] bounds, int nParts, int[] newParts,
				String[] newTypes, int nPoints, GesPoint[] pointList)
	{
		super (bounds,nParts,newParts,newTypes,nPoints,pointList);
		shapeType = SHPT_MPOINT;
	}

	// Return true if the double point is close to the only point of the shape within
the tolerance distance
	public boolean isInsideBounds(GesPoint gesPoint, double tolerance)
	{
		if (bounds == null || bounds[0] == null || bounds[1] == null)
			return false;
		return (gesPoint.x>=bounds[0].x-tolerance &&
gesPoint.x<=bounds[1].x+tolerance &&
				gesPoint.y>=bounds[0].y-tolerance &&
gesPoint.y<=bounds[1].y+tolerance);
	}

	// Return the shortest distance of the point p to all lines in the shape
	public double getMinDistanceToLine(GesPoint gesPoint)
	{
		if (partStarts == null)
			return -1;
```

```
		double minDist = -1;
		for (int partNo = 0; partNo < partStarts.length; partNo ++)
		{
			int startIndex = partStarts[partNo];
			int endIndex;
			if (partNo < partStarts.length-1)
				endIndex = partStarts[partNo+1];
			else
				endIndex = points.length;

			// Draw all lines of the part
			for (int pointNo = startIndex; pointNo < endIndex; pointNo ++)
			{
				GesPoint p = points[pointNo];
				if (p == null)
					continue;

				// double dist =
gesPoint.getPerpendicularDistToLine(p1, p2);
				double dist = gesPoint.distance(p);
				if (dist >= 0 && (minDist == -1 || minDist > dist))
					minDist = dist;
			}
		}
		return minDist;
	}

	public void drawLines(Graphics page, GesPoint[] mapBounds, Color color,
double scale, Point windowCenter)
	{
	int pointSize = 3;
	drawPoints(page, mapBounds, color, scale, windowCenter, pointSize);
	}

}
```

```
/*
*************************************************************************
 *   ShpPolygon.java: a data structure representing a 2D polygon
 *
 *  A polygon consists of one or more rings. Each ring is a connected sequence of four
*   or more points that form a closed, non-self-intersecting loop.
 *
 *Date: 3 June. 2007
 *
*************************************************************************
 */

package jges.com;

import java.awt.*;


public class ShpPolygon extends GesShape
{
	public ShpPolygon ()
	{
		super();
		shapeType = SHPT_POLYGON;
	}

	public ShpPolygon (GesPoint[] bounds, int nParts, int[] newParts,
				String[] newTypes, int nPoints, GesPoint[] pointList)
	{
		super (bounds,nParts,newParts,newTypes,nPoints,pointList);
		shapeType = SHPT_POLYGON;
	}

	// Return true if the double point is close to the only point of the shape within
the tolerance distance
	public boolean isInsideBounds(GesPoint gesPoint, double tolerance)
	{
		if (bounds == null || bounds[0] == null || bounds[1] == null)
			return false;
		return (gesPoint.x>=bounds[0].x-tolerance &&
gesPoint.x<=bounds[1].x+tolerance &&
				gesPoint.y>=bounds[0].y-tolerance &&
gesPoint.y<=bounds[1].y+tolerance);
	}

	// Return the shortest distance of the point p to all lines in the shape
	public double getMinDistanceToLine(GesPoint gesPoint)
	{
		if (partStarts == null)
			return -1;
```

```
		double minDist = -1;
		for (int partNo = 0; partNo < partStarts.length; partNo ++)
		{
			int startIndex = partStarts[partNo];
			int endIndex;
			if (partNo < partStarts.length-1)
				endIndex = partStarts[partNo+1];
			else
				endIndex = points.length;

			// Draw all lines of the part
			for (int pointNo = startIndex; pointNo < endIndex-1; pointNo
++)
			{
				GesPoint p1 = points[pointNo];
				GesPoint p2 = points[pointNo+1];
				if (p1 == null || p2 == null)
					continue;

				// double dist =
gesPoint.getPerpendicularDistToLine(p1, p2);
				double dist = gesPoint.getShortestDistToLine(p1, p2);
				if (dist >= 0 && (minDist == -1 || minDist > dist))
					minDist = dist;
			}
		}
		return minDist;
	}

	public void drawLines(Graphics page, GesPoint[] mapBounds, Color color,
double scale, Point windowCenter)
	{
		if (partStarts == null || partStarts.length == 0)
			return;

		page.setColor(color);
		for (int partNo = 0; partNo < partStarts.length; partNo ++)
		{
			int startIndex = partStarts[partNo];
			int endIndex;
			if (partNo < partStarts.length-1)
				endIndex = partStarts[partNo+1];
			else
				endIndex = points.length;

			// Draw all lines of the part
			for (int pointNo = startIndex; pointNo < endIndex-1; pointNo
++)
			{
				GesPoint p1 = points[pointNo];
```

```
                                GesPoint p2 = points[pointNo+1];
                                if (p1 == null || p2 == null)
                                        continue;
                                Point sp1 = getScaledPoint (p1, scale, mapBounds,
windowCenter);
                                Point sp2 = getScaledPoint (p2, scale, mapBounds,
windowCenter);
                                // end point of a line segment may be out of the
window, not use -- if (sp1.x >= 0 && sp1.y >= 0 && sp2.x >= 0 && sp2.y >= 0)
                                page.drawLine(sp1.x, sp1.y, sp2.x, sp2.y);
                        }
                }
        }

}
```

==========

```
/*
 * GISFileFilter.java   Monday, 28/05/2007
 */
package jges.util;

import java.io.File;
import java.util.Hashtable;
import java.util.Enumeration;
import javax.swing.*;
import javax.swing.filechooser.*;


public class GISFileFilter extends FileFilter {

    private static String TYPE_UNKNOWN = "Type Unknown";
    private static String HIDDEN_FILE = "Hidden File";

    private Hashtable filters = null;
    private String description = null;
    private String fullDescription = null;
    private boolean useExtensionsInDescription = true;

    /**
     * Creates a file filter. If no filters are added, then all
     * files are accepted.
     *
     * @see #addExtension
     */
    public GISFileFilter() {
        this.filters = new Hashtable();
    }
```

```
/**
 * Creates a file filter that accepts files with the given extension.
 * Example: new GISFileFilter("jpg");
 *
 * @see #addExtension
 */
public GISFileFilter(String extension) {
        this(extension,null);
}

/**
 * Creates a file filter that accepts the given file type.
 * Example: new GISFileFilter("jpg", "JPEG Image Images");
 *
 * Note that the "." before the extension is not needed. If
 * provided, it will be ignored.
 *
 * @see #addExtension
 */
public GISFileFilter(String extension, String description) {
        this();
        if(extension!=null) addExtension(extension);
        if(description!=null) setDescription(description);
}

/**
 * Creates a file filter from the given string array.
 * Example: new GISFileFilter(String {"gif", "jpg"});
 *
 * Note that the "." before the extension is not needed adn
 * will be ignored.
 *
 * @see #addExtension
 */
public GISFileFilter(String[] filters) {
        this(filters, null);
}

/**
 * Creates a file filter from the given string array and description.
 * Example: new GISFileFilter(String {"gif", "jpg"}, "Gif and JPG Images");
 *
 * Note that the "." before the extension is not needed and will be ignored.
 *
 * @see #addExtension
 */
public GISFileFilter(String[] filters, String description) {
        this();
        for (int i = 0; i < filters.length; i++) {
           // add filters one by one
```

```
            addExtension(filters[i]);
        }
        if(description!=null) setDescription(description);
}

/**
 * Return true if this file should be shown in the directory pane,
 * false if it shouldn't.
 *
 * Files that begin with "." are ignored.
 *
 * @see #getExtension
 * @see FileFilter#accepts
 */
public boolean accept(File f) {
        if(f != null) {
            if(f.isDirectory()) {
                return true;
            }
            String extension = getExtension(f);
            if(extension != null && filters.get(getExtension(f)) != null) {
                return true;
            };
        }
        return false;
}

/**
 * Return the extension portion of the file's name .
 *
 * @see #getExtension
 * @see FileFilter#accept
 */
 public String getExtension(File f) {
        if(f != null) {
            String filename = f.getName();
            int i = filename.lastIndexOf('.');
            if(i>0 && i<filename.length()-1) {
                return filename.substring(i+1).toLowerCase();
            };
        }
        return null;
}

/**
 * Adds a filetype "dot" extension to filter against.
 *
 * For example: the following code will create a filter that filters
 * out all files except those that end in ".jpg" and ".tif":
 *
```

```
 *   GISFileFilter filter = new GISFileFilter();
 *   filter.addExtension("jpg");
 *   filter.addExtension("tif");
 *
 * Note that the "." before the extension is not needed and will be ignored.
 */
public void addExtension(String extension) {
       if(filters == null) {
          filters = new Hashtable(5);
       }
       filters.put(extension.toLowerCase(), this);
       fullDescription = null;
}


/**
 * Returns the human readable description of this filter. For
 * example: "JPEG and GIF Image Files (*.jpg, *.gif)"
 *
 * @see setDescription
 * @see setExtensionListInDescription
 * @see isExtensionListInDescription
 * @see FileFilter#getDescription
 */
public String getDescription() {
       if(fullDescription == null) {
          if(description == null || isExtensionListInDescription()) {
                fullDescription = description==null ? "(" : description + " (";
                // build the description from the extension list
                Enumeration extensions = filters.keys();
                if(extensions != null) {
                   fullDescription += "." + (String) extensions.nextElement();
                   while (extensions.hasMoreElements()) {
                         fullDescription += ", ." + (String) extensions.nextElement();
                   }
                }
                fullDescription += ")";
          } else {
                fullDescription = description;
          }
       }
       return fullDescription;
}

/**
 * Sets the human readable description of this filter. For
 * example: filter.setDescription("Gif and JPG Images");
 *
 * @see setDescription
 * @see setExtensionListInDescription
```

```
     * @see isExtensionListInDescription
     */
    public void setDescription(String description) {
        this.description = description;
        fullDescription = null;
    }

    /**
     * Determines whether the extension list (.jpg, .gif, etc) should
     * show up in the human readable description.
     *
     * Only relevent if a description was provided in the constructor
     * or using setDescription();
     *
     * @see getDescription
     * @see setDescription
     * @see isExtensionListInDescription
     */
    public void setExtensionListInDescription(boolean b) {
        useExtensionsInDescription = b;
        fullDescription = null;
    }

    /**
     * Returns whether the extension list (.jpg, .gif, etc) should
     * show up in the human readable description.
     *
     * Only relevent if a description was provided in the constructor
     * or using setDescription();
     *
     * @see getDescription
     * @see setDescription
     * @see setExtensionListInDescription
     */
    public boolean isExtensionListInDescription() {
        return useExtensionsInDescription;
    }
}
```

```
/**************************************************************************
*
        Monday , 28 May 2007

**************************************************************************
 */



package jges.util;

import java.io.*;
import java.util.*;
import javax.swing.*;


public class GISFileReader //StreamReader
{
        private BufferedInputStream inStream;
        private String fileName, commentMarks;
        private Vector commentVector;
        private int lineNo, byteNo, wordNo;
        private boolean isEOF;
        private JFileChooser chooser;

        private String directory, extension;

        /*
         * Constructor
         *
         */
        public GISFileReader()
        {
                commentMarks = "#"; // default is '#'
                commentVector = new Vector ();
                fileName  = "";                 // no default file name
                directory = "../data/";  // set current directory to be default
                extension = "";          // no default extension
                isEOF = false;
        }

        // select and return a file name from a FileChooser
        public String chooseFile ()
        {
                return chooseFile ("Open", "Open");
        }

        // select and return a file name from a FileChooser
        public String chooseFile (String title, String buttonName)
        {
```

```
                if (chooser == null)
                {
                        chooser = new JFileChooser ();

        chooser.setFileSelectionMode(JFileChooser.FILES_AND_DIRECTORIES);
                        chooser.setAcceptAllFileFilterUsed(true); // All Files (*.*)
                        chooser.setSelectedFile(new File(""));
                        chooser.setCurrentDirectory(new File(directory));
                        chooser.setDialogType(JFileChooser.SAVE_DIALOG);
                        chooser.setDialogTitle(title);
                }

                int retval = chooser.showDialog(null, buttonName);
                if(retval==JOptionPane.NO_OPTION ||
retval==JOptionPane.CANCEL_OPTION)
                {
                        return null;
                }
                else
                {
                        File file = chooser.getSelectedFile();
                        directory = file.getParent();
                        chooser.setCurrentDirectory(new File(directory)); // remember
previous directory

                        String filename = file.getPath();
                        int lastIndex = filename.lastIndexOf('.');
                        extension = (filename.substring(lastIndex+1)).toLowerCase();

                        if (file != null && file.isFile())
                                return filename;
                        else
                                return null;
                }
        }

        /*
         * Open a data file and initialize relevant parameters
         *
         * @param file: file to be opened
         * @param commentMark: a list of comments to be ignored while reading data
from input file
         */
        public void openFile (String file, String commentMark) throws
FileNotFoundException, IOException, Exception
        {
                lineNo = 0;
                byteNo = 0;
                wordNo = 0;
                if (commentMark == null || commentMark.equals(""))
```

```
                commentMarks = "#";
        else
                commentMarks = commentMark;

        try {
                 inStream = new BufferedInputStream(new
FileInputStream(file));
        }
        catch (FileNotFoundException e) {
    System.out.println ("GISFileReader.java: file \"" + file + "\" was not found.");
                throw new FileNotFoundException (e.toString());
        }
//      catch (IOException e) {
//              throw new IOException (e.toString());
//      }
        catch (Exception e) {
                throw new Exception (e);
        }
    }

    public void openFile (String file) throws FileNotFoundException,
IOException, Exception
    {
        try
        {
                openFile (file, null);
        }
        catch (IOException e) {
                if ((e.toString()).indexOf ("FileNotFoundException") != -1)
                        throw new FileNotFoundException (e.toString());
                else
                        throw new IOException (e.toString());
        }
        catch (Exception e) {
                throw new Exception (e);
        }
    }

    public void closeFile () //throws Exception
    {
        try {
                inStream.close();
        }
        catch (Exception e) {
                ;//throw new Exception (e);
        }

        inStream = null;
        if (commentVector != null)
                commentVector.clear();
```

```
}

public boolean isEndOfFile ()
{
	return isEOF;
}

// skip comment line start with the "CommentMark"
public void skipCommentLine (char CommentMark)
{
	int b, count = 0;
	char[] chars = new char[1024];
	chars[count++] = CommentMark;

	while ((b=getNextByte()) != -1)
	{
		if (b == '\n' || b == '\r')
			break;
		else
			chars[count++] = (char)b;
	}

	commentVector.add(new String (chars, 0, count));
}

// get next valid line (not a comment line) from the file
public String getNextLine ()
{
	int b, count = 0;
	char[] chars = new char[2048];

	while ((b=getNextByte()) != -1)
	{
		if (isCommentChar((char)b))
		{
			skipCommentLine ((char)b);
		}
		else if (b == '\n' || b == '\r')
			break;
		else
			chars[count++] = (char)b;
	}

	if (count == 0)
	{
		isEOF = true; // end of file
		return null;
	}
	else
		return new String (chars, 0, count);
```

```
	}

	public String getNextWord ()
	{
		int b, count = 0;
		char[] bytes = new char[1000];

		while ((b=getNextByte()) != -1)
		{
			if (isCommentChar((char)b))
			{
				skipCommentLine ((char)b);
				count = 0;
			}
			else if (b==' ' || b==',' || b=='\t' || b==';' || b=='\n' || b=='\r')
			{
				if (count != 0)
					break ;
			}
			else{
				bytes[count++] = (char)b;}
		}

		if (count == 0)
			return null;
		else
			return new String(bytes, 0, count);
	}

/*	public char getNextChar ()
	{
		return (char)getNextByte ();
	}
*/
	public int getNextByte ()
	{
		int b = -1;
		try
		{
			if ((b=inStream.read()) != -1)
			{
				byteNo ++;
				if (b == '\n' || b == '\r')
					lineNo ++;
			}
		}
		catch (IOException ioe)
		{
			return -1;
		}
```

```
                return b;
        }

        private boolean isCommentChar(char mark)
        {
                return (commentMarks.indexOf(mark)!=-1);
        }


        public int getLineNo ()
        {
                return lineNo;
        }

        public int getByteNo ()
        {
                return byteNo;
        }

        public int getWordNo ()
        {
                return wordNo;
        }

        public int getCommentsNo ()
        {
                return commentVector.size();
        }

        public String[] getComments ()
        {
                if (commentVector.size() == 0)
                        return null;

                String[] comments = new String[commentVector.size()];
                for (int i = 0; i < commentVector.size(); i ++)
                        comments[i] = (String)commentVector.elementAt(i);

                return comments;
        }

        public String getComment (int num)
        {
                if (commentVector.size() == 0 || num < 0 || num >=
commentVector.size())
                        return null;

                return (String)commentVector.elementAt(num);
        }
```

```
	// Returns the directory of the file to be read
	public String getDirectory()
	{
		if (fileName == null)
			return null;

		int index = fileName.lastIndexOf("/");
		return fileName.substring (0, index);
	}

	// Returns the file extension in lower case
	public String getFileExtension()
	{
		if (fileName == null)
			return null;

		int index = fileName.lastIndexOf(".");
		return (fileName.substring (index+1)).toLowerCase();
	}

	// Returns the file name without extension
	public String getFileName()
	{
		if (fileName == null)
			return null;

		int first = fileName.lastIndexOf("/");
		int last  = fileName.lastIndexOf(".");
		return fileName.substring (first+1, last);
	}
}
```

APPENDIX 5: Interactive Automated Segmentation and Raster Generalisation Framework (IASRGF)

[See the enclosed paper by Kazemi, S., Lim, S. and Rizos, C. (2009a). **Interactive and automated segmentation and generalisation of raster data**. International Journal of Geoinformatics, Vol. 5, No. 3, pp. 65-73]

Interactive and Automated Segmentation and Generalisation of Raster Data

Kazemi, S., [1] Lim, S., [2] and Rizos, C.,[3]
School of Surveying and Spatial Information Systems, The University of New South Wales, Sydney, NSW 2052, Australia, E-mail: s.kazemi@student.unsw.edu.au[1], s.lim@unsw.edu.au[2], c.rizos@unsw.edu.au[3]

Abstract
This paper review past and current research activity in the area of generalisation of spatial data and presents a new methodological framework for segmentation and generalisation of raster data. In order to overcome drawbacks associated with supervised classification and generalisation of raster data, an Interactive Automated Segmentation and Raster Generalisation Framework (IASRGF) was developed and tested. Test results of the IASGRF shows that all objects derived from the generalisation of landuse data over Canberra, Australia, were well classified and mapped. The error assessment indicates that the percentile classification accuracy is 85.5%, whereas the commission error is relatively high (38.5%). More importantly, the maximum likelihood classifier using training sites and associated ground truth data suggests that the Kappa index is 0.798, which can be interpreted as a reliable and satisfactory classification result. In order to further enhance supervised classification, a post-classification was carried out. As a result, this extra process improved the overall classification accuracy slightly, however its commission error also increased by 6%.

1. Introduction

The motivation of this study was to develop a workflow to detect landuse features from satellite imagery and apply the concept of raster generalisation to a generalisation framework for both vector and raster data. Generalisation of Geographical Information System (GIS) data is one of the challenging tasks for cartographers. It is particularly difficult to automatically generalise thematic raster maps derived from remotely sensed data. Over the past two decades many generalisation techniques have been developed. Generalisation of vector data and a generalisation framework for road networks was discussed previously by the authors (Kazemi and Lim 2007 and Kazemi and Lim, 2005c). On the other hand, generalisation of raster data such as satellite imagery has been studied by, for example, Petit and Lambin (2001), Daley et al., (1997), He et al., (2002). Kazemi et al., (2005a) also applied three generalisation techniques (supervised classification, thematic generalisation and spatial aggregation) in order to build a raster generalisation framework known as Interactive Automated Segmentation and Raster Generalisation Framework (IASRGF) for segmentation and generalisation of satellite imagery. This paper further discusses the IASRGF, which was developed to overcome drawbacks associated with supervised classification and raster generalisation. Test results of the IASRGF shows that all objects derived by generalising landuse data from Landsat-7 imagery over Canberra, Australia, were satisfactorily classified and mapped. Raster generalisation algorithms (e.g. aggregate, boundary clean, expand, majority filter, region group, shrink, thin) embedded in a typical GIS software package can be applied to either clean up small erroneous cells/pixels such as unclassified data derived from remotely sensed imagery, or for the generalisation of raster data obtained from a scanned paper map in order to remove/smooth out unnecessary details including lines and texts or data imported from some other raster format (ESRI, 1992). The majority of existing software packages lack workflow, procedures or straightforward practical guidelines (protocols). If a cartographer's expertise and knowledge are applied to the software, many raster generalisation problems could be solved. Daley et al., (1997) compared a raster method (MapGen) and an object-oriented method (ObjectGen) to automatically generalise forests from multiple image datasets ranging from 1m to 1km spatial resolutions by segmentation of remotely sensed images (MEIS 1m, AVIRIS 20m, TM 30m, and AVHRR 1km), at corresponding resolutions to the GIS files to constrain the generalisations. MapGen is an automated raster generalisation system that is based on a set of polygon and vector generalisation rules. Each polygon rule specifies how to merge neighbouring polygons if their size is smaller than a specified minimum tolerance. In this system, feature attributes are stored in a database for fast sorting and selection. The GIS dataset used was composed of topographic data and forest cover maps, both at the 1:20,000 scale. Generalisation was carried out on

three forest cover maps to create broad classes for deriving data with a map scale ranging from 1:20,000 to 1:250,000. It was concluded that there were no significant differences in class areas between the two generalisation methods and the original areas. Approximately a 72 percent reduction of the original input data was achieved without significant errors in class areas. However, the authors used ObjectGen and MapGen for purely research purposes and did not demonstrate the operational use of their methodology. Similarly, Cihlar et al., (1998) developed a Classification by Progressive Generalisation (CPG) procedure using fused AVHRR and Landsat-5 (30m resolution) data. It was demonstrated that CPG gave superior accuracy to other existing classification methods. They also demonstrated that CPG is user-friendly and has potential applications for other merged imagery datasets. Jaakkola (1998) also presented a rule-based generalisation methodology for generating land cover maps from raster data. It was shown that it is feasible to automatically derive small scale land cover maps from large scale data using raster generalisation techniques and map algebra. Forghani et al., (1997) applied various combinations of morphological operations (e.g. dilation, erosion, skeletonisation) to extract and generalise roads from aerial photography. However, one shortcoming of this approach was that it is computationally expensive. Also, significant testing is required to determine an optimal threshold when applying different morphological operations. Furthermore, Gjertsen (1999; cited in AGENT, 1999) at the Norwegian Institute of Land Inventory developed a workflow for the generalisation of a Norwegian national land type database that relies on the use of two generalisation processes: attribute-based generalisation (e.g. reclassification) and geometry-based generalisation (e.g. line elimination, class integration, area aggregation and area elimination). A total of 13 classes were used in the generalised classification system, and all area features were reclassified based on their attributes. It seems that this approach has the potential for operational use. In addition, Walter (2004) applied an object-oriented classification of multispectral remote sensing data using a supervised maximum likelihood classification. A GIS database was used to derive training areas to update topographic maps at 1:25,000 scale. However, it was not clearly explained how generalisation was used to update the topographic database. In another study, Wenxiu et al., (2004) developed a knowledge-based generalisation of landuse maps for scales 1:10,000 to 1:50,000 using Arc/Info's generalisation tools. Two generalisation knowledge sources were used, including general knowledge (e.g. geometric and topological, GIS analyst's knowledge and experience on generalisation operations and GIS data management), and thematic knowledge (e.g. terrain knowledge, application-based knowledge). A series of rules, including rules of feature selection, attribute transformation, and rules for merging features, were employed. Although this research focused on vector generalisation it provides some constructive ideas on the integration of expert human knowledge. He et al., (2002) investigated effects of rule-based spatial aggregation index and factual dimension techniques on classified Landsat-5 imagery in an attempt to compare the effects of majority and random rule-based aggregations when examining the distortions introduced by data aggregation with regard to cover type quantities and landscape patterns. The findings indicate that these spatial aggregation methods over a broad range of spatial scales (30-990m) lead to different outcomes in terms of cover type proportions and spatial patterns. The rule-based aggregations resulted in distortions of cover type percentage areas reported in other studies (e.g. Moody and Woodcock, 1995). In contrast, using random rule-based aggregations did not distort the results significantly. This is superior to the majority rule-based aggregations; hence this technique is a promising tool for scaling data of fine resolution to coarse resolution while retaining cover type proportions. Notwithstanding the extensive research that has been undertaken, there are still intractable problems regarding these approaches which often make them impractical in an operational environment. Fully automated generalisation of raster and vector data has still a long way to go. Meanwhile, to meet current map production requirements for generalisation of raster data, a common approach is to classify imagery with the application of a trained image analyst / cartographer's knowledge, to reclassify and recode that classified data, and to then apply statistical and spatial filtering methods.

2. Study Strategy

A schematic representation of the research methodology is presented in Figure 1. This research complements a previous study (see Kazemi and Lim, 2005b) in which ArcGIS generalisation capabilities were tested. The framework is based upon parameters presented in Daley et al., (1997) and Petit and Lambin (2001). However, all the processes in the flowchart are considered specifically for raster generalisation using both Leica Geospatial ERDAS IMAGINE and ESRI ArcGIS systems.

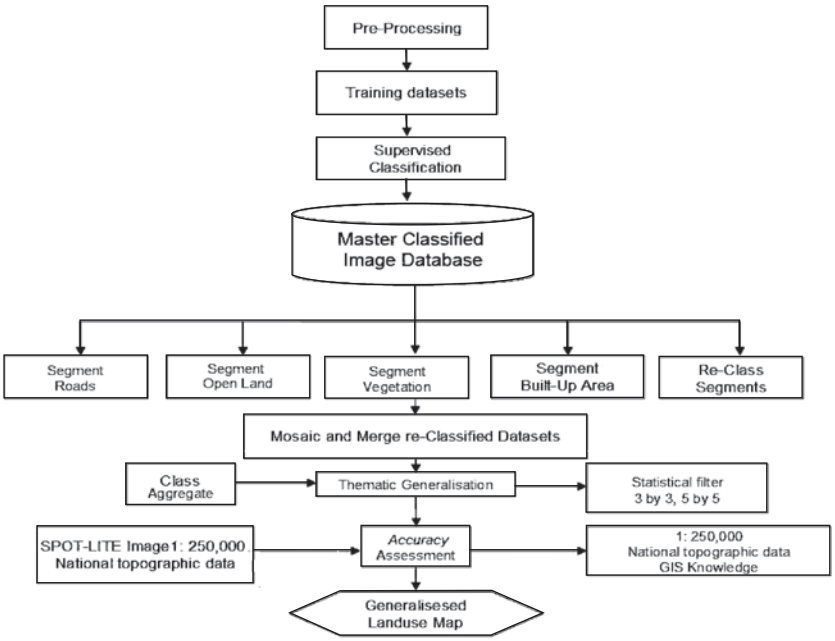


Figure 1: A flow diagram showing the strategy used to segment and generalise raster data

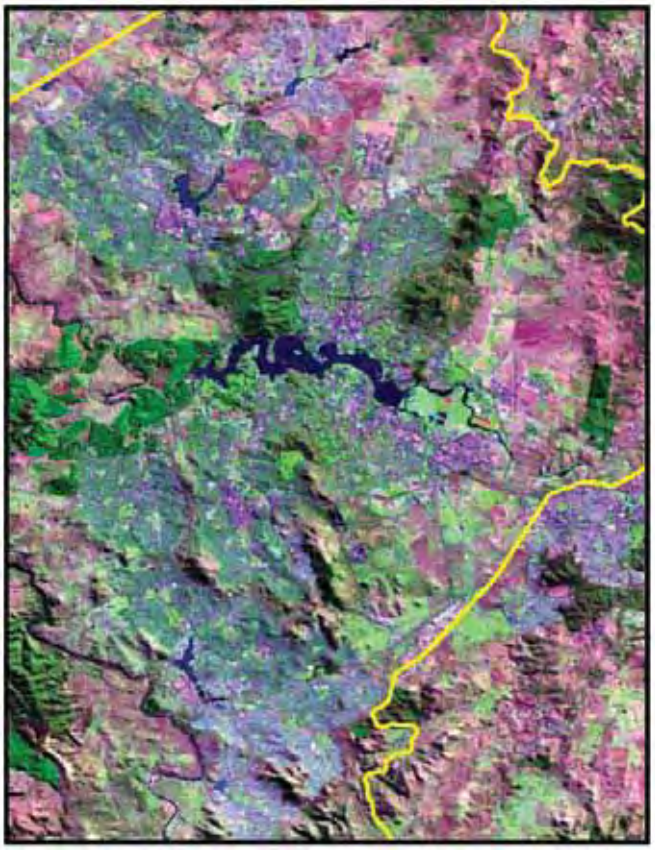


Figure 2**:** Study area, Canberra, Australian Capital Territory (ACT) Australia.
(Data courtesy Geoscience Australia © 2001)

These provide generalisation functions through a variety of algorithms and operations on raster data. Various functions, such as the maximum likelihood classification, filtering, aggregate, merge algorithm, with a variety of parameters, have been tested. Image segmentation applies both spectral information (feature vector of the pixels) and spatial information (size, shape, texture, contextual information and adjacency to other pixels) to detect appropriate segments / classes within a given image. In this study Landsat-7 imagery acquired over Canberra (Figure 2) was pre-processed and segmented into 19 landscape themes, each containing several sub-classes, forming a total of 10 classes in the master image database. The road class was firstly derived from the pixel-based classification because, at a resolution of 30m, and the geometric linear characteristic of the object, the road consists of many pixels and needs to be verified with existing ancillary GIS data (e.g. 1:250,000 national topographic data, Landsat-7 imagery). Similar themes based on the landuse category were collapsed and, thereby, generalised. Problematic features (e.g. residential buildings and industrial buildings, roads and building roofs) which exhibit similar spectral characteristics were then reclassified. This was done through on-screen digitising of the problematic features based on the GIS analyst's knowledge and the ancillary information (e.g. existing topographic maps, street directory road maps). These features were then subtracted from the master classified database followed by reclassification of each of the subset datasets. Also, existing roads were buffered and rasterised. In a later step, the reclassified subset databases were mosaiced with the master database to form the final segmentation thematic map. Finally, thematic generalisation was applied.

2.1 Data Pre-Processing

Because data could come from disparate sources and often the user has limited knowledge of the source, it is essential to perform some validation and testing of data prior to starting generalisation. Topology validation, geometric feature validation and feature attribute quality checks were performed on the 1:250,000 national topographic roads data which was used in this study.

2.2 Generalisation Methods

Three techniques were used to control the properties of the generalised data which should be distinguished:

2.2.1 Supervised classification: was used to group objects, using an object-oriented segmentation method, into several landscape themes, each containing several sub-classes, forming a total of 19 classes. Similar classes were combined to form a total of 10 land cover categories.

2.2.2 Thematic generalization: was used to merge the derived classes, on the basis of similarities in their landscape characteristics. It is difficult to automate this process due to its complex, diverse and non-deterministic nature, particularly when attempting to effect a subjective and intuitive procedure (i.e. when, where and how to generalise).

2.2.3 Spatial aggregation: was applied to raster data to aggregate cells on the basis of their spatial neighbourhood. Experience shows that this method is particularly applicable to the treatment of land cover (Moody, 1998 and Petit and Lambin, 2001). The buffered road data was rasterised at the closest resolution (30m) to the ETM+ resolution, and was gradually aggregated with a majority filter. In this way, generalisation of cluttered features becomes an easy task.

2.3 Supervised Classification

The objective of multispectral image segmentation is to segment the image into spectrally homogeneous regions. Multispectral image classification is categorised into two main approaches: whether supervision from an operator is required or not. The GIS or image analyst closely controls a supervised classification process by selecting pixels that represent known landscapes. Obviously, the analyst uses GIS data and ancillary information (e.g. ground truth data, existing landuse data, SPOT-4 data, etc) to facilitate the classification. In this process the analyst trains the machine to recognise the similar patterns (pixels) or homogeneous regions that represent each class. Defining a set of desired classes is an integral part of the process and it is possible to define as many classes as required. Then an appropriate strategy is selected for assessing the information in the available data, and to make decisions for labelling classes. The spectral signatures of landscape types might be confirmed by reference to ground data (e.g. Foody, 2002 and Wang et al., 2004) or by the analyst defining the classes by interactive extraction of training areas (Leica, 2004). These spectral signatures are then used to classify all pixels comprising the image. The supervised technique is well documented in the literature and is the most popular multispectral classification method (Richards, 1986, Johnson, 1994 and Petit and Lambin, 2001). Among the most popular supervised classifiers are minimum distance, parallelepiped, and maximum likelihood. A maximum likelihood algorithm estimates the likelihood of a particular pattern belonging to a

category. One of the advantages of the maximum likelihood classifier (MLC) is that it is easy to use and theoretically guarantees minimisation of the classification error. This is the most widely employed classification algorithm for digital classification of imagery (Bolstad and Lillesand, 1991, Trietz et al., 1992, Johnson, 1994 and Forghani et al., 2007). The MLC technique was used in this study for image classification as it takes into account spectral descriptions for classes modelled using multivariate Gaussian densities (Yang, 2007). In this work the training areas were selected with the use of area-of-interest tools and stored in a Signature Editor file. The result of this supervised classification of the ETM+ imagery is presented in Figure 3. In built-up areas, problematic features such as roofs, streets, and pavements, exhibit similar spectral reflectance and this increases misclassification errors. To overcome this problem and to improve the classification success rates, several classes were initially selected and assessed by available GIS datasets. Maximising the number of training areas helps to minimise confusion between similar features and decreases the misclassification rate.

2.4 Image Reclassification

The main concern in this research is to map out the landuse classes. The buffered road map in raster format was used as control data. The segmented data was then compared to the existing roads map and landuse data to assess the classification accuracy. Later, spectral classes with similar cover types were merged to form 19 classes. The resulting modified signature file was applied to perform the MLC using a standard deviation of 2. Then finally, a map was produced showing 10 classes (see Table 1).

2.5 Thematic and Spatial Generalisation

Raster-based generalisation of Landsat ETM+ imagery land cover data using statistical filtering available in ERDAS IMAGINE and the aggregation operation tool of ArcGIS is shown in Figure 3. These tools provide raster map algebra methods that resample raster map layers using various aggregation techniques such as averaging and interpolation of raster map layers. The input raster layer was generalised so that each raster cell covers an area of 30 x 30 meters on the Landsat imagery. Aggregate is a generalization function applied for raster data. A similar function called 'dissolve' that is used for polygon themes stored as vector data. The 'dissolve' function removes borders between polygons having the same value for a given attribute.

Table 1: Descriptions of landscape classes from ETM+ data supported by a GIS database

Landscape categories	Categories in performing aggregation	Generalisation
Water - lake - river	1, 2	Water
Open land - bare soils - asphalts - concretes	3, 4, 5	Bare land
Agriculture - crops - orchards	6, 7	Agriculture
Grass - Grassland - Grassland and shrubland	8, 9	Grasslands
Shrubs	10	Shrublands
Forest - deciduous - perennial	11,12	Forests
Built-up areas - commercial - industrial	13,14	Commercial and industrial areas
Built-up residential areas (houses, townhouses and apartments)	15	Residential areas
Roads - highways - main roads - streets - lanes	16, 17, 18	Roads
Unknown	19	Other

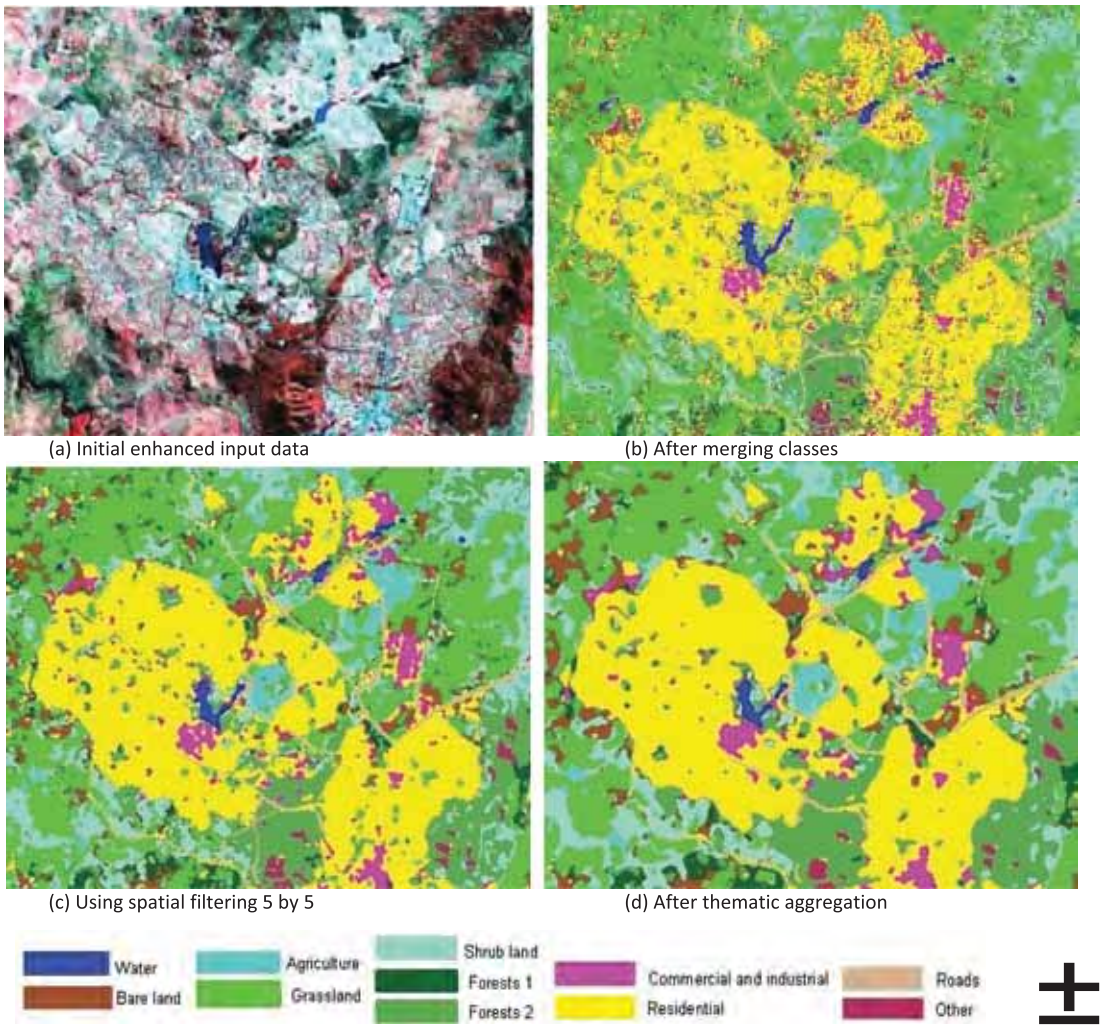


Figure 3**:** Comparison of classification and generalisation outputs.
(Data courtesy Geoscience Australia © 2001)

3. Discussion

It is important to gauge the accuracy of the segmentation method. To evaluate output of the proposed raster classification and generalisation framework, the accuracy of the final segmentation and generalised map was visually evaluated by superimposing the road buffer layer and other landuse data over the classified image. Also, the segmented image was numerically compared to the existing landuse map using coincidence matrices and accuracy measures. Mapping man-made features in a heterogeneous environment such as a built-up area is problematic since the roads and other man-made structures (e.g. buildings) exhibit similar spectral characteristics. Bare soils, dirt roads, tracks, and roofs have very similar spectral signatures. The confusion of these features with roads is clearly demonstrated. Overall classification accuracy was 85.5 percent. In Table 2, the accuracy measures (Kappa coefficient, overall classification accuracy, producer and user's accuracy, commission and omission errors) corresponding to the various classes are presented along with the error matrices for supervised classification of Landsat ETM+ data. Gurney (1981) indicated that when an automated classifier is used over an urban area, errors increase considerably. The results demonstrate that approximately 85.5 percent of all objects were correctly classified. The results were validated by calculation of the pixel-by-pixel accuracy. The certainty and uncertainty of the classified pixels were validated for the study area. The accuracy for classification and generalisation of man-made structures increased over rural areas, as the image exhibited a greater contrast between the roads and other elements. The estimated overall accuracy based on the reference map agreed with the overall

visual interpretation of the output. To demonstrate the quantitative assessment of image segmentation, the classification accuracy, omission errors and commission errors for each class were computed and presented in a confusion matrix (Table 3). This approach to image segmentation accuracy evaluation is widely documented in the literature (e.g. Harris and Ventura, 1995 and Wang et al., 2004). Ton et al., (1991) described the omission errors as the omitted pixels in the classification process divided by the total number of pixels in the land cover type, whereas the commission error represent pixels labelled as the land cover type by the algorithm but not by the ground truth data. The classification accuracy of a land cover type is defined as the number of correctly classified pixels divided by the total number of pixels in the land cover type. It is clear that classification accuracy plus the omission error should sum to 100 percent. The commission error is considered as a separate statistic.The accuracy assessment presents the overall accuracy, Kappa, commission (% of extra pixels in class) and omission (% of pixels left out of class) errors, producer accuracy and user accuracy for classified image. The confusion matrix results demonstrate how each of the accuracy assessment is derived. The maximum likelihood classifier using training sites and associated ground truth data produced an overall accuracy of 85.5% and a Kappa coefficient of 0.798. Cohen's Kappa coefficient statistical measure (κ) is considered to be a robust measure for the uncertainty associated with spatial information (Hope & Hunter, 2007). While there are no absolute cut-offs for the Kappa coefficient, 0.7 or higher is widely accepted as a satisfactory value. In this study, the Kappa value for the classified image was 0.798 indicating that an observed classification is in agreement to the order of 79.8 percent. This value indicates a substantial agreement based on the Kappa categories by Landis and Koch (1977). The overall accuracy is defined as a percentage of the test-pixels successfully assigned to the correct classes. The overall classification accuracy decreased in areas where the roofs of buildings, bare soils, concrete, and tracks roads have very similar spectral characteristics. This problem can be related to increasing noise due to the heterogeneous nature of the spectral response of urban areas (Gerke and Heipke, 2008) and cleared agricultural lands. Similar classification accuracy has been reported using maximum likelihood classifier with identification of six broad cover types over a rural/urban area from SPOT and TM data (Welch, 1985, Moller-Jensen, 1990 and Barr, 1992).

Table 2: Image segmentation accuracy summary statistics

Accuracy measures						**Classified Ccategories**
Kappa	Commission %	Omission %	Overall accuracy %	Producer's accuracy %	User's accuracy %	
0.798	4	1	99	96	100	Water
	18	12	88	84	82	Bare land
	17	9	91	80	87	Agriculture
	21	7	93	83	79	Grasslands
	32	13	87	87	85	Shrub lands
	24	5	95	95	87	Forests (1 & 2)
	60	25	75	79	78	Commercial and industrial areas
	63	19	81	82	75	Residential areas
	74	33	67	79	74	Roads
	65	20	79	83	86	Other
	37.8	14.4	85.5	84.8	83.3	Total

Table 3: The confusion matrix results generated from classified data and reference ground data

Classified Categories	**Actual Categories**										
	Wa	Ba	Ag	Gr	Sh	Fo	Co	Re	Ro	Ot	**Total**
Water	95	0	1	2	0	2	0	0	0	0	100
Bare land	0	86	2	0	0	0	3	4	0	5	100
Agriculture	0	4	85	1	3	5	0	0	0	1	99
Grasslands	0	0	6	76	5	3	0	0	0	2	92
Shrublands	0	0	0	1	90	1	0	0	0	3	95
Forests (1 & 2)	0	0	3	2	4	85	0	0	0	1	95
Commercial	0	0	0	0	0	0	74	13	8	4	99
Residential	0	0	0	0	0	0	8	78	4	2	92
Roads	0	0	0	0	0	0	0	3	84	2	89
Other	0	5	3	1	0	0	3	10	4	69	95
Total	95	95	100	83	102	96	88	108	100	89	
Note: Wa = Water, Ba = Bare lands, Ag = Agriculture, Gr = Grasslands, Sh = Shrublands, Fo = Forests (1 & 2), Co = Commercial / industrial, Re = Residential, Ro = Roads, Ot = Other											

Accuracies in the range of 55-81% have been reported. To enhance the supervised classification results, post-classification was carried out. It was found that this improved the overall classification accuracy slightly, but commission errors also increased (by 6%).

4. Conclusions

The aim of this study was to develop a new methodological framework for segmentation and generalisation of raster data. The Interactive Automated Segmentation and Raster Generalisation Framework (IASRGF) can be used to map out features from satellite imagery using an object segmentation and database generalisation approach. Three methods – supervised classification, thematic generalisation and spatial aggregation – were used to test the raster generalisation framework by applying the methodology to the integration of GIS data and remotely sensed data to segment landuse classes. The study demonstrated the usefulness of an image segmentation technique for deriving landuse categories. This framework can serve as a practical methodology to segment Landsat-7 or similar datasets over urban and rural areas where traditional image classification methods fail to deliver satisfactory results. It is suggested that the methodology can be tested further for multiple scale generalisation through the integration of different satellite images from fine resolution to coarse resolution.

Acknowledgements

This study was sponsored by the Faculty of Engineering's PhD Scholarship and Supplementary Engineering Award (SEA) at the University of New South Wales. The authors would like to thank Geoscience Australia for provision of the GIS data (1:250,000 national topographic data) and the remote sensing data (Landsat-7 and SPOT-4 imagery). The authors also would to sincerely thank the anonymous reviewers and Dr Alan Forghani of Geoscience Australia for their insightful comments and constructive suggestions of the original manuscript. Dr Nitin Kumar Tripathi of Asian Institute of Technology & Editor-in-Chief, International Journal of Geoinformatics has kindly facilitated print of the paper into this issue of the Journal. This paper is extracted and modified from a paper that was presented at the first International Symposium on Cloud-prone & Rainy Areas Remote Sensing (CARRS), Hong Kong, 4-8 October 2005.

References

AGENT, C., 1999, Selection of Basic Algorithms. Technical Annex, AGENT Consortium, 1-31.

Barr, S. L., 1992, Object-Based Reclassification of High Resolution Digital Imagery for Urban Land-Use Monitoring. *Proceedings of the XVIIth Congress of International Archives of Photogrammetry & Remote Sensing,* Washington D.C., 2-14 August, Commission VII, Vol. XXIX, 969-976.

Bolstad, P. V., and Lillesand, M., 1991, Rapid Maximum Likelihood Classification. *Photogrammetric Engineering & Remote Sensing,* 57(1), 67-74.

Cihlar, J., Xiao, Q., Chen, J., Beaubien, J., Fung, K., and Latifovic, R. 1998, Classification by Progressive Generalisation: A New Automated Methodology for Remote Sensing Multichannel Data. *International Journal of Remote Sensing*, 19(14), 3685-2704.

Daley, N., Goodenough, D. G., Bhogal, A. S., Bradley, Q., and Yin, Z. 1997, Comparing Raster and Object Generalisation. *Proceedings of the IGARSS 1997* Singapore*,* 3-8 August, 677-679.

ESRI, 1992, Arc/Info Documentation. ESRI Redlands California, USA.

Foody, G. M., 2002, Status of Land Covers Classification Accuracy Assessment. *Remote Sensing & Environment*, 80, 185-201.

Forghani, A., Osborn, J., and Roach, M., 1997, An Image Interpretation Model to Support Integration of Image Understanding Techniques within a GIS: Delineation of Road Structures from Aerial Imagery. *Proceedings of the International GIS/GPS'97 Symposium,* Istanbul, Turkey, 15-19 September, 10 – 15.

Forghani, A., Cechet, B., and Nadimpalli, K., 2007, Object-Based Classification of Multi-Sensor Optical Imagery to Generate Terrain Surface Roughness Information for Input to Wind Risk Exposure Simulation. *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS),* Barcelona, Spain, 23-27 July, 1-6.

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

Gurney, C. M., 1981, The use of Contextual Information to Improve Land Cover Classification of Digital Remotely Sensed Data. *International Journal of Remote Sensing,* 2(4), 379-388.

He, H. S., Ventura, S. J., and Mladenoff, D. J., 2002, Effects of Spatial Aggregation Approaches on Classified Satellite Imagery. *International Journal of Geographical Information Science*, 16(1), 93-109.

Hope, S., and Hunter, G. J., 2007, Testing the effects of positional uncertainty on spatial decision-making. *International Journal of Geographical Information Science,* 21(6), 645–665.

Jaakkola, O., 1998. Multi-Scale Categorical Databases with Automatic Generalisation Transformations Based on Map Algebra. *Cartography & Geographic Information Systems* 25(4), 195-207.

Johnson, P. O., 1994, Development of the Sample Survey as a Scientific Methodology. *Journal of Experiential Education,* 27, 167-176.

Kazemi, S., and Lim, S., 2007, Deriving Multi-Scale GEODATA from TOPO-250K Road Network Data. *Journal of Spatial Science*, 52(1), 171-182.

Kazemi, S., Lim, S., and Ge, L., 2005a, Integration of Cartographic Knowledge with Generalisation Algorithms. *Proceedings of the 2005 IEEE Geoscience & Remote Sensing Symposium,* Seoul, South Korea, 25-29 July, 3502–3505.

Kazemi, S., Lim, S., and Ge, L., 2005b, An Interactive Segmentation and Generalization Framework for Mapping Features from Landsat 7 Imagery. *Proceeding of the First International Symposium on Cloud-prone & Rainy Areas Remote Sensing (1st CARRS)*, Hong Kong, 4-8 October, CD-ROM procs.

Kazemi, S., and Lim, S., 2005c, Generalisation of Road Network using Intergraph DynaGen System. *Proceedings of the SSC 2005 Spatial Intelligence, Innovation and Praxis: The National Biennial Conference of the Spatial Sciences Institute, 12-16 September, Melbourne, Australia,* 1285-1296.

Landis, J. R., and Koch, G. G., 1977, The Measurement of Observer Agreement for Categorical Data. *Biometrics,* 33, 159-174.

Leica Geosystems, 2004. ERDAS Field Guide, USA.

Moller-Jensen, L., 1990, Knowledge-Based Classification of an Urban Area using Texture and Context Information in Landsat-TM Imagery. *Photogrammetric Engineering & Remote Sensing,* 56(6), 899-904.

Moody, A., and Woodcock, C. E., 1995, The Influence of Scale and The Spatial Characteristics of Landscapes on Land-Cover Mapping using Remote Sensing. *Landscape Ecology,* 6, 363–379.

Moody, A., 1998, Using Landscape Spatial Relationships to Improve Estimates of Land Cover Area from Coarse Resolution Remote Sensing. *Remote Sensing of Environment,* 64, 202-220.

Petit, C. C., and Lambin, E. F., 2001, Integration of Multi-Source Remote Sensing Data for Land Cover Change Detection. *International Journal Geographical Information Science,* 15(8), 785-803.

Richards, J. A., 1986, Remote Sensing Digital Image Analysis: An Introduction. Springer-Verlag, Berlin.

Ton, J., Sticken, J., and Jain, A., 1991, Knowledge-Based Segmentation of Landsat images. *IEEE Transactions on Geoscience & Remote Sensing*, 29(2), 222-232.

Wang, L., Wayne, P. S., Gong, P., and Gregory, S. B., 2004, Comparison of Ikonos and Quickbird Images for Mapping Mangrove Species on the Caribbean Coast of Panama. *Remote Sensing of Environment,* 91, 432-440.

Walter, V., 2004, Object-Based Classification of Remote Sensing Data for Change Detection. *ISPRS Journal of Photogrammetry & Remote Sensing,* 58, 225-238.

Welch, R., 1985, Cartographic Potential of SPOT Image Data. *Photogrammetric Engineering & Remote Sensing, 51(8),* 1085-1091.

Wenxiu, A., Jianya, G., and Zhilin, L., 2004, Knowledge-Based Generalisation on Land-Use Data. *Proceedings of the 5th Generalisation Workshop,* Paris, France, 28-30 April, 1-5.

Yang, X., 2007, Integrated use of Remote Sensing and Geographic Information Systems in Riparian Vegetation Delineation and Mapping. *International Journal of Remote Sensing,* 28, 353-370.