Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

Thesis Submitted for Admission to Degree of

Doctor of Philosophy

1 9 9 9

Jason Dykes

Department of Geography
University of Leicester
UK
"Cartography has undergone a profound change over the past decade," MacEachren (1997).

One of the primary reasons for this technology-induced change is that highly dynamic maps are being used increasingly across the sciences as tools through which initial exploratory analyses of geographic information can be made in a process termed 'visualization'. Maps are no longer used solely to record information and communicate summaries to large audiences. The dynamic and highly interactive software descendants of the paper maps of the previous decade form an active part in the process of knowledge acquisition and are changing the way in which science is undertaken. They are used by highly skilled individuals to determine patterns and elicit trends from huge, complex and growing databases of spatial information.

The work presented here describes research efforts undertaken to develop generally applicable new methods through which the techniques derived for the cartography of the past can be effectively applied to the map use of the present and future. It makes the case for an approach to visualization that synthesises techniques developed in a number of fields. It utilises cartographic symbolism in a two-dimensional software environment through which maps can be produced that contain high levels of interaction and flexibility. The argument is supported at three levels:

- An approach is presented that makes imaginative and innovative use of a scripting language for graphical user interface production, in order to provide an environment for visualization.
- Working software that implements the approach in order to address issues concerned with the analysis of enumerated spatial information is then provided and detailed.
- This is then applied to a number of scenarios in which visualization is undertaken, and to address evolving research issues of concern to the cartographic and statistical communities.

Finally assessments are made of the usage and utility of the methods presented and some extensions are both suggested and offered. The changes to cartography that occur in the next decade are likely to be just as profound as those experienced in the past decade. The realm of cartography continues to expand, as do the types of map that are achievable, the applications to which maps are put and the numbers of map users. An assessment is made of likely trends and ways in which the approach presented can provide a useful contribution to the future of cartographic visualization. Interactive software, dynamic figures and computer scripts provided in separate appendices form a major element of the work, in terms of the illustration, demonstration and specification of the methods used.
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

I. Preface

Jason Dykes

Department of Geography
University of Leicester
UK
I. Preface: History, Continuity, Cartography, Content and Context

1. History

In September of 1991 I sat on an ergonomically disastrous lab stool at a science laboratory bench, switched on a computer terminal, logged on to a VAX VMS mainframe and tried to decide how I would approach research into appropriate and useful depiction of an enumerated data set with which I was extremely familiar. I had conducted an investigation into the effectiveness of manipulating the parameters associated with traditional static cartographic products (Dykes, 1991) and in response to an anticipation of advances in computing was keen to develop novel mapping techniques from which researchers could glean extra information in parallel with the broader trend in science as a whole towards 'visualization' (McCormick et al., 1987). The first issue was to find a way of mapping data. The VAX VMS service that I was using was out of date and closing down, and the FORTRAN UNIRAS graphics libraries that formed the basis of my maps were no longer available.

The 'surfing' metaphor is a popular, but somewhat tenuous one in my opinion, for describing the process of tracking down information on the Internet. In 1991 there was no Internet. However, I still felt that I was metaphorically floating at the edge of a sea of extremely exciting technological innovation anticipating waves of approaching opportunity and wondered how to catch one, rather than have these waves wash by and away into the distance. The technological waves through the field of computer cartography over the last eight or ten years have been notable and of increasing frequency and magnitude. Researchers operating within this stormy sea assess the waves of technology as they approach, and either catch them and contribute appropriate scholarly effort or allow them to pass by as the research process continues. Considerable re-skilling and re-consideration have been required over the period to keep one's work relevant and at the forefront of academic endeavour in order to stay afloat. As a computer cartographer with no computer or ability to create cartography, I caught the first wave that I could, which involved transferring my data to a more modern and powerful Sun Sparc Station and gaining competence in the world of UNIX. I began by exploring the possibilities that the operating system and its software and systems programming languages offered computer cartography.

Since 1991 I have been involved in catching a number of waves of technology, some more substantial than others. My work has been based upon, and taken appropriate advantage of, at least six different combinations of machine, operating system and software:

- the original VAX VMS using FORTRAN 77 with UNIRAS graphics
- the Sun Sparc Station with the 'C' systems programming language and Unix shell scripts
• a Silicon Graphics Irix Mainframe where I used an open source GIS and a rudimentary graphical user interface builder
• a Silicon Graphics Indy workstation where I assessed the utility of the graphical programming paradigm with Visualization in Scientific Computing (ViSC, McCormick et al., 1987) software and began to discover that the scripting languages that were available could be more powerful for cartography than the systems programming languages of the time
• a Windows 95 PC that allowed me to concentrate on scripting
• a Windows NT Laptop that provides an enormous amount of power and flexibility

The amount of change that these progressive technologies have brought about is further emphasised by considering the magnitudes of available disk space and data and the quality of the tools available for development over this period. In 1991 the disk space available to me was extended from 5MB on the University mainframe to 20MB at the request of my supervisor. As I sit at home typing at my PC, the external disks strewn across the desk have a combined storage capacity of 8GB. Today telephone companies collect “gigabytes of spatially referenced data every hour” (Wills, 1999) and services such as the TerraServer (Microsoft Inc., 1998) and CASWEB (Harris, 1999) provide fabulously easy access to enormous geographic data sets. When the UNIRAS libraries became unavailable I began experimenting with the xGen graphical user interface (GUI) builder, a language of frustratingly limited functionality and adaptability when compared to today’s array of highly flexible scripting and prototyping tools with interface and communications possibilities.

Several ‘moments of significance’ have resulted in a frantic front-crawl, to continue with my surfing metaphor, as the magnitude of specific technological breakthroughs has been instantly apparent. These emphasise the changes that have occurred over the period of my research still further. For example, I remember clearly my first experience of a WWW browser in 1993, and being both astonished that I could view satellite data in near real-time and aware of the enormous potential consequences of this form of communication. In 1995, as a UNIX devotee, I remember being sceptical when told by a senior figure in the UNIRAS hierarchy that the power of the PC market dictated that that would be where developments in visualization occurred, despite the massive investment in UNIX machines in academia and the fact that the very term had been defined by UNIX hardware in the McCormick Report. I began to take PCs seriously as research machines. And the week spent at SIGGRAPH in Los Angeles in the summer of 1997 demonstrated just how incredibly advanced the field of computer graphics and computing interfaces had become as I was able to experience the imaginative use of state of the art graphics in non-data applications. I took part in a virtual karate bout by ‘fighting’ a colleague at a safe distance as the movement of our limbs was tracked and projected onto computer animated figures in front of us, I played virtual
basketball with my hands linked to haptic sensors that gave the impression of holding and releasing a ball, and I marvelled at the realistic images produced by the movie industry of landscapes and people and Jurassic dinosaurs... generated using PCs!

2. Continuity

Over this period the kinds of cartography and investigation that have been possible have varied enormously, as have the kinds of research that have been going on in parallel to my own. The effort required to re-skill as each of the identified waves has approached has been considerable (think about it as the short sharp strokes that a surfer must take to get up to speed with an incoming breaker). In terms of published research, Dorling et al. (1998) identify a huge increase in the number of publications on 'visualization' in the broad field of 'Social Science' between 1991 and 1998 (see figure I.1). When Internet 'papers' are included in the figures the growth of the body of research literature has been exponential. A conservative extrapolation of the numbers that Dorling et al. (1998) present would suggest that between two and four hundred additional contributions are likely to have been published in the year since their survey. These figures focus on the social sciences. Research on visualization and exploratory techniques is occurring in a whole variety of broader fields in the natural and environmental sciences and statistics.

Figure I.1 Visualization in the Social Sciences: A Bibliographic History, from Dorling et al. (1998).

The graph shows the estimated number of 'visualization-relevant' conference papers, journal articles, books and chapters published in each year of the 20th century based upon the bibliography provided by Dorling et al (1998). A cumulative total is also shown. The series excludes articles published on the Internet, for which Dorling et al. (1998) estimate an exponential trend from zero the early 1990s to upwards of 250 papers per annum by 1998.
In this ever changing environment theories become outdated before they can be published, let alone tested and verified. Researchers are faced with new techniques and potential applications on a monthly, if not weekly basis. There are few constants in such a changing research environment. Two of these are software that demonstrates ideas and the current cutting edge and a stream of applications and solutions that respond to and take advantage of the new possibilities. Software is a particularly appropriate response in the medium in which we find ourselves as it can be replicated, used and assessed by other researchers across the world using the Internet.

In combination, flexible software that provides an environment where new ideas and possibilities can be incorporated quickly is especially significant. That is what I have attempted to produce throughout the period. Pioneering software that draws upon my knowledge of spatial information, geographic enquiry and the cutting edge of technology. Software that takes the best features of ideas in a number of fields and applies them to our new geographic problems and uses our new media to target specific solutions. Software that demonstrates, reveals, engages and excites. Communication has always been an important goal of the cartographer and never more so than in the age of communication when the current lasts for less and less time. MacDougal (1992) presents his ideas with demonstration software in 'operational prototype' form. His definition of an operational prototype is "a program that operates only when I use it". The software that has been developed throughout this research has been produced with wider application and a broader audience in mind in order to communicate ideas and substantiate their application.

3. Cartography

The changes that I have documented have had a colossal effect on cartography summarised by Alan MacEachren in his statement that "cartography has undergone a profound change over the past decade" (MacEachren, 1998). The cartographic literature developed from the 1950s right through to 1980s concerns itself with static maps and average users and explores issues that are associated with the paper medium. These are no longer problems for a huge number of map-makers and users, and often simply don't matter in the modern mapping era. Despite their relatively recent publication, the relevance of many of the papers produced before 1990 belongs to a time long since gone. The tools and equipment that were and are available bear no resemblance, and slowly map users are realising that they can equip themselves more successfully for certain tasks with extremely novel products, just as non map-users are becoming enlightened by the utility of thinking graphically and organising information spatially. The development of this thesis has involved an evolution of ideas from the cartographic Jurassic of the pre-1990s to produce techniques that embody the computer graphic power of the 1990s, demonstrated by the computer generated dinosaurs of Jurassic Park, and apply them to real world situations.
My work in this area began with relatively limited and small-scale experiments with traditional maps and ideas about possible cartographic futures, presented at academic meetings and reproduced in refereed journals (Fisher et al., 1993; Dykes, 1994). More interactive graphics and innovative forms of map use were incorporated as techniques became available (Dykes, 1996) and work has been presented to the academy through the normal channels as well as via the Internet (Dykes, 1999), throughout.

As my research has progressed I have presented ideas and demonstrated example software to academics in: Los Angeles and Santa Barbara, California; Gavle, Sweden; Delft, Amsterdam and Wageningen, the Netherlands; Chicago, Illinois; Barcelona, Catalonia; Madison, New Jersey; Sydney, Australia; Minneapolis, Minnesota; Warszawa, Poland; Paderborn, Bonn and Augsburg, Germany; as well as Manchester, Edinburgh, Greenwich, Loughborough, Leeds, Keele, Newcastle, Southampton, Coventry, Plymouth, Nottingham and the Royal Statistical Society, London in the UK.

The increase in computer power that has taken place over the last eight years has made the initial experiments using simple maps with small numbers of symbols all the more important as the techniques that were applicable then to small data sets are now possible for relatively large, and far more revealing spatial data sets. For example, the UK census of population contains over 14,000 symbols when mapped at ward level. Techniques demonstrated in 1996 using 10 UK regions (Dykes, 1996) can now be applied to the 14,000 electoral wards within them, in preparation for the 2001 Census (Dykes et al., in press). The relevance of scaling these mapping technologies can be assessed and new interactive techniques developed to suit the medium, data set and data magnitudes.

4. Content

I have been working in this area of rapidly changing technology with a variety of tools, trying to acquire and combine influences and ideas from a number of developing fields. Retaining any continuity at all over this period has been a considerable feat, and doing so whilst providing usable software and flexible code that addresses current and important research issues is the very real and considerable achievement that I have tried to accomplish.

The purpose of this volume is to collate the work presented in the publications and presentations that I have produced in order to demonstrate that I have achieved this aim and that my research fulfills the requirements for the degree of Doctor of Philosophy in the Faculty of Science at the University of Leicester. It is effectively the story of the research that I have undertaken in the last eight years. The submission replicates much of the contribution that already resides in the public domain, but places these within a broader structure and with continuous reference to the thesis in...
hand. It fulfils the terms and conditions outlined in the 'Postgraduate Regulations' (University of Leicester, 1998) in the following ways:

- It reports on advanced study and research
- It contains original work
- It contains material which has proved to be worthy of publication
- It contains appropriate appendices & supplements including published work in the general field of the study

The submission consists of a main body of text and figures constituting a collation of published papers, documented computer code and an appendix of refereed journal papers. In collating the papers every attempt has been made to ensure that the text flows as a piece of work in its own right. Producing a suitable trade-off between demonstrating the breadth and amount of the work that has been published and creating a solid, consistent and continuous argument in this volume has proved to be a difficult task. The reader is asked to recognise this quest for compromise when considering the thesis.

5. Context

All of this work has taken place in a human context and the last eight years have been a time of enormous personal change and significance. I have graduated twice, changed Ph.D. supervisor been employed in three full-time research positions, learned to drive and bought my first car (and second, and third), spent six months nursing my ailing father who died of his cancer, taken on a mortgage and renovated a house and garden at great effort, started teaching and developed appropriate courses, taken part in five field trips, ridden the highs and lows brought about by birth and death and serious illness in the family, turned from a bright and sharp young thing in an attacking position on the football field to a semi-crooked veteran at the back (the terms 'playmaker' or 'libero' are often used to make the demotion seem more attractive) and found an immensely loving and supportive partner who has sustained me through all of these events and encouraged me enormously in my endeavours.

This volume tells the story of the research that I have undertaken over the period, and presents the fruits of my labours. The surfing has been hard but fulfilling work.
Interactive Maps for Exploratory Spatial Data Analysis: Cartographic Visualization Approach, Implementation and Application

II. Acknowledgments

Jason Dykes
Department of Geography
University of Leicester
UK
II. Acknowledgments

People:

Primarily I owe great thanks for ideas, encouragement, contacts, patience, support and more patience, to David Unwin, Peter Fisher and Jo Wood. I've been extremely lucky to be associated with them and wouldn't have been able to achieve the work presented here without their considerable assistance. I'm grateful and honoured to have been part of 'the team' for so long.

Many other people have been extremely helpful over the years. I've probably forgotten most of them. Some that spring to mind are: Mitch Langford, Alan Strachan, Lucy Bastin, Bill Hickin, Kate Moore, Chris Taylor, Dominic Kniveton, Clare Madge, Martin Phillips, David Orme, Charlie Amot @ Leicester University; David Unwin, Kath Stynes & Nathan Williams @ Birkbeck College; Alan Jenkins @ Oxford Brookes University; Jonathan Raper & David Mountain @ City University; Wim van der Knaap & Birgit Elands @ the University of Wageningen; Antony Unwin, Adi Wilhelm, Martin Theus, Robert Steck, Heike Hofmann Meike Sander @ the University of Augsburg; Graham Wills & Steve Eick @ Lucent Technologies; Andreas Buja @ AT&T Labs; Jackie Carter & Keith Cole @ Manchester Computing; Alan MacEachren & David DiBlase @ Penn State; Menno-Jan Kraak @ ITC; Kathryn Strickland @ Dartmoor National Park; Jason Pascoe & Nick Ryan @ the University of Kent; Joost Buurman & Andrea LoCashio @ the Vreij University, Amsterdam; Juergen Symanzik; Di Cook; John Haslett; Martin Dunsby; Al Inselberg; Roger Bivand; Dave Martin; Phil Rees; Danny Dorling; Ian Bishop; Chris Brunsdon; Bill Cartwright; Andrew Skidmore; Jim Petch; David Fairbairn; Fraser Taylor; Mike Wood; Terry Slocum; John Ousterhout @ Sun/Scriptics; numerous students including Richard Hillman, Matt Duckham, Ben White, Sanjay Rana; and many many others.

The work on 'panoraMap' presented in chapter 6 has only been possible with the help, advice, ideas and enthusiasm of a large number of people including: The University of Leicester Department of Geography - especially Chris Taylor, Colin Williams, Dominic Kniveton, Jane Wellens, Mitch Langford; University of Leicester First Year Undergraduate Geographers; Jaakko Kähkönen and members of the International Cartographic Association Commission on Visualization; Dartmoor National Park; Andy Elmes, Carl Ishemo, Andrew Williams, Erik Meyles, Francien van Soest, Brian Chalkley, David Griffiths, Pauline Framingham, Adrian Holmes @ the University of Plymouth.

The people at home who are especially important and who have had to put up with it all for far too long include Emma, Phyl, Tobe, Bettie, Kath, Peter, Kath, Jacob, Rebecca, Mark, Deborah and Zac. I wish my Dad was still around to read this too.
Funding:

The 'cdv' software presented in chapter 3 was developed as part of the JISC New Technologies Initiative, Project Argus. I am indebted to colleagues working on the project including Kath Stynes, Jo Wood, Peter Fisher and David Unwin.

The 'panoraMap' software, presented in chapter 6 was developed as part of the Virtual Field Course (VFC) project, which is a joint venture between Birkbeck College (University of London), the University of Leicester and Oxford Brookes University and is funded by the UK Joint Information Systems Committee Technology Applications Programme (JTAP). Contributions to the content from the entire team consisting of Peter Fisher, Alan Jenkins, Jonathan Raper, David Unwin, Kate Moore, David Mountain and Jo Wood are gratefully acknowledged.

Most of the work reported in this thesis was conducted in association with one of these projects and all who have been involved with them deserve thanks their contribution and for enabling me to undertake the research presented here.

The local statistics section resulted from collaboration between Project Argus and colleagues at the Institute for Mathematics, University of Augsburg, Germany, which was funded by the British Council Anglo-German Research Fund.

Data:

"This work is based on data provided with the support of the ESRC and JISC and uses boundary material which is copyright of the Crown and the Post Office."

The Census data provided here are copyright of the ONS and are supplied with the support of the ESRC and JISC. The boundary material is provided by and the ED-Line Consortium which is copyright of the Crown and the Post Office.

Much of the raster map information is copyright of the Ordnance Survey. Thanks to Sally Payne for providing it efficiently.

The simple example data set using the British regions is based on data provided with the support of the ESRC and JISC and use boundary material which is copyright of the Crown and the Post Office. The attribute data are entirely fictitious have no authenticity or bearing on reality whatsoever.
The USA socio-economic data were acquired from the US Bureau of Census WWW Server. Kath Stynes put together the Midwest data set.

The aerial photographs of Dartmoor were provided by Kathryn Strickland & Geoff Haynes and remain copyright of the Dartmoor National Park.

David Mountain stitched many of the panoramic images, Kate Moore put together many of the raster backdrops and Jo Wood provided the fabulous and free ‘LandSerf’ software which does everything you could ever want, and if you can think of something it doesn’t do, then it will by the following morning. Jo also pointed me towards Td/Tk at the outset. I greatly and gratefully appreciate the insight and assistance that he has provided throughout my time at Leicester.

Wim van der Knaap and Birgit Elands at the Centre for Recreation and Tourism Studies at the University of Wageningen, The Netherlands, collected the time/space data set and made it available. Their collaboration and help with visualizing the data, hospitality and enthusiasm were greatly appreciated.

A significant amount of the imagery presented here was collected by students at the Universities of Leicester, Plymouth and London. Their co-operation, diligence and enthusiasm, even in the most testing field conditions, was greatly appreciated. The rest of the imagery was collected by myself. I thank all of the friends and family who have shuffled behind me in neat circles without a murmur of complaint as I have spun about on remote mountainsides clicking away with the digital camera on the various walks, bike rides and snowboard trips that we have undertaken over the years.
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

III. Contents

Jason Dykes
Department of Geography
University of Leicester
UK
III. Contents

I. Preface: History, Continuity, Cartography, Content and Context

II. Acknowledgments

III. Contents

IV. List of Figures and Tables

1. New Maps for Old: Dynamic Maps and Contemporary Map Use
   1.1 Mapping and Enumerated Information
      1.1.1 The mapping transform
      1.1.2 Enumerated information and cartography
   1.2 Recent Developments and Current Issues
      1.2.1 New visual variables
      1.2.2 New computational possibilities
      1.2.3 Dynamic maps
      1.2.4 Combinations and additions
   1.3 Dynamic Map Use
      1.3.1 Visualization map use and EDA with graphics
      1.3.2 ESDA techniques for visualization
      1.3.3 Towards a research agenda
      1.3.4 Some repercussions
   1.4 Mapping for Visualization
      1.4.1 Data-view links that support visualization map use
      1.4.2 Production and prototyping of novel views and symbolism
      1.4.3 Assessment of utility
   1.5 An Approach to the Visualization of Enumerated Data Sets
      1.5.1 Optimised design criteria
      1.5.2 Novel Views
      1.5.3 Interactive Mapping
   1.6 The Research Undertaken
      1.6.1 Research aims
      1.6.2 Outline
      1.6.3 Structure
2. Scripting: A Methodology for Dynamic Cartography 31
   2.1 The GUI Approach 31
   2.2 The Scripting Approach 32
   2.3 The Td/Tk GUI Builder 34
      2.3.1 Td fundamentals 35
      2.3.2 Tk fundamentals 37
   2.4 Td/Tk for Mapping 42
      2.4.1 Td/Tk for static maps 42
      2.4.2 Td/Tk for dynamic maps 46
   2.5 Visual Variables and Simple Cartography 50
   2.6 Testing the Approach 52

3. Dynamic Mapping for Visual Thinking: A Case Study 54
   3.1 Example Data 55
   3.2 Modes of Software Operation 55
   3.3 Data Views and Software Control 56
      3.3.1 Spatial geometry data 57
      3.3.2 Multivariate attribute data 60
      3.3.3 Attribute data keys 61
      3.3.4 Associating geometric and attribute data 62
      3.3.5 Selecting a data set for visualization 62
   3.4 Basic Visualization Functionality 62
      3.4.1 Dynamic comparison 64
      3.4.2 Brushing to highlight 65
      3.4.3 Dynamic re-expression 65
      3.4.4 Observer motion 66
      3.4.5 Brushing for linking 67
      3.4.6 Novel spatial views 68
   3.5 Advanced Visualization Functionality 72
      3.5.1 Combining variables 72
      3.5.2 More advanced brushing for subset selection 74
      3.5.3 Multivariate analysis 75
      3.5.4 A novel attribute view 78
      3.5.5 Scripting with 'cdv' 81
   3.6 Conclusion 84
4. **Extending the Approach: Dynamic Mapping for Exploring Representation**  
   4.1 Exploring Cartographic Representation  
      4.1.1 Spatial structure  
      4.1.2 Attribute composition  
      4.1.3 Cartographic abstraction  
   4.2 Exploring Statistical Representation  
      4.2.1 Statistical distribution within a locality  
      4.2.2 Visualizing spatial autocorrelation  
      4.2.3 Visualization of distance weighting coefficients  
      4.2.4 Investigating the variogram cloud  
   4.3 An Example Script

5. **Extending the Approach: Dynamic Mapping Examples and Applications**
   5.1 A Day in the Park at Schwalm-Nette  
      5.1.1 Tourist behaviour data  
      5.1.2 A dynamic mapping solution for visualization  
      5.1.3 Exploring representation  
   5.2 Combining Statistical Graphics and Geographical Views  
      5.2.1 Identifying and visualizing geographic neighbourhoods  
      5.2.2 Visualizing all locational information  
      5.2.3 Combining locational and statistical information – A dynamic ‘scatter map’  
   5.3 Local Multivariate Views  
      5.3.1 The need for local statistics  
      5.3.2 Locality selection – an interface  
      5.3.3 Visualization of multivariate local variation  
   5.4 Assessment of the Flexibility and Extensions

6. **Extending the Approach: Virtual Environments and Multimedia**
   6.1 An Application Specification  
      6.1.1 Software rationale  
      6.1.2 Method of implementation  
   6.2 Interactive Panoramas in Td/Tk  
   6.3 A Navigable Virtual Environment in Td/Tk  
   6.4 Incorporating Multimedia and Running External Processes  
   6.5 Visualization for Education  
   6.6 The Application in Use  
      6.6.1 Counter-urbanisation in the Teign Valley  
      6.6.2 Environmental management on Haytor Down
6.6.3 Catchment hydrology on Holne Moor

6.7 An Assessment of the Application

6.8 The Potential of the Approach

7. Discussion, Context and Consequences

7.1 Results, Outcomes and Significance

7.1.1 Use of the approach for dynamic mapping

7.1.2 Use of the implementations for visualization, demonstration and education

7.1.3 Development of the applications for additional visualization

7.2 Assessment of the Approach

7.2.1 An environment for software development

7.2.2 Limitations and drawbacks

7.2.3 Map complexity and map dynamism

7.2.4 Future support and development

7.3 Extending Flexibility and Functionality

7.3.1 Additional flexibility: Systems programming languages

7.3.2 Extending and formalising cartographic language

7.4 Conclusion

8. Conclusion and Futures

8.1 Synopsis of Thesis

8.2 Review of Assertions

8.3 Assessment of Recent and Future Trends

8.3.1 Combination of technologies

8.3.2 Advances in communications and sharing of data and software

8.4 Some Effects of The Internet

8.5 New Issues to Address

8.6 A Short-term Example

8.7 Conclusion
9. Appendices

Appendix A – Text
A1 Optimised Classification Stuff
A2 Notes on Data Assembly for ‘panoraMap’
A3 Selected Scripts from ‘cdv’ Software
A4 Scripts that Extend ‘cdv’ Functionality
A5 Virtual Reality and Visualization in the Fieldwork Context
A6 Scripts that Specify Focus and Orientation

Appendix B – Software on Compact Disk
B1 Tk Examples
B2 Interactive Figures: Map Examples 1-6 and Data Sets
B3 Visual Variables in Td/Tk
B4 ‘cdv’ Software
B5 Interactive Figures: Map Examples 7-9 and Data Sets
B6 ‘simplePanZoom’ Example Software
B7 ‘panoraMap’ Application Software
B8 Software to Visualize Td/Tk Speed Test Results
B9 ‘focus’ Example Software
B10 ‘orientation’ Example Software

Appendix C – Supporting Materials on Compact Disk
C1 Dynamic and High Resolution Figures
C2 Guide to using ‘cdv’ - Hypertext and Images
C3 ‘cdv’ scripts - Fully Formatted and Documented Source Code

Appendix D – Academic Papers
D1 The Cartographic Journal: (Fisher, Dykes & Wood, 1993)
D2 Computers and Geosciences: (Dykes, 1997)
D3 The Statistician: (Dykes, 1998)

10. Bibliography: Cited References
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

Jason Dykes
Department of Geography
University of Leicester
UK
IV. List of Figures and Tables

Preface: History, Continuity, Cartography, Content and Context

Figure 1.1 Visualization in the social sciences: A bibliographic history, from Dorling et al. (1998).

1. New Maps for Old: Dynamic Maps and Contemporary Map Use

Figure 1.1 A transformational view of cartography.
Figure 1.2 A transformational view of cartography - New cartographic possibilities.
Figure 1.3 A transformational view of cartography - Dynamic maps.
Figure 1.4 Map use in the process of scientific discovery (DiBiase, 1990).
Figure 1.5 Axes of the [Cartography]^3 (MacEachren, 1994).
Figure 1.6 Research aims – Cartographic visualization is addressed at three ‘levels’: Approach, implementation and application.
Figure 1.7 Research outline – Cartographic visualization is addressed at three ‘levels’: Approach, implementation and application.

2. Scripting: A Methodology for Dynamic Cartography

Figure 2.1 Tk Example 1 – Basic widgets classes and communication.
Figure 2.2 Tk Example 2 – Reconfiguring widget option values.
Figure 2.3 Tk Example 3 – Creating geometric items on a canvas widget.
Figure 2.4 Tk Example 4 – Dynamic graphics, transient symbolism and observer-related behaviour.
Figure 2.5 Map Example 1 - Static cartographic symbolism in Td/Tk.
Figure 2.6 Map Example 2 - Re-expression in Td/Tk.
Figure 2.7 Map Example 3 - Transient symbolism in Td/Tk: Changing the map.
Figure 2.8 Map Example 4 - Transient symbolism in Td/Tk: Changing multiple Views.
Figure 2.9 Map Example 5 - User interaction with symbols: Interrogating for values in Td/Tk.
Figure 2.10 Map Example 6 - User interaction with symbols: Brushing between views in Td/Tk.
Figure 2.11 Visual variables - Definition and specification in Td/Tk.

3. Dynamic Mapping for Visual Thinking: A Case Study

Figure 3.1 Running Td/Tk with the 'wish' Application.

Figure 3.2 'cdv' Geometry File Format: ASCII view shown in the 'Polygon Boundaries Quick View' through the Data Selection GUI.

Figure 3.3 'cdv' Geometry File Format: Geometry view shown in the 'Polygon Boundaries Quick View' through the Data Selection GUI.

Figure 3.4 Observer-Related Behaviour: Brushing for Polygon Ids in the 'Polygon Boundaries Quick View'.

Figure 3.5 'cdv' Attribute File Format shown in the Data Selection GUI.

Figure 3.6 The main 'cdv' windows: The view control panel and base map.

Figure 3.7 A dot-plot showing values of enumeration units using location as the visual variable.

Figure 3.8 Dynamic comparison using the variable selection list box on the view control panel.

Figure 3.9 Dynamic e-expression. Interactive classification of a choropleth.

Figure 3.10 Observer motion: Zoom and pan functionality in 'cdv'.

Figure 3.11 Linking views: Statistical groups displayed in a geographic view.

Figure 3.12 Dynamic graduated circle maps: Numbers of cars owned in wards in Leicester, 1991.

Figure 3.13 Non-continuous population cartograms created using Dorling's (1993) algorithm.

Figure 3.14 Geographic brushing between views in 'cdv'.

Figure 3.15 Linked views: Choropleth, cartogram and dot-plot.

Figure 3.16 Bivariate symbolism in dot-plots.

Figure 3.17 Dynamic scatter-plots: Linking between views.

Figure 3.18 Brushing for focus: Selecting data sub sets from a geographic view.

Figure 3.19 Brushing for focus: Selecting data sub sets from a statistical view.

Figure 3.20 Multivariate colour composites mapped in geographic and population space, US Midwest.

Figure 3.21 Multivariate colour composites and associated box-plots: Age structure, Leicestershire, UK.
Figure 3.22  Parallel dot-plots with a selected case highlighted.
Figure 3.23  Dynamic parallel-coordinates plot for selected cases.
Figure 3.24  Dynamic parallel-coordinates plot: Age structure, Leicestershire, UK, 1991.
Figure 3.25  A simple scripted extension to 'cdv': Colour composites using relative and absolute figures.
Figure 3.26  A simple scripted extension to 'cdv': Brushing by geographic distance.

4. Extending the Approach: Dynamic Mapping for Exploring Representation

Figure 4.1  Map Example 7 - Spatial smartness: Closest neighbours, brushing and contiguity neighbourhoods.
Figure 4.2  Calculation of topological relationships between polygons in 'cdv'.
Figure 4.3  The contiguity probe for spatial querying in 'cdv'.
Figure 4.4  Interrogation with transient symbols in an enduring additional view: Attribute composition.
Figure 4.5  Map Example 8 - Dynamic views to aid interpretation of cartographic abstraction.
Figure 4.6  Observer-related behaviour that permits the investigation of cartographic abstraction.
Figure 4.7  Performing local statistical queries with the contiguity probe.
Figure 4.8  Performing multi-scale local statistical queries with the contiguity probe.
Figure 4.9  Using the contiguity probe and local statistics to produce a local covariance matrix.
Figure 4.10  Interrogation of a covariance matrix.
Figure 4.11  Interrogating a covariance matrix at a spatial lag of two zones.
Figure 4.12  Adding a weight function to the index of spatial autocorrelation.
Figure 4.13  Interactive variation of distance functions in a weighted local statistic.
Figure 4.14  Interactive variation of zone centroid locations in a weighted local statistic.
Figure 4.15  Interactive variation of locations at which a weighted local statistic is measured.
Figure 4.16  Investigating the variogram cloud - interaction with a linked map and statistical graphic.
Figure 4.17  Map Example 9 - Multiple linked views of spatial data.
5. Extending the Approach: Dynamic Mapping Examples and Applications

Figure 5.1 Visualization of Schwalm-Nette time-space data set - Dynamic comparison.
Figure 5.2 Visualization of Schwalm-Nette time-space data set - Dynamic re-expression.
Figure 5.3 Visualization of Schwalm-Nette Time-Space Data Set - Geographic brushing techniques.
Figure 5.4 Visualization of Schwalm-Nette time-space data set - Temporal querying.
Figure 5.5 Viewing neighbourhoods in statistical graphics with transient symbols.
Figure 5.6 Observer-related behaviour that permits the investigation of cartographic abstraction.
Figure 5.7 Adding geographic information to a statistical data view - Colour composite.
Figure 5.8 Adding geographic information to a statistical data view - Glyphs and colours.
Figure 5.9 The 'scatter map' - Glyphs representing geographic locations.
Figure 5.10 Steps from the 'scatter map' sequence between map and scatter plot.
Figure 5.11 Visualizing neighbourhoods and distance weighting coefficients from a trace.
Figure 5.12 Interactive local parallel-coordinates plots for multivariate spatial data query.

6. Extending the Approach: Virtual Environments and Multimedia

Figure 6.1 A Transformational View of Cartography (Tobler, 1979): Virtual Reality.
Figure 6.2 A 360-degree panoramic image taken at Haytor, Haytor Down, Dartmoor, UK.
Figure 6.3 Creating panoramic imagery using a standard digital camera.
Figure 6.4 'simplePanZoom' - Panoramic image viewing in Td/Tk.
Figure 6.5 Calculation of the relative bearings of geo-referenced images.
Figure 6.6 'panoraMap' - A navigable virtual environment using geo-referenced panoramic imagery.
Figure 6.7 Extending the forms of representation available to the map-maker and map user.
Figure 6.8 Synthesis of qualitative and quantitative data for exploratory analysis.
Figure 6.9 'panoraMap' - Visualization map use in a virtual environment.
Figure 6.10 Synthesis of qualitative and quantitative information for visualization.
Figure 6.11 Interactive techniques for assessing the spatial nature of multivariate distributions.
7. Discussion, Context and Consequences

Figure 7.1 Research undertaken – Cartographic visualization has been addressed at three ‘levels’: Approach, implementation and application.

Figure 7.2 Time taken to produce maps for a single platform.

Figure 7.3 Comparison of times taken to produce maps on variety of platforms.

Figure 7.4 Comparison of speed at which maps of different complexity are produced on two platforms.

Figure 7.5 Assessing future trends - Speeds at which maps are produced on a PII 400.

Figure 7.6 New item symbolism - Incorporating the ‘focus’ visual variable.

Figure 7.7 Varying the focus of glyphs that show locations to represent geographic certainty.

Figure 7.8 New item symbolism - Incorporating the ‘orientation’ visual variable.

8. Conclusion and Futures

Figure 8.1 Further extending the approach - ‘panoraMap’ and the Internet.

Figure 8.2 Scaling the approach - Cartogram visualization at the national scale.
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization

Approach, Implementation and Application

1. New Maps for Old: Dynamic Maps and Contemporary Map Use

Jason Dykes

Department of Geography

University of Leicester

UK
1. **New Maps for Old: Dynamic Maps and Contemporary Map Use**

1.1 **Mapping and Enumerated Information**

Geographers, and others used to working with large amounts of information with a locational component, are familiar with elaborate methods for representing the complex multi-dimensional phenomena of the real world as a series of two-dimensional symbols on a plane. Graphic communication as a means of explanation can be traced through the centuries (Ferguson, 1977) and maps play a major part in much geographic research and teaching. Consequently a huge variety of techniques have been developed for encapsulating spatial information on paper and communicating aspects of it. Such aspects might include locations, data values, the error inherent in the measurement, geographical and statistical outliers, and trends either across the region mapped or in localities within it.

1.1.1 **The mapping transform**

Tobler (1979) presented an explanatory view of cartography as comprising of a series of three transformations occurring in a linear fashion between four states:

- the real world
- a recognizable data abstraction that models the world
- a representation of that model
- an individual's interpretation of that representation

The transformations between these states relate to the processes of data collection, map-making and map reading respectively. This is shown schematically in figure 1.1 (Robinson *et al.*, 1995). When creating static maps cartographers attempt to represent information in a way that models either the world or the data set suitably and is relevant to the map purpose and the characteristics of the map viewer. In essence an effective second transformation is used to clarify the first transformation from world to data and to minimize the amount of noise associated with the third transformation by considering the perceptual skills, requirements and cognition of the likely map user. Traditionally the second transformation has been the focus of academic cartography, and much literature has concentrated on appropriate 'mappings' of graphic symbols onto elements of the model to be presented through the processes of selection, simplification, classification, exaggeration and symbolisation.
Most famously, Bertin (1983) identified six 'visual variables': shape; size; orientation; colour; value (the variation of lightness or darkness); and texture (pattern). He described maps as combinations of these that portray elements of reality through their location in two planar dimensions. The colour variable is now generally sub-divided into hue (wavelength) and saturation (purity of hue), and the focus of a symbol is often added to the list. The primary objective of using these visual variables to represent data is to exploit the human visual system's ability to recognize spatial structure that provokes appreciation of the salient characteristics of the data set (Robertson, 1988). Robertson states that "this is achieved by mapping relevant aspects of the data on to visual representations that can be understood intuitively by the observer" (p. 243).

In their definition of the scope of cartography Robinson et al. (1995) use Tobler's (1979) transformational model and in doing so embed cartography firmly in the process of scientific explanation. However, map production has traditionally been an arduous process, meaning that single visual representations have had to fulfil conflicting functions, as both accurate data inventories for ideation, and as vehicles for conveying specific messages (Board & Taylor, 1977). The intricate and time-consuming process of map-making has meant that most maps have had to accommodate a range of purposes and viewers who will put maps to different uses throughout the process of...
scientific inquiry, from the initial analysis of spatial patterns for ideation to the clear communication of results. Indeed Robinson et al. (1995) describe the job of the cartographer with the following statement: "Clearly there are many possible maps of the same information. The cartographer's task is to explore the ramifications of each mapping possibility and select the most appropriate for the intended task."

With multiple (and often unknown) map users and map uses this is an impracticable objective and so academic cartography has developed rules and guidelines have been designed so that clear, consistent maps are produced in order to optimise their potential utility and ensure that they are of broad application.

1.1.2 Enumerated information and cartography

These are, however, multifarious and complex issues that are best clarified by way of an example. Enumeration is a common method of performing Tobler's first transform (Tobler, 1979), by structuring the physical environment into manageable units for data collection, management and analysis. In doing so a set of recognized geographic information is created from the real world. Zones that often have no physical embodiment, or relevance to the task in hand, are used to agglomerate individuals for reasons of privacy, data management and collection on the ground. A popular and traditional way of mapping geographical information of this form is to produce shapes that represent the boundaries of the enumeration units and symbolize them in some way that indicates the statistical values collected for each zone. Such a representation is designed to enable a map-reader to recognize individual zones by their shape, identify an overall geographic distribution, detect variation within localities and, when classified accordingly, to read data values for individual units with some confidence. The representation is described as a 'choropleth'. The dawn of computer mapping made choropleths extremely prevalent for a number of reasons. These included:

- the ease with which simple vectors could be drawn and in-filled with shading patterns on rudimentary computers
- the evident link between the representation and the data model used for census and other enumerated surveys
- a perceived ease of interpretation of the map product leading to a belief that the third transformation could be achieved relatively predictably and successfully from such maps
- a strong desire to reveal some of the geography of hugely expensive surveys such as the decennial census of population in the UK and the availability of these data in digital format

However, a number of issues must be considered if choropleths are to be used successfully. These relate broadly to the numbers mapped, the areas used and the symbolism applied to the maps. Problems with the numbers revolve around the fact that mapping population data that are based upon counts of people in choropleths reflects information other than the population characteristics
that are actually mapped. The number of people enumerated in each zone, the areal extent of the zone and the underlying population distribution each have a significant effect on the distributions. Problems with the areas relate to the fact that the data resulting from enumeration will depend greatly upon the precise boundary definitions used in the first transformation. This issue is often referred to as the modifiable area unit problem (Openshaw, 1984) and is illustrated and exacerbated by aggregating enumerated zones into new composite areas during the second cartographic transformation. Additionally a range of different maps and distributions can be created depending upon decisions made over parameters such as the classification and colour schemes used. This can result in very different potential interpretations that are, to an extent, controlled by the cartographer rather than any underlying geography (Monmonier, 1991). A full account of these issues, along with some guidance on how to map effectively, is provided by Dykes and Unwin (1998).

A long history of cartographic research exists to provide recommendations in the use of choropleth maps. Indeed the combination of popularity and the cited representation issues mean that the choropleth may well be the most heavily investigated map-type of recent times. Numerous experiments were conducted in the latter half of this century to produce recommendations on classifying and symbolizing the statistical distribution. These tended to take a behavioral, rather than cognitive approach (MacEachren, 1995), reflecting the need to produce an optimal compromise in order to produce single solutions that best suit most potential map users and uses and accommodate their likely reactions.

Classification schemes, in particular, have received a lot of attention. These can be designed to aid the recognition of data values, either by selecting a technique that ‘fits’ the distribution (Evans, 1977), by minimising the total difference between data value and classified value (Jenks & Caspall, 1971) or attempting to classify to produce an homogenous spatial distribution (Monmonier, 1972; Dykes, 1991). A range of recommendations exists for the use of colour schemes when producing such maps. Mersey (1990) bases her conclusions on the objective of communicating patterns to map readers, and Brewer (1994) identifies suitable schemes dependent upon the data type and nature of variation. The production of non-classified choropleths was first suggested by Tobler (1973), who used variable width cross-hatching, and debated by Dobson (1973) before Muller and Honsaker (1976) extended the idea by using emerging technology to produce a large number of ‘continuous’ grey shades. Dobson (1983) demonstrated that the use of redundant symbolism was a successful technique, improving the accuracy of responses to map questions concerning location, categorisation and comparative judgement.

Most of this research and indeed even some of the problems that it is attempting to overcome, relate to deficiencies in the second transformation that compromise the way in which a map-user might perform the third transformation, that of map use. They focus on the data model rather than
the problems with the numbers and the areas or even the world, people and issues that are being analyzed and from which the abstractions are made. Indeed post-war cartographers could be easily accused of failing to look further than their data models.

These issues and inadequacies, along with the noted popularity of the representation, mean that enumerated data sets and their representation constitute the primary and recurring focus of the work presented here.

1.2 Recent Developments and Current Issues

The 1990’s have given rise to a series of advances in computing technology that have had a colossal effect on the types of map that can be created from a data set and the types of data that are available. This technological progress led MacEachren (1997) to proclaim that "Cartography has undergone a profound change over the past decade".

Initially, advances in computing led to a number of efforts to overcome some of the limitations of the static choropleth that had been identified and contended with for a number of years. These relied upon running larger numbers of iterative calculations to optimise some aspect of the map, rather than advances in graphical techniques. For example, methods were developed to optimise the necessary ‘trade-off’ between classifying mapped values in choropleths for data recall and retaining values that retain a high degree of accuracy (Jenks & Caspall, 1971). Considerable research effort went into achieving cartographic products from which values can be read, and patterns detected, for numerous different variables. Even with the advent of digital spatial information and huge data sets containing massive amounts of complex temporal and spatial behaviour much of the initial endeavour went into computational reproductions of traditional, manual, cartography (Fisher, 1998).

Slowly, but steadily, technological advances made increasingly complex computer processing and graphics more and more speedy, affordable and accessible. Alternative, more imaginative uses of computer power were developed specifically for computer generated maps as the new software medium enabled less constrained and more impressive cartography to be developed. As huge numbers of maps were able to be generated at speed, and displayed on high fidelity screens with good colour quality and definition, potential map uses and so map objectives, began to change. In addition to the new perspectives on old issues concerned with static maps, Fisher et al. (1993) identified three consequences of changes driven by evolving technology:

1) The Availability of New Visual Variables
2) The Development of New Computational Possibilities
3) Increasing Levels of Interaction Between User and Map

1.2.1 New visual variables

A series of new ways of symbolising result from the described technological innovations. DiBiase et al. (1992) present a thorough assessment of the dynamic features of symbols that can be used to map data post-Bertin. They offer three additional primary visual variables for dynamic cartography: order, duration and rate of change, which, if a sequence of images is spatially dependent, can be applied to depict existence, attributes or change in one of these (MacEachren, 1994).

The temporal dimension of an animated map has a number of uses other than for direct symbolism. Campbell and Egbert (1990) outline a long tradition of attempts at animated cartography that order a sequence of predefined static views with slightly different characteristic in order to highlight change. Early examples were produced using analogue media, including Tobler's (1970) film showing the development of Detroit and Moellering's (1976) study of traffic accidents. Early digital examples include an application simulating forest evolution (Van Voris et al., 1993) and another showing surface temperatures across the USA over the last century (Weber & Buttenfield, 1993). The ordering of images is often chronological in animated displays, but examples exist where images are ordered by other attributes. For example Monmonier (1992) orders maps of electoral results by attribute value to convey the correlations of variables and stability of patterns over time and by region to highlight geographic patterns. Salvemini (1991) maps the population of all townships in Italy with graduated circles that appear in order of size to facilitate understanding in an otherwise cluttered display. Slocum, Robeson and Egbert (1990) suggest that students recall information more successfully from classed choropleths when presented with an animated sequence of consecutive map classes rather than traditional static maps. Fisher (1993) uses random ordering to show error in soil maps and land cover classifications of remotely sensed images. An early example by Shoup (1979) discusses the advantages of colour cycling by successively changing colour tables.

1.2.2 New computational possibilities

The evolving technology has also had an effect on the design of static maps, the use of which remains an important factor for modelling aspects of data in a revealing manner in order to gain insight and to communicate spatial patterns to a large audience despite possibilities for dynamic mapping. And in essence, many dynamic maps are used as a series of user-specified static representations. Even when analysing data interactively the spatial scientist may operate by altering visual variables to produce successive 'static' maps to illustrate and highlight patterns in an exploratory manner until a notable trend is uncovered. The production of a static cartographic representation can thus be a creative process, in which a model of reality is formed by the application of a series of parameters to achieve a
particular goal. A further role is illustrated by the reminder that "one cannot publish interactive systems in a journal" (McCormick et al., 1987, p.6). Whilst this factor is being overcome in technological terms (Dykes, 1996), not everyone has the time or capacity to process their colleague's data, and a static map can, in the same fashion as a graph or chart, quickly communicate relevant aspects. Gilmartin (1982) gives further credence to the role of the static map reporting that pictorial representations and iconic memory are more resistant to fading than other forms of memory.

With the advent of digital data storage and real time computer mapping the need for raw data values to be tabulated graphically is reduced dramatically. The development of Geographical Information Systems, or GIS (Burrough, 1986), with specially structured spatial databases distinct from mapping and statistical analysis functionality reduced this need greatly. GIS form a stable standard data model that can be repetitively accessed and updated — a cartographic base station from which multiple different maps can be produced as and when required. As a whole variety of views of a spatial data set became increasingly easy to produce, Tobler's (1979) second transformation, that of map making, became an even more essential focus of modern cartography. Techniques for cartographic modelling (Tomlin, 1990) are concerned with formalising this transformation of spatial data sets into graphics. A number of new static mapping possibilities emerged as a result of the ability to produce maps for specific purposes from archives of digitally stored data. These possibilities represent new
transformations from digital data to imagery and can be thought of schematically as a widening of the link between geographical information and the representation (figure 1.2). They include the calculation of derivative information and projections of three-dimensional views.

Derivative information

Increasing computer-processing power facilitates the calculation of derivatives from geographic data and makes more complex derivatives more achievable. The production of multi-scale slope maps from elevation matrices (Wood, 1996), for example, could not have been considered until relatively recently. A good example of the derivation of additional useful information for enumerated data sets is the population cartogram. Cartograms aim to overcome one of the 'problems with the areas' of choropleth maps. They are an optimised trade-off between an abstract space that shows the characteristic under study and a recognisable geographic space and distribution. Symbols are used to represent enumerated zones as is the case with choropleths, but the symbols are sized so as to portray the number of people represented by the enumeration unit rather than its' geographic extent. Increasingly efficient algorithms are becoming available to perform this optimisation (Upton, 1991; Dorling, 1992; Gusein-Zade & Tikunov, 1993). Effectively cartograms are rejecting the traditional planimetric mapping of geographic space onto the plane where locations can be read and identified using Cartesian coordinates in order to emphasise the statistical distribution. Some retain aspects of zone shape to aid geographic interpretation, whilst others discard this information to aid statistical comparison by mapping similar symbols of different sizes. Dorling's 'A New Social Atlas of Britain' (Dorling, 1995) includes a large number of these non-continuous cartograms that are effectively using the spatial dimensions of the page to overcome the clutter that Salvemini (1991) addressed by adding a temporal component to maps. In his atlas Dorling argues that social spaces can be advantageous for studying social structure noting that population cartograms give prominence to people depending not on how much land they occupy, but on their numbers. Applications that interactively demonstrate Dorling's optimisation technique (Dorling, 1993) have been developed recently and are available in the public domain (Herzog, 1998).

Projections of three-dimensional views

The computation that goes into calculating less abstract projections of terrain and other surfaces and solid models is also being achieved more quickly. The use of oblique views for representing landscapes is a traditional cartographic technique suitable for a number of purposes, however it is a time-consuming process that requires great skill when performed manually (Imhoff, 1953; Berann, in Troyer, 1999). Computer systems have automated this technique, and modern software extended it. Surfaces can be produced with colour drapes that show photographic imagery of the modelled area or additional information derived from the model. The former can be used for landscape evaluation.
exercises (Lovett, 1999), educational purposes (Dykes et al., 1999) and planning scenarios (Miller, 1999) whilst the latter is useful for assessing the quality of the model – effectively the success of Tobler’s (1979) first transformation (Wood & Fisher, 1993). Advanced software systems enable the viewer to specify the position from which they are to view the model and control cartographic parameters such as the perspective, apparent depth and detail. Several systems can simulate haze (Wood, 1998), create landscapes as they might look following certain future scenarios (Ervin, 1993) and test models such as those describing the behaviour of fires (Bishop, 1993). Additionally commercial software is available that enables users to add textures to different features in the landscape and to generate likely patterns of snow cover and vegetation and effects such as skylines and reflections (VistaPro, 1999).

In addition to surface models, modern computing has enabled fully three-dimensional solid models to be developed and viewed. These applications represent voluminous data sets such as those of solid geology, hydrogeology and atmospheric and oceanic processes. Early examples include those of Eddy and Looney (1993) who showed a water contamination plume at the Savannah River nuclear plant in South Carolina, while Rhyme et al. (1993) illustrated a number of limnological and atmospheric applications including three-dimensional images of the Antarctic ozone hole. Subsurface oceanographic and geological applications have in common a paucity of spatial locations at which a large number of vertical measurements are made. This results in interesting interpolation and data management issues (Raper, 1989).

1.2.3 Dynamic maps

The third consequence of changes in the technology behind mapping identified by Fisher et al. (1993) relates to the increasing levels of interaction between map users and the products that are used. This is typified by a simple example using the choropleth map. After decades of academic discussion about the best way to classify choropleths Ferreira and Wiggins (1990) produced software containing a ‘density dial’ that enabled a user to interactively vary the statistical value at which the interval between two classes occurred on a computer map in real time. On the schematic representation of the process of cartography this interaction is represented by two-way arrows between the user and the representation, and the representation and the model. The map acts as an interface between the spatial data model and the requests of the user, which are generated by their responses to their interpretation of the representation (see figure 1.3). A non-spatial equivalent is the interactive histogram feature of MANET (Unwin et al., 1996) where users can drag the edge of a bin on a histogram and see the width of all bins change, along with the classification and resultant data distribution.

Shepherd (1995) provides a broad classification of the various ways in which maps can incorporate dynamism. His typology of ‘the dynamic in visualizations’ encompasses a wide range of changes
that can occur in map displays that he terms 'map behaviour'. These can come from a variety of sources and each form of behaviour can assist the observer in making sense of the displayed information. One such source is the map user, and under 'observer-related behaviour' Shepherd (1995) describes the type of interaction identified by Fisher et al. (1993). These forms of behaviour include observer motion, object rotation, dynamic comparison, dynamic re-expression and brushing.

 Observer-related map behaviour has a wide variety of applications, providing the user with control over which aspects of a spatial data set they see, and how and when they see them. In two dimensions observer motion and object rotation equate to varying the observer's viewpoint by rotating, zooming and panning. There are analogies with folding, rotating and using a magnifier on a static paper map. The remaining types of behaviour, dynamic comparison, dynamic re-expression and brushing involve varying map symbolism over time.

 Shepherd's use of the term 'brushing' includes a whole host of data identification and selection procedures that can be applied to maps that enable users to interrogate specific symbols in order to elicit additional information. The techniques date from the late 1980's when statisticians recognized the analytical utility of direct manipulation and instantaneous change in their graphs (Becker, Cleveland and Wilks, 1987). Techniques were introduced whereby cases could be...
identified with 'transient labels', deleted from a view to provide a 'focused' subset and linked by the 
'transient highlighting' of selected cases in statistical views of data and the corresponding cases in 
separate views. Each technique involved temporarily symbolizing with a unique colour or shape. 
Many of these techniques were appropriate and applied to rudimentary maps. For example, by 
plotting pairs of coordinates on orthogonal axes a spatial view of the data could be used to locate 
selected cases on statistical views and select regions on spatial views for statistical analysis. The 
addition of a brushable view containing polygonal boundaries for enumerated data resulted in a 
more familiar map representation (Monmonier, 1989), and techniques for geographic brushing 
were quickly enhanced and developed (Haslett, Wills & Unwin, 1990; MacDougal, 1992).

'Dynamic re-expression' involves alternating more than one graphical version of a data set. 
Shepherd (1995) generalizes this technique to include the interactive modification of cartographic 
display parameters such as the classification intervals as demonstrated by the density dial (Ferreira 

'Dynamic comparison' entails displaying more than one data set, in the same coordinate space at 
different (usually successive or cyclical) times. Map users dictating the speed or sequence of 
animated frames can be considered to be undertaking 'dynamic comparison' in a more interactive 
form that the canned animations described in section 1.2.1 above. The techniques are an 
extension of Tukey's (1973) ideas on alternagraphics.

1.2.4 Combinations and additions

The consequences of these technological improvements do not occur in isolation. When the 
variable changing in a view that is being dynamically re-expressed is the viewer's location 
movement is simulated, and when the model being used as the basis of the map is three-
dimensional and the projection oblique, a fly-by is produced. Dramatic examples of fly-bys include 
the films of planetary exploration created by NASA based on digital data collected by satellite, and 
the animation through three-dimensional representations of socio-economic data produced by 
Dorling and Openshaw (1992). These can be calculated in real time with observer-related control 
over the flight path and other cartographic parameters. A useful means of gaining insight into 
familiar or unfamiliar data can result, which may represent a realistic or abstract space (e.g. Wood, 
1996; Stynes, 1996).

Additionally techniques for brushing and interrogating dynamic maps can be applied to three-
dimensional models (Wood, 1998), derivative information such as cartograms (Dykes, 1997), 
animations and maps using temporal symbolism.

Shepherd's (1996) typology of "the rich vocabulary of temporal behaviour that is possible in 
dynamic maps" differentiates between six ways in which time may be used in mapping. In addition
to the elements of dynamic cartography identified by Fisher *et al.* (1993) these include agents that reveal information about the data set (for example Openshaw and Pereé's (1996) Space-Time Attribute Creatures) and aesthetic symbolism (such as moving rain symbols on TV weather maps). Relatively sophisticated representations can result from combinations of this wide range of recently developed and continually developing mapping possibilities.

1.3 Dynamic Map Use

The advances in computer processing power and graphics and associated new computational possibilities described here have evidently changed the way that maps look and given them behaviours that were not previously possible. The observer-related techniques identified above involve interaction with dynamic views of data either by selecting cases in single or multiple views and applying transient symbolism to the graphics that represent them to show some characteristic of the case or by reproducing successive views of a data set with minor variations in the cartography to show some change.

The ability for users to interact with maps that symbolise in such a temporary way and the development of techniques that facilitate the production of maps that display these behaviours has resulted in a shift in the way in which maps can be and are being used. DiBiase (1990) has developed a graphic model accommodating the uses to which maps and data graphics are put. It ranges from exploring data in order to unearth trends and hypothesize, to presenting details to a wide audience (see figure 1.4). These two ends of the (carto)graphic continuum are termed 'visual
thinking’ (which occurs in the private realm) and ‘visual communication’ (which occurs in the public realm).

Maps that contain observer-related behaviour can be used successfully at both ends of the continuum. Those that suit the current needs of an individual user can be ideal for visual communication. When maps and graphics need only be viewed for an instant, can change, and may be recalled, numerous alternative cartographic representations can be used to represent a data set. Maps can thus be varied temporarily by the user to extract information from the data that is required for a specific, short term, task. An analogue example is the ‘You are Here’ map which provides buttons representing other locations that light up when pressed. The Chemical Release Inventory (Friends of the Earth, 1996) that centres information upon a user-defined postcode is an effective and accessible early digital example. It also illustrates that attribute information about spatial phenomena (in this case chemical releases) does not need to be displayed when maps are dynamic. Rather than symbolizing the types and amounts of chemical and times that they were released the map has links to tabular views of these data when release sites (simply located by a large red symbol) are clicked. Thus it is useful for communicating figures to individuals who can discover where chemical releases occur, and how much of each chemical is released from where. However it fails to provide a holistic display of the temporal or spatial distribution of the amounts of chemicals that are released or a multivariate display of the magnitudes of chemicals released at any location. These are the types of views that would be appropriate for ideation or data analysis.

Observer-related behaviours and short-term views of data sets are however especially appropriate for those who are searching through large amounts of raw data for relationships, trends and patterns rather than those who are trying to locate precise information or digest summaries provided by others. Maps that embody such behaviours provide a very real solution to the problems associated with the need to analyse the ever-increasing volumes of digital spatial data. They enable researchers to sift through spatial information graphically, seek and identify important characteristics of the data and theorise. A whole range of applications and developments show that maps are no longer being used exclusively as presentation tools but are playing an increasingly significant role in the process of knowledge construction (MacEachren, 1995).

1.3.1 Visualization map use and EDA with graphics

A graphic framework for identifying and differentiating the uses to which maps are put is provided by MacEachren’s (1994) [Cartography]³, where a three dimensional space of potential uses is defined by orthogonal axes representing three continua (see figure 1.5). These describe:

1)  The domain of the map: ranging between public and private
2)  The mode of map use: ranging between presenting knowns to revealing unknowns
3)  The levels of interaction provided by the map: ranging from low to high
The types of representation permitted by the observer-related map behaviour are particularly suited to the process of producing individual views of data that answer private expert queries in the attempt to reveal unknown information in a process that assists 'visual thinking' (DiBiase et al., 1994 - see figure 1.4). MacEachren (1994) labels maps that are designed for uses that tend towards these ends of the continua or this corner of the map use cube as 'visualization' (see figure 1.5). At this stage in the research process analysts are generally unaware of what they want to see until they see it but they will have ideas about how they might find it and what representation might be useful (MacEachren, 1995). The types of maps that are appropriate tend to be highly interactive and used by individual researchers for the exploration of unfamiliar information to aid the process of 'visual thinking'. The term 'visualization' is used to describe this form of map use that supports exploratory spatial data analysis throughout this volume.

It is not just spatial scientists who are aware of the utility of locating symbols on a plane to show the spatial distributions of one or more variables at the exploratory stage of the research process. Whilst geographic data sets are customarily mapped by cartographers using the two dimensions of the plane to portray spatial location, statistical graphics of data sets that are not geo-referenced often make use of location to show data values. Graphic representation of a variety of scientific data sets has been shown to be an effective technique for knowledge acquisition (Baecker, 1979; Baecker & Small, 1990; Sharp & Bays, 1992; Jennings & Mohan, 1991).
Short-term views that allow the user multiple perspectives of the same data and permit experimentation with a series of mapping transformations until patterns emerge are particularly appropriate. Two of Shepherd's (1995) 'observer-related' categories are particularly relevant here. 'Dynamic re-expression' which involves users changing symbolism (and other cartographic parameters) in order to explore a data set, and 'brushing' where cases are identified by the user in one data view and highlighted in another.

Such techniques are being used increasingly across the sciences at an exploratory stage, to investigate large, complex and multivariate data sets, by sifting through varying views of data until interesting representations appear. They are part of a wider trend in information representation and processing towards a reliance on the use of high-powered computers that produce images of data sets for exploratory analysis. Developments in computer graphics have fuelled this trend. An escalation in data volumes and advances in computer graphics technologies gave rise to the field of 'Visualization in Scientific Computing' (ViSC) as recognised in a report to the U.S. National Science Foundation (McCormick et al., 1987). Interactive graphical displays were identified as being of use to scientists for extracting ideas from masses of multidimensional data. Indeed the previous disdain for the visual depiction of data, in favour of 'objective' analytical approaches has been noted as being superseded by a recognition that human vision is in itself a powerful scientific tool (MacEachren and Ganter, 1990). A dramatic change in the approach of science in favour of visual evidence and visual analysis was reported as scientists modelled and observed their data and measurements via computer simulations and screen graphics, using workstation based visualization systems. With such tools researchers in the natural sciences can control model, display and view parameters in real time, yielding numerous concurrent views of data, which may be dynamically linked. Information is gleaned from data by investigating, transforming and re-displaying images on the computer screen. It may be necessary to check a pattern in many different ways to become confident that it is revealing structure in the data set (Unwin, 1999). This method of analysis, termed 'visualization' provides at worst an overview of the data set, and at best insight from which new ideas about the data are developed and tested. Visualization was thus recognised as a method, and product, that integrated the power of digital computers and human vision and directed the result towards facilitating insight across the sciences (McCormick et al., 1987).

The more interactive this process of visualization, the quicker questions can be answered and thought patterns developed. Additionally, more flexible graphic software will provide required views more quickly, and cause fewer obstacles to restrict the research. The methods used correspond with Tukey's (1977) exploratory data analysis (EDA) and Openshaw's (1992) definition of descriptive, rather than inferential statistics and exploratory geographical analysis tools that are used to find and describe patterns and relationships at the onset of the research process. A whole body of research has been undertaken and gained momentum in the field of graphical statistics that utilises observer-related
map behaviour to address these objectives for EDA. Many of the issues of concern to statisticians are equivalent to those being addressed by cartographers. For instance, Unwin (1999) identifies a distinction between statistical graphics for presentation and exploration that demonstrates a parallel to DiBiase’s (1990) visualization/communication map-use continuum. He notes that the former of these “must be displayed in a limited space, usually using only one graphic, for viewing by a potentially wide range of other people” whilst the latter “have in effect unlimited space, unlimited numbers of graphics, and are primarily only for the person who draws them,” (Unwin, 1999). Evidently the two fields can gain from collaboration, and subsequently the techniques developed in statistical graphics have been successfully incorporated into cartographic packages for spatial inquiry.

1.3.2 ESDA techniques for visualization

Stuetzle (1988) was an early proponent of dynamic graphics for statistical analysis. His ‘Plot Windows’ package produced histogram and 3D scatter plot views and took advantage of a number of observer-related map behaviours using an example data set containing 6 attributes for 47 Swiss counties. Each plot was produced in a different window, and could be temporarily varied through rotation, highlighting, ‘painting’ and ‘connection’. Rotation involved changing the projection used to display the 3D plots, which could be set to spin. Highlighting allowed symbols to be identified visually (through a change in symbol intensity) and a table of data values displayed. Painting used a brush to identify subsets of the data and views of this subset could be produced. ‘Connection’ described a technique whereby any county could be selected by clicking symbols with the cursor in one view and data relating to the same observation would be identified automatically in other views by a change in the symbolism of the corresponding symbol. ‘Painting’ and ‘connection’ have been more frequently referred to as ‘brushing’ since. The system provided excellent facilities for visual thinking and graphical EDA and pointed the way forwards for graphical statisticians.

The techniques were applied to the spatial domain by Monmonier (1989) and extended by MacDougal (1992) and Monmonier (1992). MacDougal (1992) included a map view in a prototype ‘Polygon Explorer’ which synthesised ideas from a number of sources. It permitted brushing, where polygon symbols representing areas could be coloured to show the locations of cases highlighted in the statistical views, and vice versa. These programs were perhaps the first examples of exploratory spatial data analysis (ESDA) that used dynamic graphics. Neither the scatter-plots nor the polygon map views incorporated in the software made use of the range of visual variables to map the spatial variation of attribute values. However ‘Polygon Explorer’ did have a significant spatial feature in that histograms could be selected to represent counts by area. Monmonier (1992) used a stylised polygon map with colouring and some point symbols to display attributes. He also reported on experiences and outlines principles for guiding users through sequences of maps, graphics and descriptive text.
The most flexible example of a system that permits dynamic, short term statistical views was provided by Haslett et al. (1990) who presented the REGARD software (Unwin, 1994) that supported a range of plot types. REGARD has geographical capabilities as it locates points, lines and areas in coordinate spaces and supports various means of brushing between views. Image backdrops are a useful feature for contextual geographical information. The system incorporates high levels of interactivity, with an histogram re-scaling feature that permits total freedom in classification, and the tenets of interaction, data on demand, and user control (so essential for ESDA with graphics), are upheld admirably throughout. However, REGARD does not use visual variables to symbolize statistical values on maps and thus provide a spatial view of attribute information. Highlighted symbols appear in distinct bright hues, but only location (to display location on the spatial views and value on statistical views) and shape (to identify linear and areal spatial entities) are used otherwise. In none of these examples was a series of spatially arranged symbols that represent data values, the kind of map that cartographers have developed to illustrate extreme and subtle, global and local, shaped, geographic and directional trends, incorporated. One interesting and effective exception in the case of REGARD is the use of a density estimation scheme to calculate the density of scatter-plot clouds and application of intensity/value symbolism to map it.

Egbert and Slocum (1992) included methods for cartographic symbolism in ExploreMap (or EMap). The software constitutes a ‘visualization tool’ for choropleth maps to provide users with control over the symbolism used in the representation (Shepherd’s ‘dynamic re-expression’; Shepherd, 1995) and permitted some ESDA, thus acting as an interface to the underlying data set. These two separate features were termed ‘Design Mode’ and ‘Explore Mode’ in the software. The former included tools for selecting, and transforming polygons, classifying statistical values, creating a colour scheme and adding text and a legend. The latter contained class sequencing, class toggling, brushing ranges within a histogram and interactive classification. Dynamic comparison was provided through the cycling of attribute classes in order, in a technique developed by Taylor (1989). EMap formed an important synthesis of statistical graphics and cartography, but was limited by the omission of any consideration of multivariate data, relying solely upon dynamic comparison, a major drawback when attempting ESDA.

These developments, and others, have contributed to a confidence in the use of graphics at the exploratory stage of the research process where spatial data sets are viewed and reviewed under changing cartographic conditions in an attempt to elicit patterns and trends that might be present, and to assess their validity (MacEachren, 1994). Software that is intended for fostering ideas and discovery rather than presenting conclusions, by providing dynamic displays that encourage experimentation with different combinations of data and graphic symbols, is termed cartographic visualization software by DiBiase and others (1994). Such tools, which utilise modern mapping
techniques, combine geographic and statistical information and present it in a dynamic, interactive and visual manner for ESDA are an attractive proposition for spatial scientists who are interested in assessing multivariate spatial distributions.

1.3.3 Towards a research agenda

These developments were quickly noted by the International Cartographic Association, which responded by initiating a Commission on Map Use. The Commission reported that a flood of new geo-referenced data, new scientific & societal demands and uses for those data and rapidly evolving geo-information technologies were, in combination, changing the roles of maps in society (MacEachren, 1998). The McCormick Report (McCormick et al., 1987) provided cartography with fresh impetus and recent cartographic visualization literature calls for interactive analysis of mapped data so that users can manipulate the traditionally imposed viewing parameters of a single map, in parallel to ViSC. Emphasis has thus been on dynamic, interactive mapping for data exploration at the expense of the static map.

The Commission noted increasing links between the fields of GIS, ViSC and virtual reality (VR) and identified the emergence of Geographic Visualization (GVIs). GVis was defined as an interdisciplinary research focus and set of tools that could enhance and fundamentally change the way in which scientists conceptualise and explore geo-referenced data, make decisions critical to society and learn about the world (MacEachren & Ganter, 1990; Taylor, 1991). According to the Commission a number of characteristics of ViSC are relevant to computer cartography. These include the ability to interrogate maps/images for information, the provision of numerous different views of a data set, the linking of views so that related information can be identified in each, and the availability of a temporal dimension for series and symbolism. The Commission identified the field of cartography as being well positioned to take a lead in ‘advancing the frontier’ and formed a ‘Working Group on Visualisation’ to address the implications of these developments (MacEachren, 1998). The working group noted that virtually all post-war cartographic research was directed towards assessing the use of static maps by individual ‘average’ map readers to retrieve specific information. It recognised that whilst the two dominant research streams in cartography, one technological, the other cognitive and perceptual, meant that the field was well positioned to take a lead in research in GVIs and even ViSC, a reassessment of the focus of cartographic research was required. The ICA responded by approving a full Commission on Visualization to focus on the use of dynamic maps as prompts to thinking. It defines GVIs as a form of information visualization that emphasises the development and assessment of visual methods designed to facilitate the exploration, analysis, synthesis and presentation of geo-referenced information and focuses attention on the implications of the change from a cartography focused on optimal maps towards one that emphasises multiple views and perspectives. Essentially cartographers aim to embrace the features of ViSC and embellish them with their techniques in order to develop effective means of handling and analysing spatial data. As the Commission and technology
matured and developed, researchers identified a set of four research priorities within the realm of cartography that respond to this change in focus. These reflect several aspects of visualization as an interaction between humans and computers that is directed towards exploratory analysis and deriving knowledge about geographic phenomena. In short they are:

- **Representation:**
  
  Notably taking advantage of advances in computer graphics to extend the forms of representation that are possible to facilitate visualization and equally to extend the type of objects that are represented.

- **Interface Design:**
  
  Extending cartographic principles, developed for static maps, into the realm of dynamic maps and the mechanisms provided to enable users to interact with those representations.

- **Database Visualization Links:**
  
  In order to enable GVis technology to deliver on the objective of aiding with the interpretation of increasing volumes of data being generated by our information society.

- **Cognitive Aspects of Visualization Map Use:**
  
  Developing a more complete understanding of spatial cognition and the perception of visual displays in order to fully harness the power of human vision and cognition for information synthesis and pattern seeking to compliment the raw information processing power of digital computers.

### 1.3.4 Some repercussions

This continuing, even accelerating, explosion of computing power and possibilities and associated profusion of new techniques for modelling and representing information is providing rich pickings for those with an interest in portraying spatial information dynamically for a whole variety of reasons. Technological advances have led to a tendency for geographic data to be digital, and computer mapping is becoming the standard. While there are still questions to be asked and research to be undertaken in the quest for a 'better' map, a new perspective has been identified that uses the computational possibilities and tools available to the cartographer to produce multiple views of the same, or related, data in order to investigate spatial information. In addition, a fresh confidence is apparent in the use of images rather than numbers for describing geographical data. Modern mapmakers evidently have a far wider range of tools and techniques available to them than did their predecessors, a cartographic superset that is being continually and rapidly augmented. A number of repercussions can be identified in consequence, three of which are of particular note here:

1) **The development of new techniques is an important research area.** This is indicated by the first of the research priorities of the ICA Commission on Visualization,
demonstrated by MacEachren's (1996) identification of the need for research into cartographic representation, both in terms of extending the types of objects that are represented by cartographers and the forms of representation that are used.

2) Maps are transient and can be provided in real-time 'on demand' from a data set, driving a welcome schism between the dual static map roles of data store and analytical device and meaning that in the software medium the 'optimal' map of ubiquitous utility is a dated concept. For a number of tasks and certain skilled users maps can be considered a spatial interface to geographic information that map-users browse by specifying and producing views of available data as desired and required. These spatial interfaces to information should still be regarded as maps due to their use of the geometry of the screen to represent the geometry of the world, but should not necessarily retain the codes, conventions and syntax of traditional cartography developed for analogue data storage and the paper medium. As maps they should still be designed for a particular purpose and user group and judged by their ability to achieve their objectives.

3) Any implementation, application or assessment of developing techniques within this context is a temporal 'snapshot' that should be regarded as a transient way-marker that provides experience and context in a tide of change. By assessing and relating their experiences researchers may better identify useful directions in this constantly changing field.

1.4 Mapping for Visualization

Within this context there is plenty of scope for addressing the dominant visualization research issues: for developing links between digital data and representation; extending the forms of representation; experimenting with interface design for users who need high levels of flexibility; and assessing cognitive aspects of these solutions. A suitable contribution to this progressing field would therefore include some of the following features:

1) The development of links between data and view that support visualization map use
2) The production and prototyping of novel views and forms of dynamic representation in a flexible software environment
3) Some assessment of the reaction to and utility of the approach taken
1.4.1 Data-view links that support visualization map use

The complexity and novelty of the techniques outlined thus far, and the multifarious nature of geographic data meant that at the onset of this study no integrated system existed that embraced the full range of visualization techniques available or provided the flexibility to address the research issues identified here. Whilst techniques and software have advanced in the ensuing eight years it could be argued that solutions still tend to be very much ‘closed’ in terms of their applicability to a range of data and visualization scenarios. Geographic Information Systems provide some of the capabilities found in ViSC systems. Pseudo 3D viewing, point and click interrogation, control over classification, interactive colour schemes and multiple viewpoints are commonplace and more advanced features such as real time fly-bys, abstract data transformations, variation of the viewed variable in real time and some linking of views are becoming more widespread. However the development of GIS from a quantitative setting has resulted in efforts being made to model spatial information in a database from which numbers can be extracted, statistics computed and replications of traditional computer cartography created (Fisher, 1998). Cartographic concerns have tended to involve automation of manual cartographic techniques and production of high quality output of superb precision. The development of data and graphical structures that permit the kind of real time dynamic display, visual interrogation and engagement between researcher and data that characterise ViSC, effectively the links between data and map, have certainly been secondary. Those who wish to employ aspects of ViSC in their analysis often have to use a GIS for geo-referencing and computing their data, before exporting to bespoke visualization software in order to animate, transform or fly by their data, or to link multiple maps and graphs of the data dynamically. Examples of such software are Macromind Director (Weber, 1994), Dorling’s (1993) PASCAL cartogram algorithm, particularly the ‘boot laces and sticky-tape’ method of producing a fly-by from various sources (MacEachren et al., 1994), and Wills’ REGARD program (Haslett et al., 1991) or MacDougal’s (1992) Polygon Explorer. The adoption of a cartographic data model that encapsulated the required geographic information and provided direct links to an open environment for visualization with potential for transient symbolism and interactive display would be a considerable step forward.

1.4.2 Production and prototyping of novel views and symbolism

Our new-found computing tools enable us to take ‘dynamic re-expression’ to the extreme by producing novel representations for specific uses to explore a spatial data set, and by applying entirely different (carto)graphic processes, often involving additional related data.

There are numerous instances of novel views being constructed of complex multivariate data sets, which either require powerful computation or dynamic graphics to display them fully. Statistical examples include the ‘projection pursuit guided tour’ (Cook et al., 1995) where multivariate data sets are projected into two dimensions, and parallel coordinates plots that use a non-Cartesian
space in which to map multivariate data (Bolorforoush & Wegman, 1988; Inselberg, 1995). Spatial examples include Dorling's (1993) algorithm to produce non-continuous cartograms where circle sizes symbolize population and the mapped spatial organization represents topological characteristics of the real world space, and the multiple alternative re-aggregations of census zones offered by Openshaw et al. (1995).

These examples tend to use software that is hard-coded in systems programming languages and compiled. With a few exceptions, notably through the ViSC 'data flow' paradigm and elegant bespoke software, possibilities for varying Tobler's (1979) second transformation and imaginative use of visual variables (Bertin, 1981; MacEachren, 1994) are limited. To fully explore the possibilities of representing spatial and statistical data in graphical form, and to take advantage of the potential for dynamic mapping fully, spatial scientists need more interactivity and flexibility. This is particularly important due to subtle variations between data sets and the constant need to develop cartographic representations that map the data appropriately and accurately. A series of tools that take advantage of the type of data model outlined above and enable users to specify symbolism and build appropriate views containing some of the dynamic capabilities outlined in this introduction would provide the environment to do this. Such an environment would require an approach at a lower level than the software solutions identified up until now.

1.4.3 Assessment of utility

Whilst we are aware that we are in possession of a whole range of potential tools for visualizing spatial information Shepherd (1995) reports that there is little experimental evidence upon which to base rules for making the most appropriate use of dynamic symbolism methods. Indeed there is insufficient indication that dynamic maps are advantageous even for communicative purposes. Slocum and Egbert (1993) conclude from their experiments with class sequencing that tailor made dynamic products for individual consumption might provide more advantages than their investigations uncovered. Koussoulakou and Kraak (1992) detect no overall communicative advantage from using dynamic maps, but suggest that those familiar with dynamic methods might find them more useful than static methods. Each of these conclusions points to the potential utility of individual transient maps for experienced users who are familiar with the data, for visualization purposes. Yet MacEachren (1995) notes how little we know about how maps work in the visualization context, nor about appropriate tools to provide, and DiBiase et al. (1994) recognise the problems of evaluating exploratory methods. McGuinness (1994) contributed evidence for using dynamic methods when she found that experienced users would create and re-create individual transient composite maps of relatively few variables when analyzing a series of distributions in her test of techniques used for a visualization task.
Lindholm and Sarjakoski's (1994) concede the difficulty in designing maps for experts and make a recommendation that the power and the flexibility of a map to be used for visualization should not be sacrificed. This is shrewd advice as we haven't yet begun to explore the potential for using multiple cartographic representations of data sets to allow researchers to see their data from different perspectives (MacEachren, 1995). And in the words of Unwin (1999) "Until you have the tools available to you in a usable form you cannot be sure how effective they will be in practice".

An environment that encourages experimentation with cartographic representation is essential if we are to start doing so. Indeed map design for visualization may currently be less concerned with appropriate forms of representation than with defining and providing an appropriate flexible environment for exploratory dynamic cartography. Such an environment must not sacrifice the flexibility that makes visualization an attractive map use and would be an inherent property of a suitable solution. The realm of visualization can thus be considered to include the interface features that enable users to interact with data through user-specified representations as well as the forms of representation themselves. Indeed in an assessment of software design issues for exploratory data analysis Bivand (1998) gives "special weight... to the extensibility of the implementation, to permit the user more control over the analysis". By producing such an environment and applying it to real applications we may begin to see whether the techniques are useful. Any contribution that did so would be a first step towards an assessment of utility and would thus be of benefit.

1.5 An Approach to the Visualization of Enumerated Data Sets

The research reported here aims to outline, implement and apply a visualization environment that addresses these issues. It builds upon the techniques reported thus far and incorporates some of the features specified as being desirable. It does so by addressing the data type identified as being of particular concern to computer cartographers, the much-discussed enumerated data set, examples of which are used to illustrate throughout. A number of responses exist that relate to the visualization of enumerated information. They fall into the three main categories identified by Fisher et al. (1993).

1) Production of Static Maps with Optimised Design Criteria
2) Novel Views
3) Increased Interactivity

1.5.1 Optimised design criteria

Initial experiments with optimised design criteria for static mapping were conducted and are reported upon in appendix A1. They aim to extend the work on choropleth classification by optimising in order
to maximise some spatial characteristic of the mapped data with the objective of taking greatest advantage of the abilities of human vision and visual processing by classifying to maximise spatial association. This was attempted by measuring a spatial statistic from a classified map and re-classifying continually under some constraints in order to maximise the chosen index. Two results dominated. Firstly, as the research progressed, the difficulty of using global spatial statistics to measure visually identified patterns became apparent. Whilst global views of mapped information provide the Gestalt with which users can identify and interpret regional patterns, spatial statistics that might be used to measure global pattern often rely on distance weightings to produce their indices of spatial association. Such coefficients work in contrast to human interpretation of the pattern, which focuses on broad swathes of colour and large scale pattern, by highlighting and responding to small scale variations in space. Secondly the increasing availability of interactive maps and ease with which they could be produced and accessed meant that the necessity to produce static maps for analytical purposes was rapidly diminishing. In combination these findings highlighted the need to see beyond and into the static map comparing global distributions with local variations and the utility and potential of dynamic maps for providing a means to do so. This resulted in a decision to focus on novel views and interactive mapping.

1.5.2 Novel views

Visualization techniques were deemed particularly appropriate for use with enumerated data due to the evident potential for representing multivariate attribute data in novel ways (Cook et al., 1995; Bolorforoush & Wegman, 1988; Inselberg, 1995), new and accessible algorithms for combining the spatial and statistical elements of such information (Dorling, 1993) and the mass of data available for analysis from expensive surveys such as the UK census of population. Some efforts seemed to contradict graphic good sense as outlined by Tufte's (1983) principle of maximising the proportion of graphical information that is used to represent data (e.g. Jenks & Caspall, 1971). The result is a challenge and opportunity to produce simple and effective views that utilise appropriate theories of data graphics (Tufte, 1983).

Such uncomplicated graphics are particularly suitable in the digital medium where information can be added to, or removed from, a display instantaneously at the request of a map user with transient symbols and symbolism. Users can effectively explore representations for additional details. Novel views can be particularly useful and appropriate in software. Brushing means that they can be associated with familiar representations to aid interpretation. Additionally the levels of abstraction involved in the transformation from the real world to the recognised quantitative geographical information are often extremely high. The provision of additional information about the transformation, or data resulting from other parallel qualitative transformations, may thus be a beneficial aid in the interpretation of real world phenomena from these data and address some of the criticisms of post-war cartography identified in section 1.1.
1.5.3 Interactive mapping

Thus techniques that outline and demonstrate the potential for interactive mapping for visualization form the main body of this thesis. A low-level approach is introduced, implemented in software, applied to address research issues and extended. Software is supplied as a major component of this submission. It outlines the implementation and provides novel views of data that are designed to take advantage of the interactive medium in which they are produced. Techniques for linking views by brushing between them, taking advantage of developments and utilising new data forms and sources are key themes as the medium is explored.

1.6 The Research Undertaken

By synthesising ideas from a number of overlapping research areas with a cartographic perspective and producing software that uses novel maps that reflect their dynamic nature and design for visualization some of the identified research issues might begin to be addressed. Software for visualizing enumerated data existed at the onset of this research but was either extendable yet bulky or flexible but lacking in spatial functionality (and still bulky). GIS fell into the former category with good spatial data handling procedures but limited static graphics whilst the data-flow paradigm visualization software such as AVS and Iris Explorer fell into the latter. Elegant interactive applications existed such as REGARD (Unwin, 1994) and EMap (Egbert and Slocum, 1992) but contained limited cartographic functionality and flexibility. A means of taking the best of each of these approaches and developing a visual environment for spatial data analysis with the flexibility for pursuing, developing and answering research questions whilst incorporating the vagaries of individual data sets was required. For example, the list of 'potential enhancements' provided by Egbert and Slocum (1992) shows the kinds of functionality that they deemed suitable for the visualization of multivariate enumerated information at the outset of this research:

- Support for 32 bit colour
- Interactive choropleth classification
- Advanced brushing techniques
- Variable symbology (Dependent upon user & data)
- Multivariate analysis
- Interactive interfaces to spatial information
- Maps that permit user-led dynamic comparison with interactive query
- Zooming/panning
- Spatial autocorrelation measurement
- Incorporation of topological relationships
Evidently an application for visualizing enumerated data requires these types of functionality.

1.6.1 Research aims

The object of this collection of research is to demonstrate a body of work that has realised this environment. By creating software that uses visualization for ESDA and applying it to address research issues and real world scenarios, the utility of the environment and techniques can begin to be addressed. The research builds on the situation identified in this introduction and postulates the following assertions:

- Spatial enquiry can take advantage of techniques being developed in a variety of academic fields
- Combining approaches allows us to re-define the 'map' as an extremely dynamic interface to information, a tool for visual thinking, rather than a single proclamation of the best estimate of some truth.
- This definition can be well served by creating maps that are as dynamic as a graphical user interface (GUI). An appropriate flexible environment for exploratory dynamic cartography can be achieved with a GUI builder that permits the use of traditional cartographic symbolism at the command of the enquirer.
- Such an environment and such map use makes additional forms of representation and novel analytical techniques appropriate.
- Attractive/working/usable/ software can be developed with the approach.
- The approach addresses a number of relevant research issues and is suitable for use with a number of data types and in a number of applications.

The aim of this thesis is to provide evidence in support of these assertions at a number of levels: approach, implementation and application.

1.6.2 Outline

Consider figure 1.6, which provides a schematic representation of the 'visualization' section of the cartographic continuum of map uses and identifies three levels at which the research presented here addresses cartographic visualization. At the level of 'Approach' a generic environment for producing cartography for visualization is introduced. This is achieved by using a programming language in an innovative way and demonstrating its suitability for dynamic cartography. A full outline and description uses elementary examples and simple data sets to demonstrate the language used, its syntax and its cartographic functionality. The approach supports two modes of operation which provide different levels of interaction. At the 'Implementation' level software and techniques that utilise this approach are outlined and provided in the accompanying appendices.
These examples demonstrate the utility of the approach and exhibit the different levels of interactivity available, so covering a range of the communication/visualization continuum. At the level of 'Application' examples are provided that pitch the approach into the research arena by utilising the methodology outlined, and on occasion adaptations of the implementations provided, to address specific issues that are of current concern to the academy. This is achieved in two distinct ways. One has a practical emphasis and involves the use of the implementations to perform cartographic visualization to analyse spatial data sets. A second is more theoretical and investigates general issues concerned with representation for visualization by providing examples and demonstrating some dynamic mapping techniques that are achievable using the approach and implementation employed.

At a broader level the thesis aims to demonstrate implicitly that visualization is an appropriate and useful methodology for exploratory geographical analysis that requires interactivity and flexibility, and that whilst the field can learn a lot from cartography a different set of rules, techniques and criteria are applicable to maps that are used for visualization that those that have been developed for communicative maps.

1.6.3 Structure
Empirical evidence is provided for the thesis along with a framework for demonstrating it and specific illustrative examples. The strategy is as follows and can be clarified by comparing figures 1.6 and 1.7:

Chapter 1 has catalogued relevant work that preceded and underpins this study and notes the state of play at the start of the research. It identifies some issues concerned with the status quo and assesses some attempts to address these along with the potential for progress. Recent trends and current research issues are described and the case is made for a contribution.

Chapter 2 outlines a methodology adopted, developed and cultivated throughout the period of research in order to contribute to the research effort. The approach taken has proved to be relatively flexible and robust. A full explanation of the way in which the approach supports dynamic cartography for visualization is provided.

Figure 1.7 Research outline – Cartographic visualization is addressed at three 'levels': Approach, implementation and application.

This volume commences by outlining an approach to cartographic visualization. It then provides software and an environment for visualization by way of example before employing the approach to address research issues and perform exploratory analysis.

Chapter 3 is effectively an 'initial results' chapter. It demonstrates how the approach taken provides a means for implementing visualisation software and a flexible visualization environment so that research can be undertaken both into and using visualization. The development and functionality of a software system that uses the described methodology is outlined. The software uses a particular form of spatial information and applies appropriate interactive techniques for
visualization of those data. It has evolved into a resource consisting of 126 script files containing 421 procedures for visualizing spatial information which consist of more than 22,000 lines of code. A working version of the software is provided in appendix B4 on the accompanying compact disk. All software provided here runs under Microsoft Windows™ 32 bit operating systems and comes with the example data sets utilised within the text. A fully formatted and documented version of the software is provided in appendix C3 with selected significant scripts provided in paper form in appendix A3.

Chapter 4 uses the approach and extends the implementation to address some issues in cartographic representation by demonstrating how interactive graphics can be programmed to provide additional information about the marks and symbols used to portray geographic information. Examples that permit exploration of cartographic and statistical representations of information are provided, as are additional example scripts that demonstrate the way in which the methodology enables this kind of interaction. These are included in appendix A4.

Chapter 5 demonstrates the flexibility of the approach by providing examples where it has been applied to conduct research. These range from relatively small scale 'add-ons' to provide additional statistical functionality, to extensions that apply the basic spatial display and interaction procedures, to applications using entirely different types of data sets. The examples include self-contained bespoke graphical user-interfaces and flexible, open source scripts. An assessment of the demonstrated flexibility, robustness and future proofing delivered by the scripting approach is provided.

Chapter 6 addresses a number of issues by outlining an additional application of the approach. The production of distinct software that provides a virtual environment suitable for the visualization of information collected during fieldwork is documented. The software addresses representational issues by incorporating novel media that were not available at the onset of the research, providing extremely interactive graphics for analytical use and incorporating exogenous contextual information. An assessment of the application, which is included in appendix B7, is provided.

Chapter 7 assesses aspects of the approach taken and points to the future by outlining a number of recent/current developments in the field along with opportunities for future development. Context is provided to the approach taken here and a number of methods are provided and proposed for formalising the composition of dynamic cartography.

Chapter 8 concludes with a retrospective assessment of the initial objectives and their satisfaction. A series of achievements and ongoing projects are described in order to demonstrate the utility of the research, its impact and the fulfilment of the stated objectives.
It is essential that a number of important appendices are used in conjunction with this volume. They contain additional information and demonstrate the quality and quantity of work involved, and provide a record of the research that resides in the public domain. They utilise the digital medium to present information in ways that cannot be achieved on paper and include ten pieces of software and over forty figures that contain information that is either interactive, animated, annotated or at a resolution that cannot be achieved within the confines of a printed volume.

The information is organised into four appendices distinguished by the letters A to D. Each contains a series of numbered elements.

Appendix A: Text appendices provided on paper including a report on research conducted into the use of spatially biased classification in choropleth mapping, notes on taking panoramic imagery and a selection of scripts from the ‘cdv’ software.

Appendix B: Software appendices provided on compact disk including dynamic mapping examples, the ‘cdv’ software complete with data, a demonstration showing how visual variables are defined and symbolised using the adopted methodology and interfaces to orientation and focus and the ‘panoraMap’ software which constitutes a prototype virtual environment for cartographic visualization.

Appendix C: Support materials appendix provided on compact disk including dynamic and high resolution figures to supplement those provided in this document, fully formatted and documented ‘cdv’ source code and a full guide to using the ‘cdv’ software.

Appendix D: Academic papers. A pull-out folder is provided containing reprints of papers appearing in a number of refereed journals and edited volumes including The Cartographic Journal, Computers & Geosciences, The Statistician and The International Journal of Geographic Information Science.

Where reference should be made to an appendix the letter and number of the appendix are mentioned in the text. Supplementary information exists for many of the colour figures presented here, either in the form of a dynamic figure or a piece of software. If this is the case the letter and number of the appendix are given below the figure. For example, many of the figures are provided in appendix C1 where dynamic and high-resolution versions are available on the accompanying compact disk.
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

2. Scripting: A Methodology for Dynamic Cartography

Jason Dykes

Department of Geography
University of Leicester
UK
2. **Scripting: A Methodology for Dynamic Cartography**

2.1 **The GUI Approach**

The types of map that are suitable for visualization map use vary from their static predecessors and counterparts in a number of ways. They do not need to show all of the information all of the time, nor do single representations need to perform dual or multiple roles as cartographic parameters can be changed successively. Each symbol can thus be the source of a new piece of information or behaviour enabling the representation to be explored for additional material. Indeed, extremely abstract, specific and non-spatial views of transient nature may be appropriate in this context. In terms of Tobler's (1979) transformations, the capacity for feedback between map and user in the process of visual thinking requires more flexibility in the second transformation - from recognised geographic information to the map.

In light of the proposed research, a software environment was required that enabled novel views to be produced and representation types and the objects being represented to be extended in an interactive way with rapid links between user, data and representation. Such a visualization environment would enable us to assess the utility of interactive techniques for visualization and should be flexible enough to permit modifications, view prototyping and experimentation. DiBiase *et al.* (1994) define cartographic visualization software as being dynamic displays that comprise of an interface tailored to a specific set of users, which is interactive, and encourages experimentation with different combinations of data and graphic symbols and intended for use in fostering discovery rather than presenting conclusions. In describing the types of interaction that cartographic visualization software might permit they list selection and transformation, which broadly concur with Shepherd's (1995) concepts of brushing and dynamic re-expression. They argue that if an analyst is able to vary their perspective of the data through selection and transformation, then meaningful relationships amongst data variables are more likely to be revealed.

To achieve this, a dynamic environment for visualization should permit the location of multiple graphic symbols in space with the variety of symbolism outlined by Bertin (1983) and others. It should manage and record the relationship between the symbols that constitute the map and the map user, and enable symbols to vary the way in which they symbolize. A more general case is the capability of a symbol to issue software commands when it is interacted with in a specified manner, in the way that graphic user interface (GUI) widgets do. Lindholm and Sarjakoski (1994) refer to such symbols as 'smart objects'. A dynamic map for visualization could be made up of smart symbols that contain spatial information and permit user-related dynamism through transient labeling, transient symbolism, and transient graphics and might link to other views and media. A smart map would provide the ability to create views by locating symbols in space, show
multiple perspectives through re-expression and brushing with transient symbolism and also be capable of incorporating the kind of multimedia linking that is being used successfully for communicative purposes (Cartwright, 1996) to provide additional information about the phenomena represented by selected symbols.

DiBiase et al. (1994) state that cartographic visualization software is tailored to a specific set of users with a specific set of uses. Yet the complex, multifarious nature of spatio-temporal data and geographic problems mean that applications tend to be unique in terms of their combination of data types and the kinds of questions that are appropriate. The statistical and graphical representations that are suitable to best permit the enquiring scientist to perform effective exploratory analysis may also be unique. The continuing development of new techniques and availability of new media further complicate the situation. In these circumstances the potential rigidity of a tailored interface might not be the most beneficial for the visual thinking stage of the research process when scientists are likely to need to view their data in a variety of ways which may not be apparent at the onset.

An environment for visualization is required that allows users to view and review their data, select and reselect geographical and statistical subsets, create and prototype suitable representations and ask questions and modify enquiries repeatedly. To retain flexibility and openness an environment of this sort should be essentially programmable, and allow experts to prototype their representations in the search for both individual maps that illustrate unknowns and suitable techniques which provide assistance in doing so. A solution that grants such flexibility and is becoming increasingly popular is that of the high level scripting language. Bivand (1996) notes that for exploratory analysis most commercial products permit and require extensive customisation and implement 'little languages' that support scripting and interface construction at a high level. Command-line interfaces such as those provided by scripting languages offer powerful, flexible structures for development that are suitable for use by specialists with a deep understanding of the information that they are using (Unwin, 1999). This description evidently includes those undertaking cartographic visualization for ESDA.

2.2 The Scripting Approach

Scripting is flexible as it uses high-level languages to connect sets of existing software tools or functions. Systems programming languages such as Java, or C++, are used to create these software components and so are complimentary to scripting languages. Consequently most major computing platforms provide both. Interpreted scripting languages tend to trade-off ease of use and flexibility against the speed of compiled systems programming languages. Recent trends in computing, including the availability of faster machines and better scripting languages and the
growth of the Internet, have made scripting increasingly appealing (Ousterhout 1998). Tcl, Perl, Python, Visual Basic, the UNIX Shells and JavaScript all provide means for combining software components through scripting.

System programming languages are well suited to building components where the complexity is in the data structures and algorithms, while scripting languages are appropriate in situations where the complexity is in the connections. GUI applications are a good example of the latter, as existing software functionality is used to define the widgets of a known class but the connections between the interface items, their precise behaviour and appearance and the internal functions differ. Using a high-level scripted call to a complex widget, such as a button, with arguments that specify the appearance and actions to be performed when it is interacted with makes interface development rapid and efficient. This is of interest to cartographers as the concept can be equally applied to the symbols that make up a dynamic map. Maps are specific organisations of known symbol components with particular appearance characteristics. Dynamic maps perform certain software tasks when a series of defined user interactions take place with specified symbol components. They are effectively GeoGraphical User Interfaces to the recognised geographic information that they represent through some specified cartographic transformations. If map symbols are transient and regarded as part of a wider GeoGUI to information the utility of an approach to mapping for visualization that uses a GUI as its basis rather than a traditional map becomes apparent.

Essentially, visualization map use requires the kind of transient symbolism and user-defined interface response that scripting can supply. Some scripting languages provide these possibilities through high level graphic commands that can be used as cartographic syntax. These link display, behaviour and data more fully in a model that revolves around graphic objects representing spatial entities and thus encapsulates spatial and attribute information in the map. Such a model fulfils the ICA Commission on Visualization research priority for smoother links between the data model and representation that are required for more responsive links between the user and the map. Scripting is also appropriate for 'rapid prototyping'. The savings in coding obtainable by scripting are particularly high for complex interface objects such as dynamic map symbols. Consequently new GUI interfaces, or maps, can be scripted with relatively few commands and flexibility is achieved. By programming at this level researchers can rapidly experiment with representation and prototype new dynamic views of data in a 'hands on' fashion to suit particular, transient, exploratory tasks and tackle other research priorities in visualization.

Computer users take advantage of GUIs to navigate through the huge amounts of information on their machine. The concept of the map as GUI, a dynamic window to assist the user in extracting and sorting information from a spatial data set is appropriate for the type of map use that MacEachren (1994) termed 'visual thinking'. The development of elegant graphical scripting languages means that the kinds of levels of interaction with which computer users are familiar
when they use windows operating systems can be expected of map interfaces. The rapid increase in machine performance means that the speed lost through scripting is becoming less significant and that more complex types of GeoGUI are achievable. Most importantly, the simple cartographic syntax means that 'casual programmers', such as interested geographers who require maps for visualization, can take advantage of the complex functionality and develop applications rapidly.

2.3 The Tk GUI Builder

Tk/Tk (Ousterhout, 1994) is an increasingly popular scripting language. Td, the Tool Command Language, began life in the late 1980's developed by John Ousterhout at Berkeley. It is similar to UNIX shell languages, offering familiar programming features such as variables, control flow, procedures and the ability to run other programs. It contains commands that access operating system services and can therefore access the file system and provide platform-specific interfaces to it whilst retaining platform-independent commands. The Tk graphical extension was added in 1990 to provide a series of GUI widgets types such as buttons, scale bars, menus and list boxes to connect Td programs. In combination Td/Tk supplies a rich environment for developing new code and linking existing software with GUIs. The language has developed rapidly, accumulating a large user community and being further developed by Sun Microsystems in the mid-1990's. The current status of Td/Tk is as an 'open source' package that is freely available for download from an independent platform company for Td/Tk formed to provide support and develop additional commercial tools (Scriptics Corporation, 1998).

The language includes a variety of means of symbolism and excellent support for dynamic linking, data reading/writing, image read/manipulate/display capabilities and means of interacting with external programs. The GUI exhibits many of the properties of dynamic maps allowing observer-related behaviour to be programmed quickly and in a consistent way. It locates graphic elements on the screen, and links them so that when the cursor moves over a widget, say, a button, its appearance changes, and when the mouse is clicked, an action takes place (for example, a program is run, or another widget, such as a listbox, appears). GUI builders allow programmers to create GUI applications by providing widget classes, such as buttons and listboxes, that can be configured, placed on screen and linked to each other and external processes. Td/Tk contains these functions and in addition has a lot to offer cartographers in terms of symbolism and syntax. By the early 1990's a community had developed around the language and by 1992 the Td/Tk core included a 'canvas' widget that enabled symbols to be located in co-ordinate space. The language provides an extremely flexible high level scripting environment and the range of symbolism required by cartographers. Consequently it is suitable for developing a range of highly dynamic
maps. Example code and dynamic mapping software is provided here to demonstrate the possibilities and document the process.

The Td/Tk language is interpreted, rather than compiled providing researchers with ultimate control whereby they can modify their maps from a command line, as they need a new angle on the information that they are mapping to answer successive queries of their data. The command line scripting environment permits rapid prototyping and the kind of level of engagement with data that is required for exploratory analysis. Sets of Td commands can be stored in script files and re-run at any time. The approach is thus a flexible one that provides open tools so that relevant views can be created with dynamic symbolism for specific data sets and individual needs, a crucial requisite of a cartographic approach to ESDA. A minimum of speed and performance are sacrificed for this flexibility as compilers exist and even the interpreted code is pre-compiled into byte-code to improve speed and performance. New views, specific to certain data types/sets or queries, can be created to take advantage of the symbolism options that are available in a suitable way. This mode of interaction is documented in electronically published papers (Dykes, 1997) and by a series of online tutorials (Dykes, 1996) that demonstrate the approach. Further support can be obtained from the growing Td/Tk user community, many of whom (myself included) are accessible at the click of an e-mail 'send' button to provide experience and exchange, debug and comment upon code. The 'comp.lang.td' newsgroup has evolved to fulfil some of these needs. Indeed Td/Tk has a broad community of users who primarily use the GUI widgets such as buttons and entry boxes to build interfaces for and between programs. However, the programming and graphics structures can be utilised to provide the kind of visualization environment described in the previous chapter.

An introduction to the language specifying how it can be used for the kind of interactive cartography for visualization that is the subject of this thesis, forms the body of this section. Examples of Td command formats and code are provided throughout. Formats are coloured in dark blue, commands in lighter blue. The outcomes of commands are indicated by a ‘>’ sign. Any lines of a Td script commencing with a ‘#’ hash are regarded as comments, and are used in the examples provided here to document the code which can be tested on the command line of a Td console. All code included here can be found, and interacted with, in software appendices B and C.

2.3.1 Td fundamentals

Td syntax consists of words separated by white space. The parser expects a recognised Td or Tk command followed by a series of arguments that dictate the operation of the command. The format of all Td commands is:

command [argument1 argument2 ... argumentN]
For example, the 'set' command takes two arguments, the name of a variable in which to store a string and the string to store in the variable:

```tcl
set n 1
```

Sets a variable to an integer value

Arguments that contain spaces are enclosed in "quotes" if they are to be expressed immediately, or (braces) if command substitution is to be delayed.

```tcl
setarray(l) "A Number"
```

Sets an array element to contain a text string

Associative arrays are useful structures in which to store spatial information. These can be defined to hold more complex data structures using an array name followed by an array identifier contained in (brackets). Any number of identifiers can be specified, each separated by a comma and containing any string.

```tcl
set array(2) 22.5682
```

Sets an array called 'array7  to contain a real number in element '2'

Another example of a Tcl command is 'puts', which can be used to put a string to the command line. It simply evaluates the single argument given to it and returns it to the screen on the command line. In the following example the value of variable 'n' is specified by a preceding dollar sign that leads to the variable value being substituted for the variable name.

```tcl
puts "The value of n is $n"
> The value of n is 1
```

A 'puts' statement will always express the information contained in the argument, so those variables that are preceded by dollar signs are replaced by the variable values as set above.

```tcl
puts "Here is $array($n) ( $array(2) )w
> Here is A Number ( 22.5682 )
```

Mathematical functions are available through the 'expr' command, which takes a mathematical expression as it's single argument and returns the result of that expression:

```tcl
expr $array(2) * 3
> 67.7046
```

An enormous number of Tcl commands exist allowing all sorts of complex command structures to be developed and enabling files to be read and written, strings to be manipulated and system variables and programs to be accessed and controlled (Ousterhout, 1994). Groups of commands can be given a single command name with the 'proc' command that takes three arguments: a name for the new command; a list of arguments; a list of commands to issue when the new command name is executed. For example, here is a simple procedure to add or subtract two
numbers and print the result. It takes three arguments. Two are numbers and the third is a string identifying a mathematical argument. The ‘switch’ command in the procedure is used to set a variable ‘n’ depending upon the value of this argument ‘t’:

```tcl
proc add-sub {n1 n2 t} { 
    # -- Use 'switch' statement to determine whether 't' argument is relevant 
    # -- and set value of 'n' to appropriate mathematical qualifier. 
    switch $t { 
        add {set n +} 
        sub {set n -} 
        default {puts "Bad Argument '$t'"; return} 
    } 
    puts "$n1 $n $n2 = [expr $n1 $n $n2]"
}
```

The new command is called with the command name and space separated argument list:

```
add-sub 12.6 4 sub
```

> 8.6

If an appropriate value for 't' is not detected in the ‘switch’ procedure the default action occurs, printing “Bad Argument” and returning from the procedure:

```
add-sub 12.6 4 mult
```

> Bad Argument 'mult'

The commands and data structures available in Td, the variable substitution and pattern checking functionality, and the ability to group and repeat these commands, are very suitable features for modelling and manipulating spatial information.

### 2.3.2 Tk fundamentals

Tk consists of several classes of windows objects or ‘widgets’. Objects from each class can be created by stating a widget class, such as ‘button’, and a name for the object. Names are hierarchical, rather like the DOS or UNIX file naming convention, but full stops (‘.’) are used to separate names rather than slashes (‘/’ or ‘/’). Each object derives default characteristics depending upon its class, but these can be varied by adding a relevant value to a legitimate option when defining a widget. Specific characteristics are encoded as ‘option-value’ pairs. For example a button can have a value designated for its ‘text’ option by adding `-text value`, or for it’s ‘height’ option by adding `-height value`.

**Tk Example 1**

Tk Example 1 shows instances of five widget classes, each with relevant options and suitable values. The `-bg` option sets the background colour of the specified instance. Colour options accept either an X-Windows colour or a hexadecimal red/green/blue specification preceded by a hash,
where '#F00' describes red, and '#FFF' white. Each instance contains class dependent dynamic
behaviours that respond to the users mouse/cursor actions in a way consistent with the operating
system that is being used. For example buttons depress when 'clicked' and scroll bars slide when
dragged with the mouse button. More specific observer-related behaviours can be programmed.
Command options accept any relevant Tcl/Tk command and execute it when a widget is clicked.
Once widgets have been defined a 'pack' statement is used to display them on the screen in a
specified location. The standard form of a command that creates a widget is:

```
widget-type widget-name [-option1 value1 -option2 value2 ... -optionN valueN]
```

Six example widgets are created in Tk Example 1, which uses and displays five different classes
(button, label, scale, checkbutton and canvas).

```
# - Tk Example 1.
button .b -text Hello -bg Red -command {set v Clicked}
label .l1 -text "A Label" -bg #0f9
scale .s -from 10 -to 20 -orient horizontal -variable s
checkbutton .k -onvalue On -offvalue Off -variable v
label .l2 -textvariable v -bg Yellow
canvas .c -width 50 -height 25 -bg SeaGreen4 -relief sunken

pack .b .l1 .s .l2 .c -fill x
```

Figure 2.1 Tk Example 1 — Basic widgets classes and communication. (B1)

Widgets are used as an interface to underlying information that is encapsulated in the widget
option values. Note that in Tk Example 1 the checkbutton '.k' has been defined to store a value in
variable 'v' dependent upon its state with the combination of '-variable' '-onvalue' and '-offvalue'
option-value pairs. This value is shown in the yellow label as it has a '-textvariable' option, which
specifies that the text to show in the widget is the value of 'v'. This means that whenever the
checkbutton '.k', or red button '.b' are pressed, the value of 'v' and the resulting label text change.
Appreciation of the dynamism provided by GUI objects such as this which pass messages between
one another and can be programmed to interact in a specified way is provided by interacting with
the interactive versions of these example scripts that are available in appendix B1 on the
accompanying compact disk. This combination of graphic element with data is typical of Tcl/Tk and
corresponds with the kind of smart map symbol required here.

**Tk Example 2**

Once Tk widgets have been defined, widget names become commands that can be used to effect
the values of the options of the named widget and so change its appearance and behaviour. In the
second example the same widgets are created as in Tk example 1 with identical options and
values.
A series of widget commands then reconfigure the appearance and behaviours of the graphical elements. The button `.b` has its background colour changed with a widget command of the form `'widgetname configure -option value7`. The canvas `.c` has its height increased and colour changed in a similar way. The value of any option can be altered including the 'command' option. Here the red button `.b` is reconfigured with a command that re-sets the scale widget `.s` to a value of 10 rather then setting the variable `v` to 'Clicked' whenever it is clicked.

```
# -- Reconfigure defined widgets with widget commands:
.s set 15
.b configure -bg SkyBlue1
.c configure -bg Orange -height 50
.b configure -command { .s set 10 }
```

Bindings provide a more flexible means of linking mouse/cursor events to widgets and so programming widget behaviours. A whole range of cursor events is recognised and managed by Tk. They are defined with commands of the form:

```
bind widgetname <Event> {Td Script}
```

```
# -- Tk Example 2.
button .b -text Hello -bg Red -command {set v Clicked}
label .l1 -text "A Label" -bg #0f9
scale .s -from 10 -to 20 -orient horizontal
checkbutton .k -onvalue On -offvalue Off -variable v
label .l2 -textvariable v -bg Yellow
canvas .c -width 50 -height 25 -bg SeaGreen4 -relief sunken

pack .b .l1 .s .k .l2 .c -fill x

# -- Reconfigure defined widgets with widget commands:
.s set 15
.b configure -bg SkyBlue1
.c configure -bg Orange -height 50
.b configure -command { .s set 10 }
```

```
# -- Use 'bind' command to define behaviours in Td on certain interaction events:
bind .l1 <Button-1> {set v "Label Clicked (1)"
bind .l1 <Button-2> {set v "Label Clicked (2)"
bind .l1 <Button-3> {set v "Label Clicked (3)"
```

Figure 2.2 Tk Example 2 – Reconfiguring widget option values.
In this example the label 'M l' is bound to commands that change the value of the variable 'V' depending on which of the mouse buttons is clicked over the label. The visual effect is immediate as the '-textvariable V option-value pair ensures that the value of V is always shown in the yellow label.

**Tk Example 3**

These features are useful in themselves, providing colour symbolism, text labels and communication between 'smart objects' (Lindholm & Sarjakoski, 1994), and the kind of flexibility and dynamism required in the specified visualization mapping environment. However, these standard windows interface objects do not permit use of the 'shape' and 'location' visual variables that are fundamental to so many data maps. The key to these forms of symbolism required to produce dynamic maps is the canvas widget class, which provides control over the location of objects. Canvases comprise of a planar coordinates system with an origin at the top left of the widget, and classes of graphic symbols, known as items that can be located on the plane. Item types include polygons, rectangles, ovals (of which circles are a special case), lines, arcs, bitmaps and images. Each class is located by a series of coordinates that allow the shape and position to be defined precisely and has its own characteristics that are specified by a series of option-value pairs added at the end of the item definition. Many of these relate to the visual variables used by cartographers to symbolise (these are demonstrated in section 2.4). Each item on a canvas has an individual identifier that can be used to manipulate these options, just as the button name was used to reconfigure the values of the options of the button widget. This arrangement forms the basis of an environment for smart mapping as items are dynamic objects that can be assigned spatial locations, symbolisation to reflect data values and behaviours based upon specified types of interaction.

In Tk Example 3 a larger canvas is used, and four canvas items are defined using the widget command that is a result of the canvas being created. These are followed by a series of arguments. The first of these is 'create', followed by the type of item that is required and a list of coordinates that define its shape and location. The standard form is:

```
canvas-name create item-type coordinates [-option1 value1 -option2 value2 ... -optionN valueN]
```

Oval items require coordinates as arguments to define their bounding boxes, and lines and polygons expect a list of x-y coordinate pairs that identify their constituent points in order. The '-fill' option specifies the colour with which an item is filled. Other options available include the colour and width of the item outline and the name of a bitmap that is mapped into the symbol. So, for example, a green oval with bounding coordinates (10,10) and (90,90), resulting in a circle of radius 40 units centred on (50,50) would be created with:

```
.c create oval 10 10 90 90 -fill Green
```
A number of units can be specified, the default being screen pixels. Items can be identified by text tags that can then be used to specify groups of items on a canvas. The `-tags` option is used to add one or more identifying labels to each item. These are the equivalent of widget names as they provide an identifier through which items can be reconfigured to change their symbolism or behaviour. Here colour, shape and simple geographic names are used as tags, but any string is valid. Items can be tagged with any number of strings and tags thus provide a way of encapsulating geographic attribute information to symbols. The following example creates four items of three different item types by way of example.

```
# - Tk Example 3.
canvas .c -width 100 -height 100 -bg Orange
pack .c
 .c create oval 10 10 30 30 -outline Black -fill Red -tags "red oval"
 .c create line 10 40 10 60 20 60 30 70 40 60 -width 6 -fill green
   -tags "green line water"
 .c create line 90 90 70 90 50 80 40 70 40 80 50 70 50 50 -width 4
   -fill red -tags "red line road"
 .c create poly 40 10 60 20 80 10 85 40 70 60 50 40 40 50 -fill #28C
   -tags "blue poly water"
```

Figure 2.3 Tk Example 3 – Creating geometric items on a canvas widget.

**Tk Example 4**

A series of symbols, such as those shown in the previous example, can be used to create a static map, with item shapes and locations representing geographic information and item options symbolising one or more attribute. Dynamic behaviour is enabled by the ability to reconfigure Tk canvas items in much the same way as widget option values were changed in Tk Example 2. Just as widgets can be reconfigured with widget commands, item tags can be used to reconfigure items and define bindings. Canvas commands are the key to dynamic mapping in Tcl/Tk. A canvas command to reconfigure the symbolism of an item has the form:

```
canvasname itemconfigure tag -option1 valuel [-option2 value2 ... -optionN valueN]
```

In Tk Example 4 all items with the 'road' tag are given a width of 8 pixels and a yellow fill colour.

A second command shows that tags can be changed temporarily, adding the tag 'lake' to all items with the 'blue' tag. Temporary tags are appropriate for transient labelling or symbolism. Finally four item bindings are demonstrated. They have a similar form to the widget bind commands:

```
canvasname bind tag <Action> {Tcl script}
```

In Example 4 clicking items with the 'red' tag with the primary and secondary mouse buttons (usually the left and central mouse buttons, but the exact outcome will depend on local operating system settings).
system, mouse and driver configuration) will change the colour of items tagged ‘line’. The ‘water’ tag is bound to commands that colour all ‘water’ tagged items red whenever such an item is touched with the cursor and blue when the cursor leaves the item.

```
# -- Tk Example 4.
canvas .c -width 100 -height 100 -bg Orange
.pack .c
.c create oval 10 10 30 30 -outline Black -fill Red -tags "red oval"
c.c create line 10 40 10 60 20 60 30 70 40 60 -width 6 -fill green -tags "green line water"
c.create line 90 90 70 90 50 80 40 70 40 80 50 70 50 50 -width 4 -fill red -tags "red line road"
c.create poly 40 10 60 20 80 10 85 40 70 60 50 40 40 50 -fill #28C -tags "blue poly water"

# -- Configure all items with 'road' tag to specified width and fill options:
.c itemconfigure road -width 8 -fill Yellow
# -- Configure all items with 'blue' to also contain tag 'lake':
.c addtag lake withtag blue

# -- Bindings to define behaviour of items with 'red' tag when clicked:
.c bind red <Button-1> {.c itemconfigure line -fill SkyBlue}
c.c bind red <Button-2> {.c itemconfigure line -fill Red}

# -- Bindings to define behaviour of items with 'water' tag when touched:
.c bind water <Any-Enter> {.c itemconfigure water -fill Red}
c.c bind water <Any-Leave> {.c itemconfigure water -fill Blue}
```

Figure 2.4 Tk Example 4 - Dynamic graphics, transient symbolism and observer-related behaviour.

Tags are also used to link all items to a scale bar to a procedure that scales the symbol coordinates. The colour and width tags are unchanged by the procedure, which can be viewed and interacted with by inspecting appendix B1 on the accompanying CD.

### 2.4 Tcl/Tk for Mapping

As symbols are created with simple, high level commands, and symbolism is changed in a consistent way for all symbol characteristics, Tcl/Tk provides an appropriate syntax for cartographic visualization. Here, a series of basic examples are presented using a small and simplified data set containing integers for clarity and simplicity. Items can however be created from variable values and more complex structures such as arrays of lists that contain appropriately transformed spatial information. Additional Tcl commands for opening and closing files and managing data structures such as strings, lists and arrays, mean that real geographic information can be readily incorporated into Tcl/Tk maps.

#### 2.4.1 Tcl/Tk for static maps
These basic properties can provide an appropriate environment for cartographic visualization. The rest of this section is used to demonstrate some examples of how spatial data can be mapped for exploratory analysis. The data used for these examples stored in small arrays as lists of coordinates and individual data values. They depict the ten British regions using highly generalised coordinates transformed to fit on to a canvas on the screen. The emphasis is on the way in which the scripting language and interface widgets provide a model for dynamic cartography, not the numbers. Examples that incorporate larger and more impressive data sets will be presented in due course.

In these examples the data are stored in 4 arrays. Tcl contains powerful file handling and data manipulation capabilities for incorporating data sets for genuine applications. The arrays represent information for the regions numbered 1 to 10. The ‘poly’ array contains ten elements, each is a list of x and y coordinate pairs that comprise an enclosed polygon representing a region. Brunsdon and Charlton (1996) have outlined some of the advantages of modelling spatial data in list form. Centroids for each polygon are stored in two arrays, ‘x’ and ‘y’, which have ten single numerical elements. For example, the geometric data representing zone 10, the ‘South-West’ region, is represented by data structures created by the following Tcl commands:

```tcl
# -- Array containing list of bounding x and y coordinates in screen units:
set poly(10) {106 155 106 183 106 183 95 188 80 184 76 195 69 191 48 198 70 172 86 173 92 163 92 163 159 92 159 106 155}

# -- Array containing centroid in screen units:
set x(10) 85
set y(10) 180
```

The spatial data are scaled to fit the 200 by 200 pixel canvas used throughout the examples, but Tk does provide scaling, zooming and scrolling capabilities that are not demonstrated here. A simple procedure could be used to convert from real World to screen coordinates and return the screen value. The attribute data consists of twelve variables, scaled between 0 and 100. They are stored in an array of dimensions 13 by 11 that has the form:

```
data(zone-number,variable-number)
```

For example, the data value of variable 3 ‘Age15-44’ for zone 10, the ‘South-West’, is encapsulated in the array with:

```
set data(10,3) 39
```

The names of the appropriate region are stored in data(zone-number,0) and variable names in data(0,variable-number). So the data set for the South-West is recorded in Tcl/Tk with:

```
# -- Array containing data values for 12 attributes scaled to 0-100 range:
set data(10,0) South-West
set data(10,1) 42 ; set data(10,2) 38 ; set data(10,3) 39
```

---

Interactive Maps for Exploratory Spatial Data Analysis 43
And the title of variable number 3 is stored in an array with:

```tcl
# -- Example from array containing variable names:
set data(0,3) "Age15-44"
```

A list of zone numbers, 'zones', contains the numerical identifier of each of the included zones and is used to loop through each of the cases:

```tcl
set zones "1 2 3 4 5 6 7 8 9 10"
```

Static images of the dynamic maps produced by the example scripts are presented as figures in the following sections. The code provided in the text is essentially accurate, but stylised slightly in places. The full script that produces each map and the interactive maps that should be used in conjunction with this text are provided in appendix B2 of the accompanying CD, where the full data set is available for inspection.

**Map Example 1: Creating a Static Map View**

Static maps and statistical views are created by locating symbols on canvases in a way that reflects data values. In Map Example 1 a canvas is defined with the name '.view 1'. A loop is used to progress through each element of the list 'zones'. The 'foreach' command is designed for looping through elements of a list in turn. It takes three arguments: the first is a local variable that is set to a number of values in turn; the second is a list of values for the local variable; the third is a series of Tcl/Tk commands to effect each time the local variable changes:

```tcl
foreach variable list {Td Script}
```

For example, to produce the square of a list of values:

```tcl
foreach i "1 2 3 4" {puts "$i squared = \[expr $i*$i\]"}
> 1 squared = 1
> 2 squared = 4
> 3 squared = 9
> 4 squared = 16
```

The list can consist of a value held in a variable or be the result of a Tcl command. For example, to display the values contained in the series of lists in the 'polygon' array that provides a structure for the geometry of the spatial data used in these examples, the following use of the 'foreach' loop could be made:

```tcl
foreach zone $zones {puts "Polygon $zone: poly($zone)"
> Polygon 1: 60 65 52 78 57 40 55 21 63 20 65 3 71 8 89 4 72 29 85 25 104 30 89 54 94 58 84 64 101 68 101 68 87 87 87 76 91 69 87 64 89 59 87 69 65 64 59 60 65
```

Interactive Maps for Exploratory Spatial Data Analysis
In order to create a static map the variable ‘zone’ is set to each of the values identified in the list ‘zones’ in turn. Two commands are issued for each value of ‘zone’ to create a shaded polygon map. First a variable ‘intensity’ is calculated using the ‘expr’ command which returns the result of a mathematical expression. The variable comprises of two parts, the string ‘Grey’, and the result of the expression that subtracts the attribute value held in the ‘data’ array at the location specified by the zone number and variable 1. For example, a data value of 42, which is the value of variable 1 for zone 10, will result in ‘intensity’ being set to Grey58, which is one of 100 legitimate X-Windows grey shades. This scale of colours can be used to shade canvas items and so produce a virtually classless choropleth (Muller and Honsaker, 1976).

Secondly a polygon item is created, as demonstrated in the Tk Examples above, using the list of x and y coordinates contained in the array ‘poly’ in the element specified by the value of ‘zone’. The polygon is filled with the calculated ‘intensity’ and given a unique tag for future reference comprising of the letters ‘id’ followed by the zone number (for example id1). This tag can be returned by canvas commands and used to locate elements in the available data structures, forging the link between map and data. The map shown in Example 1 is the result of looping through the ten cases. Maps for far larger data sets can be developed from spatial data files by looping through an identical set of commands with a longer list of zone ids. The RGB colour definition capabilities mean that appropriate and alternative colour schemes can be applied to the...
symbols, either using the RGB colour model that is fundamental to the operation of the cathode ray tube and popular in ViSC, or by transforming to the HSV model (Foley & Van Dam, 1982) that is utilized more readily by cartographers (Bertin, 1983). This ability to specify precise colour definitions enables the schemes advocated by Brewer (1994) to be used.

**Map Example 2: Re-Expression**

Alternative map views can be created by using different or additional symbols. This is Shepherd's 're-expression' (Shepherd, 1995). In Map Example 2 a graduated circle map is produced using polygons and ovals. A button that changed the mapping from polygons to ovals would provide an example of 'dynamic re-expression' (Shepherd, 1995) and enable the map-reader to compare alternative versions of Tobler's (1979) second transformation. Statistical views can be created by plotting symbols on a canvas in attribute space where canvas coordinates are taken from the 'data' array, rather than the cartographic space of the 'polygon' array. Examples are provided later in this section.

![Figure 2.6 Map Example 2 - Re-expression in Tcl/Tk.](B2)

**2.4.2 Tcl/Tk for dynamic maps**

Dynamic properties of the language, such as binding cursor movement and events to symbols so that the commands that change maps can be linked to user behaviour, equip mapmakers with a series of new dynamic symbol characteristics. We have seen that these are particularly appropriate to visual thinking as views can be changed by the user to suit specific conditions, uses and questions.

**Map Examples 3 and 4: Adding Dynamism – Changing Map Symbolism**

Transient symbolism can be applied to items with specific tags from item bindings, from widget commands, or via the command line. This permits brushing, dynamic re-expression and dynamic
comparison and is the crux of dynamic cartography for visualization. In Map Example 3 a procedure is defined and called from the '-command' option of a button to perform the latter two of these. Procedures were introduced in section 2.3.1 and are defined with a standard Tcl 'proc' command of the form:

```
proc procedurename {arguments} {Tcl commands}
```

They are called by the procedure name followed by values for the necessary arguments. For example a procedure to add two values would be defined as follows:

```tcl
proc addvalues {v1 v2} {
    set answer [expr $v1 + $v2]
    puts "$v1 + $v2 = $answer"
}
```

And called with:

```
addvalues 10 4
> 10 + 4 = 14
```

Procedures provide an efficient way of repeating common tasks or recreating required view types. In Map Example 3 a 'foreach' loop creates twelve buttons named '.b1' to '.b12' with options that show the appropriate variable name, and call the procedure 'reconfig' with a single argument, a variable number from 1 to 12 specifying the variable number associated with the button.

The procedure accesses the global variables 'data' and 'zones'. A 'foreach' loop structure is used to set the local variable 'zone' to each zone identifier in the list 'zones' in turn. The temporary variable 'intensity' is calculated for each zone number, based on the value stored in the data array for the element specified by the variable number passed into the procedure. A canvas command is used to configure the item tagged with 'id$zone', so that it is filled with the value of 'intensity'. Variable number 1 is thus mapped with the command 'reconfig 1', which is called from button 1. A map that
responds to widgets and maps the requested information by changing the symbolism of smart map symbols results. This is Shepherd’s (1995) ‘dynamic comparison’.

In Example 4, both the polygon map and circle map are changed by a redefined version of the ‘reconfig’ procedure that calculates new radii and bounding coordinates for the graduated circles and changes the circle configuration in ‘.view2’ as well as that of the polygons in ‘.view1’.

![Figure 2.8 Map Example 4 - Transient symbolism in Td/Tk: Changing multiple views.](image)

**Map Examples 5 and 6: Observer Related Behaviour - Linking Symbolism Changes to the User**

Buttons are an appropriate way to invoke dynamic re-expression and dynamic comparison. Other observer related map behaviours, such as brushing and labeling, require direct interaction with symbols. This is achieved with item binding of the sort introduced in Tc Example 4 in section 2.3.2 above. Map Example 5 takes advantage of item binding to interrogate symbols for information. It uses transient symbolism and produces transient labels. Along with Map Example 6 it demonstrates that the GUI builder permits Shepherd’s (1995) brushing to be implemented with ease.

Tk bindings can have two forms as shown below, each of which are employed in Map Example 5.

The first is used to bind a series of Tc commands to items on a Tk canvas when some cursor/mouse action takes place. The items are identified by a specified tag. The second binds the cursor/mouse action to an entire canvas and any location within it, irrespective of items and tags.

The two can be thought of as ways of defining symbol behaviour and map behaviour:

```bash
canvasname bind tag <Action> {Td commands}
bind canvasname <Action> {Td commands}
```

Bindings provide users with access to the way in which Tk controls and records the interaction between the mouse, cursor and canvas. For example, Tk gives the tag ‘all’ to every item that is created on a canvas and automatically updates any item that the cursor is currently touching to contain the tag ‘current’. The following binding, which is included in Map Example 5, will thus cause any item that is clicked with the primary mouse button to be coloured in red.
Tk also records and provides access to the location of the cursor in canvas coordinates. This can be passed to a procedure as part of a binding with the special variables ‘%x’ and ‘%y’. They can be used to report locations, or create graphics, as they are in the following binding that calls the procedure ‘show_location’ with two arguments – the x and y location of the cursor on the current canvas. This binding can also be tested in Map Example 5, by clicking mouse button 1 twice anywhere on the canvas.

To interrogate symbols for attribute or locational information the identifier of a selected symbol has to be detected, and the relevant information retrieved from an appropriate data structure such as the array ‘data’ in these examples. The information can then be displayed with appropriate text or graphics. In Map Example 5 a label is defined with the ‘-textvariable’ option set to ‘info’ meaning that the value of the variable ‘info’ will always be displayed in the label. Two bindings are then used to define brushing behaviours that extract attribute information from symbols that the cursor enters.

The first of these binds all items to a series of commands whenever an item is entered using the ‘<Any-Enter>’ action that is constantly managed by Tk. The commands set the variable ‘atts’ to contain the result of a Tk canvas command that returns all of the tags of the ‘current’ item. This is an extremely useful way of returning a list containing identifiers and data for the tag and has the general format:

`canvasname gettags tag`

For example, the following command would set the variable ‘atts’ to contain the list of tags associated with the item being touched by the cursor, where the square [brackets] force the command that they contain to be issued in advance of other commands:
The zone identifier is then extracted by applying Tcl commands that operate on lists to the list of
tags returned to 'atts'. The result is an integer representing the zone, which is used to retrieve
information from the data array. The variable 'info' is then set to contain the name of the current
region, the name of the currently chosen variable and the value for that combination of region and
variable, all of which are extracted from the data array. The current zone is given an orange
outline and is raised to the front of the canvas with a canvas widget command that has the form
'canvasname raise tag'. This ensures that the current item is distinguishable from other items and
visible above them. A second binding states that whenever the cursor leaves an item, its outline is
returned to black.

More elaborate bindings of this sort can be used to add transient labels to the canvas, and to
ensure that like tagged symbols are highlighted in multiple views. Transient labels can be created
on demand with line and text items, and linking is achieved by highlighting similarly tagged items
in more than one canvas. Some possible bindings are illustrated by Example 6 in which the polygon
and circle maps are linked, and where more durable brushing is invoked by clicking symbols with
the mouse button.

Figure 2.10  Map Example 6 – User interaction with symbols: Brushing between views in Tcl/Tk. (B2)

2.5 Visual Variables and Simple Cartography

The item types and associated configuration options make traditional cartography eminently
feasible in Tcl/Tk. Indeed canvas items allow simple shapes and symbols to be created and the
various options that are available bear a considerable resemblance to Bertin’s (1968) visual
variables (figure 2.11). Most of these can be implemented with relative ease using language that is
clear, consistent and explicit and that makes cartographic sense. Only ‘focus’ is unobtainable due
to the discrete nature of the items available on the Tcl canvas. Software Example 1
‘VisualVariables’, shown here in figure 2.11 and found on the accompanying CD in appendix B3,
illustrates some of the visual variables and uses pseudo-code to demonstrate how compliant Tcl/Tk canvas item specification is with the kinds of symbolism identified as being useful by cartographers. The cartographic syntax used by the scripting language to achieve the static symbolism is clearly demonstrated in table form in figure 2.11. The software shown incorporates observer-related map behaviours that enable the symbolism of each example symbol to be varied interactively and includes code that specifies how these behaviours are achieved.

Figure 2.11 Visual variables - Definition and specification in Tcl/Tk

The Tcl/Tk source provides commands that permit user-defined symbolism with familiar visual variables. The variables are shown on the left, with code that demonstrates how values of the variables are specified in Tcl/Tk (centre) and interactive graphics that show the results (right).

The software shown here is provided in appendix B3 on the accompanying CD.

Dynamic properties have been added to the symbols in the software. They demonstrate a lessening of the distinction between symbol and widget that is typified by the ability to vary symbolism interactively both from the widgets and by moving the cursor over the symbols and interacting with them directly (see 'Location'). Note also that highlighting takes place between all windows. The spatial information or smartness that is inherent in GUI builders, implicit in the symbols, used constantly by the GUI manager and so available when using the GUI approach to mapping, is also demonstrated. The spatial location of the cursor has been used to calculate the distance from each of the '?’ symbols produced on the canvas representing 'Focus'. The symbols are shaded depending upon the distance of the cursor from the symbol becoming more apparent.
as the cursor moves closer. When the canvas is clicked a canvas command is used to identify the closest item to the current location. The command is implicit to the Tk canvas and has the form:

```python
canvasname addtag tag closest x y
```

By passing the `%x` and `%y` variables to a procedure that adds a tag to the closest item to that location and then highlights the item with that tag in Red the appropriate item is identified or geographically brushed.

### 2.6 Testing the Approach

These programmable cartographic behaviours provide plenty of scope for the kind of open environment for experimentation that DiBlase and others (1994) recommend. They mean that researchers can control how, why and when maps are dynamic and extend the representation possibilities for cartographic visualization. Td variable substitution, file reading and control structures mean that scripts that read spatial data formats and produce maps containing large numbers of data-specified symbols can be developed rapidly using the techniques that have been outlined here with extremely simplistic data sets.

Another persuasive reason for using Td/Tk is its openness and flexibility. The interpreted nature of the language, and the open shell and command line mean that algorithms and graphics can be varied or tweaked to suit individual needs, questions and data interactively. Such a method of analysis has been advocated by Brunsdon and Charlton (1996) and MacEachren (1995) who observes that visualization tools should permit the interactive selection of different representations, each of which might enhance a particular aspect of the data. The scripted nature of the code means that users can vary representations rapidly. The costs of this flexibility, in terms of processing speed, are becoming less and less significant as computing technology advances.

Whilst DiBiase and others (1994) recognise that it is difficult to evaluate the success of exploratory methods, this kind of open, flexible environment for cartography seems highly appropriate and promising. In order to test the suitability of the approach a piece of prototype software was developed that used the techniques outlined above and applied them to the visualization of an enumerated data set with standard geometry and attribute information formats, taking into account the factors discussed in chapter 1.

The software was designed to allow users to select from a series of logical representational choices as advocated by MacEachren (1995) and to support a number of forms of brushing, novel views and dynamic comparison as well as providing an amount of freedom over how the data are dynamically re-expressed. The software synthesises techniques for information representation from graphical statistics and cartography such as those identified in the introduction, whilst employing
principles of graphic design to address issues concerned with visualizing enumerated information. The views are all produced on a two-dimensional plane using the techniques introduced above and use symbolism adopted from the cartographic literature. The software is called 'cdv', the cartographic data visualizer and is available to and used by the academic community as freeware. It provides the most complete application of the dynamic mapping techniques outlined here and is the embodiment of the techniques and approach under consideration.
Interactive Maps for Exploratory Spatial Data Analysis: Cartographic Visualization Approach, Implementation and Application

3. Dynamic Mapping for Visual Thinking: A Case Study

Jason Dykes
Department of Geography
University of Leicester
UK
3. Dynamic Mapping for Visual Thinking: A Case Study

The work presented in this section demonstrates the GeoGUI approach introduced in the previous chapter. It is an embodiment of the Td/Tk for mapping methodology applied to address some of the research issues identified in chapter 1, using multivariate enumerated data sets as the 'recognised geographic information'. The prototype was developed to implement the ideas on cartographic visualization outlined above, and to assess the feasibility of producing an interface to a spatial data set that incorporated useful elements of dynamic map behaviour and novel views for exploratory data analysis using the scripting language. This is an essential step as in the words of Unwin and Unwin (1998, p.418), "New ideas must be implemented so that they can be tested thoroughly. Elegant results in theory can often melt away in the heat of practical application". This volume relies upon proof and application of concept to substantiate the ideas presented throughout, through the production of usable software with accounts of its functionality and utilisation. This chapter outlines a prototype software system designed to implement and test the described methodology. The ensuing chapters contain a series of techniques developed from those introduced here that use this methodology in practical applications.

The application that implements this flexible approach to mapping for exploratory spatial analysis is 'cdv for Enumerated Data'. It is a highly interactive piece of software that has been developed using the Td/Tk GUI builder as a teaching and research tool. It demonstrates some of the dynamic cartographic techniques that might be useful when exploring multivariate, ratio scaled, enumerated data sets in line with the research priorities and methodology outlined above. 'cdv' reads ASCII files containing the geometry and attributes representing enumerated spatial information and applies interactive techniques for visualization of these data. It has evolved over a number of years into a resource currently consisting of 421 procedures for visualizing spatial information. These comprise of more than 22,000 lines of code in total. The 'cdv' software is provided in appendix B4, as a Microsoft Windows™ executable and additionally, in appendix C3, as a series of documented scripts and procedures that demonstrate the implementation, using the techniques introduced above. This section is thus effectively an initial results chapter, proving that the approach works and is suitable for the task in hand. The functionality of the software is introduced with examples (after Dykes, 1998). The chapter does not outline the full functionality of the software, nor does it provide great detail on how the scripts code the interactivity and symbolism. In order to assess the utility of this highly interactive approach, the reader is directed to the software provided in appendix B4 on the accompanying CD. For full details of the implementation the reader should review the documented scripts in appendix C3 which, due to the high level nature of the scripting language, form an explicit and relatively formal account of the implementation using the commands introduced in the previous section as their basis. In order to sample the full functionality of the software, and to use it effectively the reader is directed to the "Guide to Using
3.1 Example Data

Two data sets are used here by way of example. Both are included in appendix B2 on the CD and reside in the public domain. One is derived from the 1991 UK Census of Population and represents spatially referenced socio-economic details of the population of Leicestershire, UK. These data are collected at household level and agglomerated into a variety of spatial units for distribution. Those used here are at a level of unit known as 'wards' of which there are 187 in Leicestershire. Two types of data are present, the geometry specifying the outline of the zones projected onto some coordinate system (here the UK National Grid is used) and the counts that represent the statistical values collected for each unit. The second consists of spatially referenced socio-economic information collected in the 1990 US Census of Population. The data cover an area comprising of five states, Illinois, Indiana, Michigan, Ohio and Illinois, collectively known as the 'East North Central Midwest' by the US Census Office. They comprise of 437 cases representing the counties of these states each of which has a generalized polygon boundary and a series of 27 numeric values quantifying chosen socio-economic attributes of the population as recorded by the census (Stynes, 1996). The data were put together as part of the 'ESDA with LISA' meeting at Leicester in 1996 at which a number of software environments were used to analyse a single data set (Unwin & Unwin, 1998). In combination they represent the censuses of the UK and US relatively typically and so techniques that are applicable for these data should be of relatively general utility.

3.2 Modes of Software Operation

The higher level nature of the language means that any script written to conform to the higher level platform independent commands will run on a number of operating systems. Scripts written in Tcl/Tk can be run as a 'batch' file or from the command line by 'sourcing' script files containing a series of commands. The first of these options runs as an independent application over which the user has little control – relying on that provided to them by the functionality programmed in to the interface. Although this option can incorporate a large number of Tcl scripts (126 in the case of 'cdv', containing the 421 procedures) it has the 'self-contained' feel of an executable application to the user. It is suitable for demonstration purposes or for teaching. In fact, single executable programs can be developed from Tcl/Tk scripts using the Tcl Wrapper, part of the commercially available 'TclPro' package (Scriptics Corporation, 1999), which will compile standard scripts in to executables that run on a number of UNIX, Linux and Windows operating systems. The second means of running Tcl scripts is to run the 'wish' widget shell application and enter the names of the
script files in the Tcl console. Figure 3.1 shows the two windows that are associated with the ‘wish’
application, the Tcl/Tk Console that parses and interprets Tcl/Tk commands and the Tk GUI
window that displays any graphical consequences. This provides a ‘hands-on’ interface to the
dynamic graphic and data structures and is the recommended mode for exploratory spatial data
analysis as the views and procedures that create them can be manipulated directly from the
command line. This interface, which supplies all of the Tcl/Tk functionality, including the graphical
capabilities outlined in the previous chapter, results in an enormous degree of flexibility and
freedom. Researchers can ‘source’ a file containing scripts that define procedures from their own
archive or that of others, edit the file to change the functionality, re-source the modified script and
assess the results in the Tk GUI window. There is an evident parallel here with the degrees of
flexibility and interactivity required in cartographic products for visualization map use (MacEachren,
1994). Despite both modes demonstrating high levels of interactivity in comparison with a static
paper map, there are situations when these different levels of user control over map behaviour are
suitable. The more interactive and flexible command line mode is suitable for research whilst the
packaged application version that is constrained by the interface is more suitable for
communication, demonstration and teaching. These two forms of map use occupy different
positions on the ‘levels of interaction’ axis of the [Cartography]3 (MacEachren, 1994) shown in
figure 1.5. Both provide relatively high levels of interaction, but the command line mode of
operation offers greater interactivity and so occupies a position to the right of the GUI on the axis
shown in figure 1.5 that categorises ‘visualization’ map use.

![Figure 3.1: Running Tcl/Tk with the 'wish' Application.](image)

Left - A console is provided with a command line through Tcl/Tk commands can be issued.
Right - The graphical results of any Tk commands are displayed in the Tk GUI window.

### 3.3 Data Views and Software Control

However it is run, the software detects appropriate data files in a specified directory and produces
a GUI containing buttons from which each of the available data sets may be selected. Additionally
a button labelled 'Select Data' is available that produces a data selection widget. This consists of a
data selector with four sets of widgets, each of which can be used to choose and view one of the
four types of data file that 'cdv' reads. The buttons on the left hand side of the data selector
produce a file selection tool that is consistent with the operating system on which the software is
running and restricts selection to a specific file type. When such a file is selected the name and
locations are displayed. Clicking the '?' button to the right of the file name will provide details
about the data format, whilst clicking the button showing the eye from the plumage of a peacock
will display the selected data file (see figures 3.2 – 3.5).

All spatial data files are in ASCII format and all lines in data files are regarded as Td/Tk lists. These
are space separated ASCII words, with "quotes" to delimit words that contain spaces by
convention, in the manner outlined in the previous chapter. Files containing the four types of data
should be named with the same body, relating to the name of the location that they represent,
with a file extension describing the file type. Thus the software takes a recognised geographical
data set as its starting point and manages iterative transformations between software, maps and
the user. This is a realistic point from which to assess visualization techniques and make a
contribution. The software loads data stored in this model and converts them into data structures
from which interactive maps can be produced that are appropriate for visualization. The file types
and formats are specified as follows.

3.3.1 Spatial geometry data

Geometry for enumerated zones is stored in a file ending with the '.arc' extension that contains
sets of enclosed polygons in real world coordinates. A polygon definition has three components. It
begins with a three-word line of which the first word is an alphanumeric string that identifies the
enumeration unit. This is known as the 'id'. The second and third words represent an easting and
northing associated with that unit, known as the centroid. This coordinate pair is used to locate
point symbols that represent the unit in geographic space in some views. The second part of the
polygon definition is a series of easting and northing pairs that are recorded as two-word lines with
the easting preceding the northing and any number of space or tab characters separating them.
These can be in clockwise or counter-clockwise order, but must be successive. If the polygon is not
closed the software will add the initial coordinate pair to the end of the list of coordinates. The final
part of the specification is in the form of a line containing a single word, 'END'. This specifies the
end of a polygon definition. The Td/Tk script reads through the data file line by line. Whenever a
three-word line is encountered a new polygon is assumed to have been delimited despite the
absence of an 'END' statement. Any lines with zero, or more than three words are ignored, as are
lines of any length whose first word has a '#' hash symbol as the initial character. Lines in Td/Tk
scripts that start with a '#' hash are ignored as comments, and this feature is replicated in spatial
data files meaning that scope exists for adding headers, comments and other metadata. The
format is readily achievable from a number of GIS packages and is a more general case of the Arc/Info 'ungenerate' geometry format. Clicking the 'peacocks eye' button on the Data Selection GUI will ask the user if they would like to display the geometry of the selected data file for inspection. Select 'No' to see an ASCII view, which demonstrates the geometry file format by loading the data read from the selected file into a Tk text widget, as shown in Figure 3.2. This is the spatial element of the 'recognised geographical information' that is the result of Tobler's (1979) first transformation of the real world which in the case of the UK census of population is used to represent millions of people, their behaviour and social habits, every 10 years.

Figure 3.2 'cdv' Geometry File Format: ASCII view shown in the 'Polygon Boundaries Quick View' through the Data Selection GUI.

As the data file is read the information is stored in a series of data structures for visualization. Each identifier is appended to a list containing all identifiers and each polygon representing a unit is appended to the list of all polygons representing that unit. Typically most zones are depicted by a single polygon. Multiple polygons that represent a single enumeration unit are simply given the same identifier in the data file. The list is stored in an array element referenced by the polygon identifier. Each pair of coordinates for a polygon is appended to a list of coordinates that specify the polygon, stored in an array element referenced by the identifier and the polygon number. This forms a data structure from which individual polygons can be extracted by specifying a zone identifier, which returns a list of polygons that represent the zone. Specifying the identifier with an element from this list will return the set of coordinates for the polygon. The list of coordinates can then be used as an argument to the Tcl canvas command to create a suitable canvas item. Each item is given a tag containing the zone identifier when it is created, so that the polygon can be recognised and associated with subsequently loaded attribute information. The process is described explicitly by the 'read_spatial' procedure provided in appendices A3 and C3. To view the polygons using the techniques described in chapter 2, the coordinates must be transformed to screen coordinates. A check in the polygon reading procedure records the bounding box of the data set. A transformation procedure then takes a coordinate and subtracts the minimum northing.
or easting, divides by the maximum range in north/south or east/west and multiplies by a screen scaling factor. As the origin of the Td/Tk canvas is to the north-west transformed northing are subtracted from the height of the canvas to flip the symbols about the x axis.

Figure 3.3  'cdv' Geometry File Format: Geometry view shown in the 'Polygon Boundaries Quick View' through the Data Selection GUI.

The spatial data file can be viewed geometrically in 'cdv' by clicking the 'peacocks eye' button on the Data Selection GUI once again and selecting 'Yes'. A Tk canvas is created and a number of procedures are called that issue commands to perform the tasks outlined above. This is documented in Td/Tk by the 'draw_polys' procedure provided in appendices A3 and C2. A rudimentary outline map displaying the spatial elements of the geographic information is produced (see figure 3.3).

Figure 3.4  Observer-Related Behaviour: Brushing for Polygon Ids in the 'Polygon Boundaries Quick View'. Any polygon touched with the cursor is temporarily highlighted. The polygon ID is extracted from the symbol and shown in the label above the map.
This basic view of a geometric data set contains some 'observer-related' map behaviour, falling into the 'brushing' category (Shepherd, 1995). Binding commands have been used to associate events whereby the cursor enters and leaves polygons with Tcl/Tk commands that produce transient symbolism in the way demonstrated in the previous chapter. The commands used here reconfigure the polygon so that it is highlighted when touched. Additionally the tags associated with an item are acquired whenever a polygon is entered by the cursor. These are displayed in a label at the top of the screen so that the identifier of any item is revealed by interrogating the polygon with the mouse and cursor (see figure 3.4).

### 3.3.2 Multivariate attribute data

Statistical data for enumerated units are stored in a file consisting of rows representing spatial units and columns representing attributes. As with the geometry data file the lines of the attribute file are read as Tcl/Tk lists consisting of space and/or tab separated words, with words that contain spaces delimited by "quotes". The first two words are essential. They record the unit identifier and a name for that unit. Following words represent data values for ordered attributes. 'cdv' has been designed initially to demonstrate techniques for visualizing interval and ratio scaled data. Categorical integer values will be treated as though they are ratios by the software. Unwin et al. (1996) have identified the need to consider and represent missing values in attribute data sets. In 'cdv' any non-numeric values (such as the word "NONE") are considered as 'missings' and these will be mapped and treated accordingly.

![Figure 3.5 'cdv' Attribute File Format shown in the Data Selection GUI.](image)

The 'Attribute Table Quick View' shows the data file format (left). Observer-related behaviours are incorporated to permit brushing that highlights associated information in all data views (right).

Having selected an attribute file (identified by the '.cdv' extension) clicking the 'peacocks eye' button in the Data Selection GUI will read in the data file a row at a time. The first word of each row is recorded and appended to a list of all attribute data identifiers. The second word is stored in an array of names that are referenced by identifier and thus form a look-up table. All other words
are stored in a two-dimensional data array referenced by the identifier and the column number from 1 to n. The values are displayed in a Td/Tk text widget (see figure 3.5). Each piece of text in the widget is tagged with the row identifier and the column number in much the same way as a canvas item is ascribed tags.

When all of the data have been read a number of processes take place. Each of the data columns from 1-n are considered in turn and the full list of identifiers used to find the maximum and minimum values and hence the ranges. A new value is calculated and stored in the data array by scaling the data values to a range of 0-100 (see the 'read_attributes' and 'summary_stats' procedures in appendices A3 and C3). The columns in the text widget are adjusted to fit a grid. A number of forms of brushing become available between the map view of the geometric data set and this attribute view.

For example, moving the cursor over any polygon in the map view will issue a command that highlights any row in the text widget that has the same tag as the interrogated polygon. Moving the cursor over any row name in the attribute view will reveal any polygons with that identifier in the map view. Additionally some symbolism is available. The attribute column numbers are displayed at the top of the columns in the attribute view. Moving the cursor over any of these highlights the column. Clicking the column number results in a procedure that loops through the list of attribute ids and extracts and scales the appropriate value for each zone between 0 and 100. A canvas command is then issued to configure any polygon tagged with the attribute identifier to a grey scale based upon this value using the range of intensities named Grey0 to Grey100 available in Td/Tk. In this case the highest value would be represented by 'Grey100', which is 100% intensity, equating to white. So to produce a map with dark shades representing high values the appropriate shade is actually 'Grey[expr 100-$value']'. This process is an extremely quick one, and results in interactive brushing that permits dynamic comparison and gives a first indication of the spatial distribution of the attributes and the quality and validity of the data set.

### 3.3.3 Attribute data keys

The attribute key file is simply an ASCII metadata file that contains names for the attribute columns. It consists of a row for each attribute and two columns: the number of the attribute data column from 1-n, and a name for the variable represented by that column. Once again words are separated by spaces and or tabs, and words containing spaces and/or tabs are delimited with "quotes". Attribute key files are identified by the '.att' file extension and can be selected and viewed as outlined previously. The name of each attribute is shown in the widget that is produced when the file is viewed. Brushing links the widget with the columns of the attribute data widget and vice versa.
### 3.3.4 Associating geometric and attribute data

The geometry/attribute link file contains rows of data representing enumerated units. It is often the case that the polygon identifiers and attribute data identifiers do not match. For example, if geometric data are obtained from a GIS such as Arc/Info polygons are systematically assigned integer identifiers when data are exported, irrespective of the internal polygon identifier that relates to the spatial unit. Each row of the link file is read, and the first two words in a row are used to associate the two identifiers that relate to the same enumeration unit. The first word should contain a valid identifier from the spatial geometry file and the second a valid identifier from the first column of the attribute data file. All other words are ignored.

A geometry/attribute link file can be selected in the same way as the other file types and viewed by clicking the 'peacocks eye' button. Brushing links each row of the file with the polygons and attributes determined by the values present in the row of the link file. In addition the attribute and polygon views are linked to the view of the geometry/attribute file that determines which attributes are mapped in and represented by which polygons.

### 3.3.5 Selecting a data set for visualization

Once a data set has been selected, viewed and checked in this way a 'cdv' visualization session can be commenced by clicking the button labelled 'Start cdv with Selected Data'. The following section outlines some of the visualization functionality contained in the software. It is recommended that the reader refers to the software provided in appendix B4 and runs the 'cdv' executable under Microsoft Windows™ by double clicking the icon representing the 'cdv' application. Click the 'Leicestershire' button to use the UK census sample for Leicestershire whilst following the subsequent text in this chapter.

### 3.4 Basic Visualization Functionality

A series of requirements for interactive graphics software for EDA are outlined by Unwin (1999). These include graphics that contain observer-related behaviours for direct querying, zooming, re-scaling, selection with linking. Each of these needs to be achievable through multiple linked views of a data set and data subsets. The functionality must be fully integrated into a software 'system' with a consistent interface.

'cdv' is designed to achieve these functions in a cartographic context and do much more than the traditional map role of presenting an initial view of a digital representation of enumerated information. In order to undertake the kind of visualization map use identified in previous chapters
n a number of univariate and multivariate views are required that exhibit observer related behaviours permitting high levels of interaction and user control. In 'cdv' two windows appear initially when a data set has been selected. These are the view control panel and the base map. The view control panel has familiar GUI widgets such as buttons, list boxes and scale bars. It presents a variety of ways of representing the data. The base map looks like a standard choropleth, showing the polygons stored in the geometry file, but is an interactive GeoGUI designed for visualization (figure 3.6). These interface elements allow a number of variables from the loaded spatial and attribute data to be selected and viewed in a variety of ways. Both of these windows must remain open throughout a 'cdv' session.

![Figure 3.6](image.png)

The geographical distribution of each variable is shown in the 'base map' by colouring each polygon in an intensity scale ranging from 100% for the minimum value to 0% for the maximum whenever a variable is selected from the attribute list on the view control panel. When the software loads initially the first column in the attribute data table is selected and displayed. This is the area of each zone in figure 3.6.

The statistical distribution of single variables can be represented using symbols located along a single axis. Each case or enumeration unit is represented by an identical symbol located along the axis so that the distance from the origin to the position of the symbol is equivalent to the relative statistical value for the chosen variable for that zone. Low values are displayed to the left of the
axis and high ones to the right. The view is known as a dot-plot. A box-plot is added along the axis to show the median and quartiles (figure 3.7). It adheres to some of Tukey's (1977) principles but equally aims to conform to Tufte's (1983) notion of maximising the data:ink ratio. Interactivity is provided by a series of widgets through which the representation can be re-expressed by changing cartographic parameters. For example, the view can be redrawn at a different size if the appropriate scale bar on the ‘Sizes’ window is moved (click the ‘Sizes’ button on the view control panel to reveal this). Widgets on the dot-plot window allow symbols to be resized, made transparent and raised or lowered above or below other symbols to investigate the depth of cases at any location.

Figure 3.7  A dot-plot showing values of enumeration units using location as the visual variable.

The figures represent the percentage of those living in wards in Leicestershire in 1991 aged between 0 and 15 years. A box-plot on the axis displays the position of the median and quartile values. The interface widgets manage observer-related behaviours.

3.4.1 Dynamic comparison

The view control panel contains a list of the attributes that have been loaded from the spatial data files. It enables the map user to select up to three variables concurrently for analysis, these are referred to as ‘v1’, ‘v2’ and ‘v3’. A variable is selected as ‘v1’ by double-clicking the variable name with the primary mouse button. The list is scrollable in both horizontal and vertical directions. Whenever a variable name is selected as ‘v1’ the polygons in the base map will be shaded in appropriate intensities of grey to show the spatial distribution of the chosen attribute. Zones with no data value for the selected attribute are shown with a mottled texture. Two electoral wards in the east of Leicestershire have polygon boundaries but no attribute values as the numbers of individuals enumerated in these zones from the small area statistics 10% sample from which the data are derived are small enough for privacy to be a concern. The data are thus withheld. In ‘cdv’ polygons that do not have associated data values are symbolised with a dotted pattern (see figure 3.6). Attributes can be selected as ‘v1’, ‘v2’ or ‘v3’ by clicking on the variable name once to highlight it and then selecting the ‘v1’, ‘v2’ or ‘v3’ button. To switch a selected attribute from ‘v2’ to ‘v1’ click the ‘v1-v2’ switch button and note that the base map shows the distribution of the newly selected ‘v1’ (figure 3.8).

Dynamic comparison of successive attributes in the data file can thus be achieved by clicking variable names and using the switch buttons. Additionally the ‘Scales’ button replaces the list box
interface to the attribute files with a series of scale bars, one for each selected variable. These can be moved to select attributes from the linear list of data columns. Data that represent a time or data series might usefully implement dynamic comparison through this form of interface. Clicking the same button again will revert to the list box variable selection interface.

Figure 3.8 Dynamic comparison using the variable selection list box on the view control panel. (C1)

Selecting successive variables as 'v1' shows successive distributions. This is achieved by double-clicking variable names in the list-box or sliding the variable scale bar. Here the maps show the proportions of the populations of the wards in Leicestershire with the following criteria: those suffering from long-term illness; those living in overcrowded conditions; those who are unemployed. The final image illustrates observer-related behaviour as the map is brushed for information about the values of the three variables.

3.4.2 Brushing to highlight

Each view can be interrogated for details including the names of the enumeration units represented by the polygons and data values for the three selected variables. This is the 'transient labelling' identified by Becker et al. (1987). This information is obtained by moving the cursor over any symbols on the base map. The symbol changes colour and the information is returned from the symbol and displayed at the top of the view control panel. The speed at which these changes occur and this information is related to the user demonstrate the highly interactive nature of the software and the use of transient symbolism for visualization (figure 3.8). By combining this interrogative functionality with user controlled dynamic-comparison 'cdv' implements the dynamic querying identified as an important future feature of visualization tools for enumerated information by Egbert and Slocum (1992).

3.4.3 Dynamic re-expression

The interactive nature of the software is also demonstrated by the classification functions of the dot-plot which allow views to be classified interactively in real time in a manner that imitates the density dial (Ferreira & Wiggins, 1990). GUI Buttons on the dot-plot window provide common classification techniques, but users can slide the arrows that show class breaks on the box-plot to...
fit the distribution and highlight a variety of patterns in a manner that they choose. New class breaks can be created by double clicking the secondary mouse button in the dot-plot window and dragged with the primary mouse button. The sensitivity of any classification can thus be assessed. Clicking on the ‘Update’ button ensures that the classification occurs in real time. The first map in figure 3.9 shows a four-class ‘Equal Size’ scheme revealing the spatial distribution of the four quartiles. The subsequent maps show a two-class scheme and demonstrate the effects of dragging class boundaries interactively. All colour schemes used can be selected by the user with a RGB / CMY / HSB colour selector interface by clicking the ‘Colours’ and ‘Chooser’ buttons on the view control panel. This has cartographic significance, meaning that Bertin’s (1983) visual variables can be used to symbolise variations in attribute, and that a suitable scheme for the data type under consideration can be adopted (Brewer, 1994).

Figure 3.9 Dynamic re-expression: Interactive classification of a choropleth.

A four-class scheme with groups of equal numbers of cases shows the percentage of those living in wards in Leicestershire in 1991 aged between 0 and 15 years (left). This series of images demonstrates how interactive classification can be used to assess the stability of classified patterns. The three other images show the selection of a two-class scheme to map the same variable. The class limit is dragged interactively along the statistical distribution. The map is updated to show the effects on the spatial distribution that results from the classification in real time.

3.4.4 Observer motion

This form of observer-related behaviour relates to the zooming and panning functionality in dynamic maps. Tcl/Tk contains a canvas command that scales all items with a particular tag about an origin by a specified factor in the x and y dimensions. The command has the form:

```plaintext
canvasname scale tag xorigin yorigin xfactor yfactor
```

This enables zooming which is fully implemented in ‘cdv’ by dragging a ‘rubber box’ across a map with the secondary mouse button to display a region to zoom in to. This region is selected by moving the cursor over and away from the rubber box outline. ‘cdv’ then takes advantage of the links between canvas widgets and associated scroll bars in Tcl/Tk to provide scroll bars that automatically pan about the canvas (figure 3.10). The location of the region that is displayed in the

**Figure 3.9** Dynamic re-expression: Interactive classification of a choropleth. (C1)

A four-class scheme with groups of equal numbers of cases shows the percentage of those living in wards in Leicestershire in 1991 aged between 0 and 15 years (left). This series of images demonstrates how interactive classification can be used to assess the stability of classified patterns. The three other images show the selection of a two-class scheme to map the same variable. The class limit is dragged interactively along the statistical distribution. The map is updated to show the effects on the spatial distribution that results from the classification in real time.
base map is shown on the icon map at the top of the control panel that represents the full data set.

Figure 3.10 Observer motion: Zoom and pan functionality in 'cdv'.

The first image shows a bounding box being selected using the mouse on the base map. When the cursor moves away from this box the view zooms to the selected region (second image). The scroll bars beside the image can then be used to perform panning across the geographic space (third image) which is represented in the icon map on the view control panel (fourth image).

3.4.5 Brushing for linking

All views in 'cdv' contain a checkbutton at the bottom right corner that dictates whether they are dynamically linked to the base map. When selected a binding issues a command to link all symbols in that view so that corresponding symbols are highlighted in all other selected views whenever a symbol is brushed. The binding returns the identifier of any smart symbol that is touched with the cursor and shows the name of the associated location in the view control panel as before. Additionally it loops through a list of all selected views and applies transient symbolism to highlight any similarly tagged symbol in the view. Thus a spatial component is added to views which use location to show statistical information about spatial data integrating the graphics that represent geographic and attribute spaces. Clicking the symbol highlights it in the current highlight colour in the selected views. This is an implementation of 'transient highlighting' (Becker et al., 1987).

The brushing function allows subsets of the data to be highlighted. Any number of subsets can be selected and viewed concurrently. Each subset can be assigned a different highlight colour so that the links between the spatial and statistical distributions, and different variables, can be investigated as and when desired by the user. Figure 3.11 shows two groups of values selected from the upper end of the statistical range in different intensities of a red hue. These are displayed in the base map.
Figure 3.11  Linking views: Statistical groups displayed in a geographic view.

A selection box in the statistical view can be created interactively and used to choose cases in one view and highlight them in all others. Here two groups have been selected in the statistical view and highlighted in the geographic view.

3.4.6  Novel spatial views

Novel spatial views can be created using the forms of symbolism and interface provided by Tcl/Tk and the interactive nature of the GeoGUI can be used to overcome some of the limitations of static maps and resolve some of the trade-offs required to map the type of complex and multivariate spatial information associated with enumerated data on the plane.

The ‘circle map’ view in ‘cdv’ uses the size visual variable to display attribute information. Graduated circles represent data values at zone centroids. Robinson et al. (1995) identify that ‘size’ is an appropriate and ‘more visually effective’ form of symbolism than colour for counts such as population totals. In ‘cdv’ the map user can experiment with size, value and combinations of the colour variables in order to elicit, identify, confirm or reject a spatial pattern. The view has a number of dynamic features, including the brushing functionality described above. Additionally, circle maps contain a number of additional widgets through which the cartography applied to the view can be varied and the information dynamically re-expressed. A scale bar allows the scale factor applied to the symbols to be varied, the ‘Clear’ button makes symbols transparent so that overlaps and symbol density can be detected, and the ‘Re-order’ button changes the layering of symbols through dynamic re-expression. The default is for symbols to be layered from largest to smallest so that all are visible. ‘Re-ordering’ maps a series of random symbol orders so that the
sizes of symbols can be observed in turn. The outlines of zone boundaries can be switched on and off to suit. In order to view all symbols concurrently, graduated circles can be dragged across the map distorting the spatial distribution. Despite the loss of precise geographic information, three forms of brushing aid the user in relating moved symbols to locations. An arrow links the symbol with the original location, when a symbol is touched with the cursor the original zone boundary is highlighted, and symbols move concurrently in all selected circle maps whenever a symbol is dragged to a new location. The linked examples shown in figure 3.12 represent the total numbers of cars in the county of Leicestershire.

![Figure 3.12 Dynamic graduated circle maps: Numbers of cars owned in wards of Leicester, 1991.](image)

The locations of the symbols can be moved interactively to overcome the graphical congestion. Links to geographic locations are provided. The locations of all related symbols in selected circle maps are updated accordingly.

An additional means of overcoming the issues concerned with overlapping symbols that is becoming increasingly popular uses Dorling's cartogram algorithm (Dorling, 1993). The program runs an iterative procedure that moves overlapping symbols apart while drawing neighbouring cases together. In doing so it attempts to retain the topological structure of the geographic space (see section 1.2.2). The centroids and radii that define a non-continuous cartogram can be loaded into 'cdv' from standard ASCII data files. The expected data format involves a file of rows representing enumerated units, each of which contains at least four words. The first four space delimited words are expected to contain the attribute identifier associated with a case, the x and y coordinates of the cartogram centroid, and the radius of the symbol to be produced. Here population cartograms have been created for the wards of the county of Leicestershire in the UK, and the counties of the East North Central Midwest in the US. They are displayed in 'cdv' in figure 3.13.
The zones used in the UK and US differ enormously in their delimitation. This highlights the way in which different objectives mean that the first cartographic transformation, from the word to a recognised data set (Tobler, 1979), can have an important effect on the second transformation, that of map-making, even if the data model used is identical. The UK enumeration zones are used for electoral purposes and so tend towards equalising the population totals within wards, or certain agglomerations of wards into higher order units such as districts, counties and regions. The range of population totals between wards is thus quite small, particularly within a geographic region such as a district or county, and so whilst the polygons used on a choropleth map vary enormously in area, the symbols used in a population cartogram are of roughly equivalent size. In the US counties are loosely based on land area and so the counties are of approximately equivalent areal magnitude. The size of any cartogram symbol is thus dependent upon population density which varies enormously, especially in an area like the US Midwest that contains huge numbers of people per unit area in the major cities and extremely under-populated areas dedicated to and dominated by agriculture. Indeed the population pressure is so great in Chicago that some of the worlds tallest buildings have been created at huge expense to increase the potential population density. A huge range of population totals across the counties results, from a minimum of under two thousand to a maximum of over five million resulting in a ratio of 2500:1 between the size of smallest and largest symbols.

Population cartograms are a recognised way of producing egalitarian maps (Dorling, 1996) although their interpretation is hindered by difficulties in relating cartogram symbols and the places that they represent. Their utility is much enhanced by dynamic linking techniques that provide an interactive means of associating the abstract topological space of the cartogram with familiar geographic space. This can be achieved either by the interrogation facility, where any symbol in any view shows the corresponding symbol in all other views along with the zone name, or by selecting a region in one view and displaying it on another. Corresponding cases are highlighted in

Figure 3.13 Non-continuous population cartograms created using Dorling's (1993) algorithm.
Left - Leicestershire, UK at ward level.
Right - The US North Central Mid-West at county level.
both views in figure 3.14, demonstrating that the conventional geographic representation under-
represents densely populated urban zones. Brushing in this way overcomes the oft-quoted
difficulties in relating cartogram symbols with the locations, and ultimately people, that they
represent, forging an important link back thorough the cartographic transformations to the
phenomenon of the real world under assessment.

Figure 3.14 shows that region selection can be accomplished in any view, and the selected zones
highlighted in all other included views. This is achieved by toggling the 'Zoom'/Select' radiobutton
at the top of the view control panel to 'Select' and using the primary mouse button to create a
brushing box. By holding down the 'Shift' key and clicking the edge of the box with the primary
mouse button symbols that overlap the box are highlighted along with those that represent the
same cases in other views. In figure 3.14 a geographic region is selected and the equivalent zones
shown in the abstract cartogram space. Then a region of the cartogram is chosen and displayed on
the polygon map.

Another example combines the classification and cartogram techniques. Classification schemes
implemented in a dot-plot can be applied to any selected view. Three views are shaded in figure
3.15 with two classes depending upon area of the electoral ward. Linking these views in this way
demonstrates that the largest 25% of wards in Leicestershire dominate the geographic space upon
which population data are regularly mapped, yet represent a small minority of the population. This
leads to a consideration of issues concerning the use and occupation of space by the populace –
the essence of Geography. By making the cartogram more usable in this way the interactive
cartography is providing the user with a representation that relates to the phenomenon that is
being measured, the characteristics of society in space, rather than the recognised geographical
information provided to the cartographer. Using information technology to facilitate the
transformation of information right through Tobler’s three transformations (Tobler, 1979) is a significant progression.

Figure 3.15 Linked views: Choropleth, cartogram and dot-plot.
The data show the geographic areas of electoral the wards of Leicestershire. A two-class scheme is used to classify wards at the median value.

3.5 Advanced Visualization Functionality

These basic views and observer-related map behaviours can be extended to permit analysis of subsets of the geographic or attribute space and the assessment of more than one variable concurrently.

3.5.1 Combining variables

Variables can be combined for analysis in ‘cdv’ in a number of ways. Functionality to create new variables by mathematical combination of loaded variables is provided through the ‘Calculate’ button on the view control panel. This provides an equation parser that uses the ‘expr’ command and the mathematical functions available in Tcl/Tk. Variables are identified in the parser by their numbers as displayed in the variable selection list box. The format is that a variable is specified with the character ‘v’, followed by the variable number in [square brackets]. So for example, the first variable in the list box would be identified by ‘v[1]’. A name is chosen for the new variable which can incorporate any valid Tcl/Tk mathematical function or expression (see Ousterhout, 1994; Critchlow, 1995). For example, with attributes that record the total number of people of working age and the total in work a ratio that related to the rate of unemployment could be produced. If these two variables were numbered one and ten respectively, and both contained integers representing counts, the following expression would be appropriate to produce a variable recording percentage unemployment (‘double’ converts the divisor to a double precision floating point value):

\[ \frac{(100 \times v[10])}{\text{double}(v[1])} \]
Newly created variables are saved for use in subsequent ‘cdv’ sessions.

A graphical means of combining variables involves symbolising univariate views, such as the dot-plot or the population cartogram to show a second variable. For example, a dot-plot showing the number of people in each zone, or a population cartogram, might be shaded by the percentage unemployment. Overlapping zones can cause problems here, and an alternative was scripted for the Midwest data set where the numbers of people represented by symbols on the plots varied so drastically. Symbol sizes rather than shades are used to symbolise population totals on the dot-plot in figure 3.16. This example showing % Below Poverty Level’ allows the distribution of numbers of people living below the poverty level to be assessed as well as the distribution of zones of enumeration. It illustrates the wide range of population totals in the data set and emphasises people as well as cases. Some of the most populous zones are within, or close to, the top quartile.

Pairs of attributes can be displayed using locations along orthogonal axes with the scatter-plot, a standard form of symbolism used by statisticians. In ‘cdv’ the scatter-plot has been extended to take advantage of a series of dynamic features such as those used for querying and linking in the univariate dot-plot (see figure 3.17).
The scatter-plot also has the capacity to vary symbol size and show overlapping symbol outlines, but not colours, so that symbol densities can be detected. In addition the 'Ratio' button toggles between axes which show values in relative and absolute terms and a 'Swap' button re-expresses the information by switching the axes. Box-plots are added to each axis so that the distribution of values can be assessed, and parallel box-plots on the Y-axis allow the two statistical distributions to be compared.

3.5.2 More advanced brushing for subset selection

By creating views of sub-sets of the data the whole range of graphic symbolism can be applied to a restricted data range and local variation can be assessed using the full extent of perceptible visual cues. Symbols or areas of interest can be selected within geographical or statistical views of the data by clicking symbols or enclosing them with a rectangular lasso. Subsequent views consider and display only the selected cases and the intensity ranges vary accordingly. Thus geographic regions and statistical clusters can be examined more closely and outliers can be removed from the analysis. This is the form of behaviour that Becker et al. (1987) termed 'focus' brushing.

'\% Poverty Status Determined' in the Midwest is mapped in figure 3.18 and the distribution shown for the full data set (left & top). A locality is then selected for further assessment, with the grey intensities and dot-plot locations of subsequent views restricted to the local data range (bottom right).

![Figure 3.18 Brushing for focus: Selecting data sub-sets from a geographic view.](image)

Cases selected from a geographic view (left) are displayed in a polygon map that only considers the cases from the selected locality (bottom right). The new map has been re-scaled using the 'cdv' zoom and pan functions. The data set shows '% Poverty Status Determined' for counties of the US Midwest. Dot-plots are provided that show both the full and focused data sets. The focused data set is shown by the rectangular selection lasso on the full polygon map (left) and the highlighted symbols on the dot plot that represents the full data set (top right).
Whenever a sub-set has been selected the icon map at the top of the control panel is highlighted. The full data set can be reselected with the 'All' button by the icon map. Groupings can equally be selected from statistical views. These can be achieved by toggling the 'Zoom'/Select radiobutton on the view control panel to 'Select' and using the secondary mouse button to create a selection box. The box can be clicked once with the primary mouse button to highlight the zones, twice whilst holding the 'Shift' key to see the cases in all selected views or twice in rapid succession to set the selection, or focus, to the symbols that overlap the box. Sub-selection is achieved in this way from a scatter-plot in figure 3.19 viewed in a polygon map that has been re-scaled using the cdv zoom and pan functions.

![Figure 3.19 Brushing for focus: Selecting data sub sets from a statistical view.](image)

Cases selected from a scatter plot are viewed in a polygon map that only considers the cases from the selected locality. The new map has been re-scaled using the 'cdv' zoom and pan functions. The data set shows '% Male Unemployment' and '% Female Unemployment' for wards in Leicestershire, 1991.

### 3.5.3 Multivariate analysis

Three variables can be assessed concurrently using a variety of techniques. A third variable dimension can be added to a bivariate scatter or dot-plot view by shading symbols with an additional attribute. Alternatively each of the three primary colours can be used to represent three variables and a proportion of each hue applied to every symbol dependent upon the magnitude of the data value for the associated spatial unit. A red/green/blue colour composite results (Ware & Beatty, 1988). Greys (in a range from black – low values, to white – high values) indicate low relative variation between the three variables, so a grey scale map would indicate three perfectly correlated variables (with dark values representing the lower end of the scale in this instance). Zones coloured in any of the three primary hues indicate extreme values in the variable represented by the hue, those coloured in secondary hues indicate extreme values in two of the
variables. Other colour combinations can be assessed using a colour key. Whilst the variation in the colours produced does not vary in a linear fashion and cannot be perceived as such, the composites provide a useful visual means of identifying alike zones and local outliers. The ability to link to dot-plots of the three variables, which do represent the information in a linear fashion using location, and can be compared, makes this representation particularly appropriate in the dynamic software medium. These can also be coloured with the composites to aid recognition. In 'cdv' the variables associated with each hue can be switched interactively by clicking the 'RGB' checkbutton on and using the selected variable switch buttons (labelled '1<->2' etc.) to change the variables that are chosen as v1, v2, and v3, meaning that each of the nine possible mappings of variable onto hue can be assessed. Additional checkbuttons provide an aid to interpretation by allowing the three components to be toggled on and off so that they can be viewed individually and combined as required. Additionally, small multiples of any view can be created for three variables at the click of the polygon map button if the 'Single/Multiple Views' button is set to produce multiple views. This button shows an icon with a square representing a single view when off, and three squares representing windows containing views of each of the selected variables when on. Hence a researcher assessing the relationship between three variables might flick between individual colour components on a polygon map, adding and subtracting components as required, whilst displaying three box-plots to show the variation of each variable for each recorded case, along with three scatter plots that show the combination of each pair of variables. The nature of cartographic visualization is demonstrated by this example and the supposition that the use of multiple linked views of a data set provides more useful information than the views can individually is supported.

Figure 3.20 Multivariate colour composites mapped in geographic and population space, US Midwest.

The RGB colour model is used to show percentages of those aged twenty-five or over with a high school diploma, with any college degree, and with a higher or professional degree in the US Midwest using red, green and blue for the respective variables.
Plotting the colours in a cartogram representation gives a spatial indication of the numbers involved in each multivariate group and allows counties of similar population total and attributes to be identified. Figure 3.20 shows percentages of those aged twenty-five or over with a high school diploma, with any college degree, and with a higher or professional degree. These data are combined using red, green and blue for the respective variables. The multivariate composition of localities can be assessed as demonstrated above, by selecting a spatial subset and using colour variation over the local range.

The technique can be particularly useful for highlighting spatial variations amongst dependent variables. For example figure 3.21 shows the distribution of age groups in Leicestershire, with the percentage of those in the 0-15 cohort mapped in red, the 25-59 group in green and the 60+ group in blue. Those zones symbolised by colours that contain a large amount of red represent zones with a relatively high proportion of residents in the 0-15 age cohort. Orange zones contain a high proportion in both the 0-15 and 25-59 groups, whilst blue zones represent a more mature population profile. The 25-59 group is most significant in the larger rural areas with children dominating in a few city wards and those of pensionable age constituting a significant proportion of the population in selected urban and rural wards.

Possible explanations involve life cycles, the distribution of different ethnic and religious groups, the locations of residential homes and other services in the city, property prices, and older people moving out of certain rural areas due to an inadequate provision of public services and transport. Whatever the true explanations, a pattern is produced that describes a multivariate geographic...
space, ideation can take place, and other data views and additional techniques and data sources can be used to address these issues. The cartogram reveals the differences in age composition between the city and its rural hinterland whilst also revealing the complex local variation of the city centre.

3.5.4 A novel attribute view

Multiple dot-plots provide the key to viewing more than three variables. Producing a number of dot-plots in 'cdv' and using the dynamic links to view the distributions of cases allows multivariate distributions to be assessed. In figure 3.22 plots show the numbers of those who are unemployed or on a government scheme, have a limiting long-term illness, and live in overcrowded conditions respectively, with a single case representing the Latimer ward highlighted in each view.

![Figure 3.22 Parallel dot-plots with a selected case highlighted.](image)

Dot-plots showing numbers of those who are unemployed or on a government scheme, have a limiting long-term illness, and live in overcrowded conditions in Leicestershire, UK, 1991. The cases represent the subset selected in figure 3.18 above, for clarity. The 'Latimer' ward is highlighted in each plot.

If multivariate box-plots are ordered in parallel and the symbols representing the same case in each joined by a line, a parallel coordinates plot is produced (Inselberg, 1995). Inselberg (1995) believes that Descartes coordinate system "enabled us to work so well with two and three variables (dimensions), but perhaps led us astray by putting the axes orthogonal rather than parallel to each other". He addressed the limitations of using orthogonal axes for displaying multivariate information by producing parallel coordinates plots to provide a "systematic and vigorous way of visualizing N-dimensional geometry". The geometry of hyper-dimensional space can be derived from parallel-coordinates plots with experience. Multi-modal distributions, multiple clusters (in one or more variables), outliers and ranges can be identified. Wegman (1990) begins with a clear explanation of the representation and describes ways of interpreting it, whilst Inselberg (1996) provides 'A Guide for the Perplexed'.

Parallel plots can be created for any number of variables in 'cdv'. Variables are selected for inclusion by highlighting the variable name in the attribute selection list box and clicking the primary mouse button whilst holding down the 'Control' key. The current list is displayed at the
bottom of the view control panel. Alternatively click the 'Re-Set' button to load the three variables selected as 'v1', 'v2' and 'v3' into the list of variables that will be displayed in a parallel plot. Parallel plots provide a multivariate signature for any case. Outliers, clusters multiple modes and other qualities of multidimensional data sets are relatively easy to determine. Experienced users can interpret more subtle patterns from the geometry of such plots (Wegman, 1990). 'cdv' provides a number of observer-related behaviours that aid interpretation. Static parallel-coordinates plots only show the relationships between parallel axes, a restricted number of bivariate relationships in a multivariate space. 'cdv' orders the axes by calculating Pearson's product moment coefficient for all combinations of variables and plotting the two most correlated as the first pair of axes, followed successively by the next best relationship with the axis to the right. The two most correlated variables, those at the left of the parallel plot, are displayed with orthogonal axes as a conventional (but dynamically linked) scatter plot to aid interpretation. Indeed any pair of parallel axes can be displayed as a scatter plot by clicking an axis with the right mouse button. Any parallel plot produces an ordered set of n axes from n! possible combinations. In 'cdv' the ordering of axes can be changed by interactively dragging axes to new positions, allowing relationships between any two variables, and any of the n! combinations of axis orders to be investigated.

An example is shown in figure 3.23 where a dynamic parallel-coordinates plot has been created for the sub-set of Leicestershire selected above. The axes represent (from left to right) per capita values for: those who are unemployed or on a government scheme; those with a limiting long-term illness; those living in overcrowded conditions; those in full time employment; the number of cars; and those in part-time employment. The line symbols are coloured by the age distribution variables used in earlier examples. Lines are highlighted and shown in all other selected plots by touching them with the cursor. Here, the highlighted line represents Latimer (see how the shape of the line mirrors the distribution of the highlighted symbols on the parallel box-plots shown in figure 3.22).

Figure 3.23 Dynamic parallel-coordinates plot for selected cases.

The view shows the statistical distribution of six variables for the subset of Leicestershire shown in figure 3.22, with the path of a single case, the Latimer ward, highlighted.

The first three axes equate to the variables shown in the box-plots of figure 3.22 in which the symbols representing the same case are highlighted.
It displays relatively high values for the first three variables and low values for the last three. Two other cases reveal similar 'signatures' that buck the regional trend shown by the denser cluster of black lines (Spinney Hill and North Braunston). Geographic questions might be asked as a result of this analysis and additional information and views can be used to begin to answer them in a process of exploratory spatial data analysis.

Whilst this example demonstrates the nature of the plot and its utility, figure 3.24 shows a parallel plot containing far more data, with lines representing multivariate data for all 187 wards in Leicestershire. Here colour symbolism is applied to the lines representing each zone to emphasise the multivariate values through an RGB composite. The composite uses the values of the variables shown on axes 1, 3 and 4 to specify the amounts of red, green and blue respectively in each symbol.

Multiple modes can be detected, with the mass of green lines representing zones dominated by the middle aged. Other modes are identifiable too, in particular the blue lines that represent zones with relatively low proportions of children and high proportions of those of pensionable age, and a series of purple lines that show zones containing relatively low numbers of the middle-aged compared to the young and old. Dynamic links to a spatial representation and alternative interactive parallel plots that contain additional socio-economic data allow researchers to assess the interactions between socio-economic criteria across space and so undertake geography.
3.5.5 Scripting with ‘cdv’

The functionality described here, provided through a geo-graphical user interface, implements a whole series of novel interactive views of a geographic data set and constitutes an interactive GeoGUI that is suitable for visualization map use. Experience with the software provided in appendix B4 and the guide in appendix C2 will confirm this. However one of the suggested advantages of the scripting environment as a suitable medium for exploratory map use is the level of flexibility for modification. In addition to the GUI element of the software, ‘cdv’ provides a whole series of scripts that structure recognised geographical information into a series of internal structures such as lists and arrays and manipulate them mathematically and graphically. These are available for modification so that the geographic information can be manipulated and symbolised by the map user to suit their data set and exploratory needs. By running ‘cdv’ through the ‘wish’ widget shell application and entering commands in the Td console the structures and procedures used by the software become available to the researcher for utilisation and modification meaning that the finite functionality of the GUI can be extended with relative ease.

Two example scripts provided here demonstrate useful functionality that is not available through the standard graphical interface but were added as required by taking advantage of the ‘cdv’ data structures and Td/Tk functionality through the command line. They illustrate the way in which the flexibility of scripting can be used to provide ad hoc functionality when required and are available for inspection in Appendix A4. The first transforms variables that comprise of percentage values on to a common scale so that the magnitudes of the ratios can be compared. The absolute maximum and minimum values of the three individual variables are calculated, and each of the attributes scaled to this new range. The results are displayed in the base map and population cartogram (figure 3.25). The second identifies all wards whose centroids fall within 2500m of the Latimer centroid in Leicestershire and highlights the symbols that represent these wards in a number of views. In figure 3.26 the colour composite map and population cartogram shown in figure 3.21 are brushed in this way, along with a parallel plot displaying the same variables and colouring as figure 3.23, but for all wards in Leicestershire rather than the previously selected subset.

The first pair of colour composites in figure 3.25 show the relative spatial distributions of three dependent variables representing percentages of the workforce in Leicestershire who are employed through a government scheme, in a part-time capacity and in full-time self-employment. Each individual variable uses the range of its colour component from 0% to 100% to represent its own data range. Red ‘hot-spots’ of government intervention can be noted along with a significant rural/urban variation in the forms of employment that people are in. The darker symbols mean that a higher percentage of the workforce falls into other categories, such as ‘unemployed’ and ‘long-term ill’. A dark cluster is evident in the urban area of north-east Leicester.
The images show the proportions of the population in three forms of employment in Leicestershire, UK as recorded by the 1991 census. The top pair of images shows a polygon map and cartogram that use the full range of colours available for each individual component. The bottom pair of images shows the same information but using a common scaling factor so that the three variables represented by the colour components can be compared in absolute terms. The script calculates absolute data values by taking a number of related variables into consideration and using a common maximum, minimum and so range when scaling the data.

The second pair of images shows the results of using the script to map the absolute values, which have been calculated on a common scale so that the ratios can be directly compared. The proportions of those working on government programs can be seen to be low in absolute terms as the images are dominated by green and blue, and the interrelated urban/rural and east/west elements of the distribution are evident. Despite the schemes having the largest proportional significance in some of the most populous wards, in absolute terms even they are dominated by those in part-time and self-employed work.

The highlighted wards in the base map of figure 3.26 show zones with centroids falling within 2500m of the Latimer ward. The cartogram illustrates the relative population of this area. Note that the topology of the inner city has not been fully retained by the cartogram algorithm. The parallel plot has the same axes as that shown in figure 3.23, but includes all wards in Leicestershire rather than the subset. Much of the regional variation in the statistical distributions of the selected variables is contained within this small area of Leicester. The wards within 2500m of the 'Latimer'
centroid appear to contradict the county norm and provide extreme values for all but the fourth axis ('Full-time Employment / Capita'). The script used to achieve this selection is provided in appendix A4.

Figure 3.26 A simple scripted extension to 'cdv': Brushing by geographic distance. (C1)

The script identifies all wards that are within a specified distance of a selected ward and highlights them in a series of chosen views. This is implemented in a procedure that takes a single ward identifier as its first argument and returns the identifiers of all wards that fall within a distance specified by the second argument. Here the procedure has been bound to the action of clicking a symbol. It is then called with arguments representing the id of the clicked zone and a user-specified distance. The symbols with that are tagged with the returned identifiers are highlighted in each view. Here the 'Latimer1' ward has been clicked and all other wards within a distance of 2500m are selected.

Each of the procedures contained in these scripts could be added as observer-related map behaviours by binding them to a cursor action or GUI button. In the former case a checkbutton could switch the relative/absolute colouring on/off. In the latter case a binding could ensure that the polygon ID was retrieved from any symbol that is interacted with. The distance could be specified by an entry or scale widget, or by dragging a distance on the map. Scripts that define additional functionality such as this could be incorporated into the standard GeoGUI functionality for future use. The functionality described in 'cdv' was developed in this format.
3.6 Conclusion

The `cdv` software and the examples provided here demonstrate that the combination of VISIC techniques with two-dimensional cartography to produce the kind of cartographic visualization environment outlined in the introduction to this volume is achievable with Tcl/Tk. Indeed the examples offered here show that the approach can be used to provide a rich and appropriate environment for investigating relatively large and complex spatial data sets and a significant research environment has been achieved using the examples outlined in the previous chapter and the described methodology. The Tcl/Tk GUI language supplies a unified, object-based approach to visualization in computer cartography. The `cdv` software implements this approach with a tailored application of the dynamic symbolism capabilities of Tcl/Tk to provide an interactive visual interface to a particular type of data set. It has varying degrees of suitability for similar data sets, depending upon the questions that need to be asked. As it stands the software means that a spatial overview of a multivariate data set can be acquired, an understanding of the nature of the data developed, error or bias detected and eventually, analysis undertaken, results produced and knowledge gained.

This environment not only equips the computer cartographer with more than adequate techniques for symbolism, and excellent facilities for dynamic mapping, unifying traditional cartographic techniques and those used in visualization, but also embeds them in a rich, open and active community of Tcl/Tk software developers who swap ideas and code over the Internet. As all `cdv` code is documented and available on the Internet, additions can be made to the functionality, as demonstrated above, using the Tcl/Tk commands outlined in previous sections. This form of support, sharing and reusability, which is augmented by a global discussion group (comp.lang.tcl) and huge variety of WWW resources and expertise, is an important element of the approach. These are excellent conditions for those wishing to visualize data, as code is accessible, and scripts are modular and procedural with elaborate variable substitution capabilities. This means that each cartographic visualizer can take advantage of general procedures that can be adjusted to account for the peculiarities and quirks of every real world database. Functionality can be rapidly scripted on the fly to suit other data sets, respond to their intricacies and to prototype new techniques for representing spatial information. Examples demonstrating the way in which the software and methodology have been applied to other data sets and research issues are provided in the following chapters, which further demonstrate the flexibility of this scripting environment for visualization. In these examples the approach is used to implement visualization map use and some pertinent research issues in geographic visualization are addressed.
Interactive Maps for Exploratory Spatial Data Analysis:
Cartographic Visualization
Approach, Implementation and Application

4. Extending the Approach: Dynamic Mapping for Exploration Representation

Jason Dykes
Department of Geography
University of Leicester
UK
4. Extending the Approach: Dynamic Mapping for Exploring Representation

The functionality described in the previous chapter has been implemented in the 'cdv' software and placed in the public domain. In addition to providing a benchmark for the assessment of dynamic cartography for visualization the current research addresses the ongoing issue of 'representation' outlined as an area for research priority by the ICA Commission on Visualization (MacEachren, 1998). This chapter and the one that follows demonstrate the flexibility of the methodology adopted here by presenting a series of novel views that respond to interaction from users and by incorporating additional data sets. These are prototyped from the 'cdv' core procedures, available in appendix C3, using additional Td/Tk scripts.

In this chapter a series of examples demonstrate ways in which Td/Tk widgets have been used to create dynamic graphics with observer-related behaviour that reveals additional information about the mapped value. These example behaviours encourage exploration of the representation, providing insight into the transformation from geographic information to map, which is suitable for visualization map use. They provide more sophisticated forms of spatial and statistical query than those introduced up until now. Some of these techniques apply transient symbolism to enduring symbols using the 'configure' and 'itemconfigure' functionality of the scripting language whilst others create transient symbols upon request by the user. Each is designed to follow the mandate of Buttenfield and MacKaness (1992) who advocate the use of technology to allow us to see the unseen in order to understand the nature of spatial data. In the instances presented here, this is achieved by providing the user with additional graphical information so that the geometric representation can be fully investigated and better related to the data that underlie it, and the process of transformation that has taken place in order to produce it.

4.1 Exploring Cartographic Representation

The recognised ability of map-users to interpret spatial patterns from imagery and relate them to geographic places and processes forms the basic tenet of much of cartography. However, the trade-off between the application of cartographic techniques that simplify or generalise information in order to elicit patterns and the requirement of the user to see 'the full picture' and gain information about the cartographic process that have taken place in order to reveal more about the data model is a complex one that is map-use specific. Additionally it is possible that users can focus on certain aspects of the map and not consider other, equally important elements. For example, when presented with a choropleth map readers may be drawn to conclusions about the relationships between areas at a scale approximating that of the enumerated units, due to the way in which the map pattern is divided, rather than at larger regional levels. A solution is the provision
of dynamic mapping functionality that permits exploration of the cartography applied in creating the representation itself. This section contains examples that incorporate observer-related map behaviour to provide additional information about representations of a spatial data set that is not apparent in static views. It demonstrates the use of the spatial information inherent in the smart GUI symbols to present information about relationships in space, amongst attributes and between cartographic representations.

4.1.1 Spatial structure

Tk items possess spatially 'smart' characteristics that make them particularly useful structures for dynamic mapping. Canvas commands exist to find the bounding box of an item and to add tags to all items that either overlap a bounding box or are enclosed by it. Additionally tags can be added automatically to the closest item to a canvas coordinate. These commands were introduced in chapter 2 and are demonstrated by the 'Focus' option of the 'Visual Variables' software provided in appendix B3. They have the following forms:

```py
canvasname bbox tag
canvasname addtag tag overlapping left top bottom right
canvasname addtag tag enclosed left top bottom right
canvasname addtag tag closest x y
```

This information can be used as the basis for identifying the closest item to a location, or all items that overlap with the bounding box of an entered item as shown here:

```py
canvasname bind tag <Any-Enter> {
    canvasname addtag overlapping-item overlapping [canvasname bbox current]
    canvasname addtag closest-item closest %x %y
}
```

Map Example 7 takes advantage of these features. It builds on the examples introduced in chapter 2 and through appendix B2 and is provided in appendix B5. The example can be run by clicking the appropriate icon and can be seen to contain a variable selection GUI and two graphical data views (figure 4.1). These will be familiar from previous examples: a polygon map and a scatter plot showing the relationship between two variables. The scatter plot axes provide a label exhibiting the name of the variable mapped on an axis when it is touched (the vertical axis has been touched in the upper image of figure 4.1). The variable can be changed by clicking an axis to select it (the selected axis is normally coloured orange) and choosing a variable button. The button invokes a procedure that moves the location of each scatter plot symbol depending upon the data value of the selected variable.
The polygon map uses the 'closest' functionality to select the closest item to any location. Double clicking on the canvas creates a transient symbol showing a location, the 'X' on the map in the upper image of figure 4.1. If this symbol is clicked the closest polygon is highlighted, as is the related item in the other view. The process is as follows:

- Bind canvas to procedure on button click, passing current cursor position
  \[\text{canvasname bind <Button-1> “procedure %x %y”}\]
- In procedure add a tag to the closest item to the location passed as 'X' and 'Y':
  \[\text{canvasname addtag closest-item closest $x$ $y}\]
- Return all tags of item with this tag:
  \[\text{set tags [canvasname gettags withtag closest-item]}\]
- Strip the id from this list
  \[\text{set id [lindex $tags end]}\]
- Highlight items in all views with this tag
  \[\text{viewN itemconfigure $id -fill Green}\]

Figure 4.1 Map Example 7 - Spatial smartness: Closest neighbours, brushing and contiguity neighbourhoods.

The 'bbox' argument to a canvas command can be used to return a rectangular bounding box around any item. The 'overlapping' argument will return any items that a bounding box overlaps. A rectangular brush is implemented with this technique. Dragging the mouse whilst depressing the
primary button and the keyboard shift key over either canvas will create a transient rectangle or 'brushing box'. Clicking the rectangle selects overlapping items in the view, cases relating to which are highlighted in all other views. Transient symbols such as the rectangular brush can be removed by clicking them whilst depressing the 'Ctrl' key. This type of feature was implemented and demonstrated in ‘cmd’ and is shown in the top image of figure 4.1.

However, the 'bbox' and 'overlapping' commands can be applied to polygons on the canvas in order to extract information about neighbourhoods. Holding down the 'Shift' key and touching any item in any view results in the appropriate symbol being highlighted dark green in both views. All items within the bounding box of the highlighted item in the spatial view are identified, and then highlighted in both views in a lighter green. The result is shown in the lower image of figure 4.1. This forms a simple 'neighbourhood probe' allowing geographical relationships to be displayed in spatial and attribute views. A rectangle item representing the bounding box is drawn around the selected item in the polygon map to visualize the process that is taking place (figure 4.1, lower image). These techniques permit a degree of geographical brushing and add a spatial component to the statistical views (Dykes, 1996).

Contiguity relationships are implicit in any digital data set that comprises of polygons, including enumerated data sets. Regions and neighbourhoods based upon these topological relationships can provide a useful basis for spatial analysis at a variety of spatial lags and can extend analytical attention away from the spatial scale of the enumeration. The method indicated here will not result in a list of the contiguous zones, it does however provide a useful geographic subset for testing contiguity, from which a list, ‘neighbours(zone)’, can be developed efficiently. Whilst it can be useful to identify immediate neighbours, spatial structures operate at a range of scales, and appropriate 'neighbourhoods' have varying sizes depending on the phenomenon being studied. Unwin (1981) stores information about map spatial structure in a binary connectivity matrix 'C', with rows and columns for each zone in a polygon map. A value of 1 signifies that the zones represented by the row and column are neighbours and 0 denotes non-neighbours. He demonstrates that by powering the matrix ‘C’ neighbours at spatial lags of ‘n’ contiguous zones can be derived. Thus lists of neighbours at a variety of spatial lags can be constructed for each zone, and the array of lists ‘neighbours(zone)’ can be given an extra dimension ‘neighbours(zone,lag)’. It also demonstrates that the kind of topological database listed by Egbert and Slocum (1992) as a requirement of analytical software for enumerated data sets is implicit in a Td/Tk GUI. Using this information it is possible to create a neighbourhood probe at a variety of spatial lags, which might be determined by particular mouse action or a selection widget. Anselin and Srnivov (1996) have demonstrated that a list structure provides a more efficient way of generating higher order adjacency matrices. An algorithm that creates such matrices and provides
a graphical statement of progress is implemented in 'cdv' (see figure 4.2). This allows us to see the unseen and assess spatial structure at a range of scales.

![Figure 4.2 Calculation of topological relationships between polygons in 'cdv'.](image)

Symbol smartness is used to create a bounding box around each polygon in turn and to report other polygon symbols that overlap this box. The list of coordinates is checked to assess whether each potential polygon is in fact a neighbour. The lists of neighbours are then used to generate higher order adjacency matrices. Here zones representing the electoral wards used in the 1991 census are displayed for the Teign Valley region in South Devon, UK. The zones are considered in turn and shaded in green when their neighbours have been calculated. The most south-westerly zone is under consideration here and the bounding box shown. Five overlapping polygons are tested for contiguity.

This has been developed to produce a form of observer related behaviour that identifies topological relationships within a geometric view of an enumerated spatial data set. The 'contiguity probe' is demonstrated in figure 4.3 through a sequence of five images. The figure shows how a data set comprising of the wards of the county of Leicestershire has been mapped in 'cdv' and can be dynamically interrogated for information about the way in which data for a continuous region have been collected in discrete areas. The central orange zone shows the location of the cursor as it is moved across a representation of Leicestershire. A list of the case identifiers (ids) of contiguous neighbours exists for each zone for a number of spatial lags. Those symbols that are tagged with an id contained in the list of first order neighbours are coloured dark green. Their neighbours, those which are second order links from the central orange zone, are coloured lighter green, and so on, up to a specified spatial lag from the interrogated zone. In figure 4.3 the maximum lag shown is that of order four.

This form of probing extends the representation in order to overcome some of the interpretation 'problems with the areas' (Dykes and Unwin, 1998) and so to assist the map user in interpreting the spatial structure of the information provided to them.
4.1.2 Attribute composition

Transient symbols can be combined to produce transient graphics for the interrogation of data values, to show more information at a location than is provided by a single symbol, such as those representing enumerated zones in 'cdv'. An example of a transient graphic that shows attribute composition at an interrogated location comes from an application that maps employment structure for an enumerated data set through a dynamic choropleth. The application utilizes Tcl/Tk for visualization as outlined in the previous chapters and is reproduced with permission (Duckham, 1996). Responses to the map user's interrogation of a choropleth are provided by reconfiguring items in an enduring additional view. This view is the human shaped dynamic glyph shown in figure 4.4. It appears next to the choropleth that shows the spatial distribution of selected variables in a similar way to the examples outlined above. However, when a zone on the dynamic choropleth is touched the graphic is reconfigured to contain stripes of distinct colour and texture. The height of each stripe shows the percentage of employees in various employment sectors of the economy for the selected zone as recorded by the 1991 UK census of population.

Contrasting hues and bitmaps distinguish the nominal classes. A pick-axe for 'Agriculture & Mining', a computer for 'Manufacturing', a chefs hat for 'Hotel & Catering', a dollar sign for 'Money & Finance' and so on. The clothing of the glyph appears to vary as the cursor is moved around the region displaying the composition of any brushed zone. As the map designer pointed out this view through the multivariate data space provides a signature for each zone and might be a useful way for some users to interpret the data set. For example a location that is represented by a glyph with 'money bags pants' and a stripy jumper could represent a good location for industrial investment,
with plenty of financial skills and a broad industrial base (Duckham, 1996). The application also contains a toggle button that reconfigures the icon shape to a more sedate rectangle enabling users to compare the heights of the stripes without the additional noise and demonstrating the interactive nature of these kind of graphics. Figure 4.4 shows the variation in composition of the graphic as the cursor is moved across a number of zones. The sequence provides an impression of the dynamic nature of the additional view.

Figure 4.4 Interrogation with transient symbols in an enduring additional view: Attribute composition. (C 1)

The glyph shows information about sectors of the workforce in brushed zones on an interactive choropleth. The widths of the bands represent different sectors as recorded by the 1991 UK census of population.

4.1.3 Cartographic abstraction

A third aspect of cartographic representation that can be investigated with observer-related dynamic techniques is that of the geometric projections performed in Tobler’s (1979) ‘mapping’ transform. We have already seen how cartograms use the visual variable of space to depict an attribute (usually population) whilst retaining a partial mapping of geographic space (usually restricted to the topological aspects of it). Dynamic graphics mean that the spatial dimensions can be used to map more than one variable almost instantaneously.

Strachan (1996), praises Dorling’s non-continuous cartograms (Dorling, 1995), but expresses a familiar hindrance, that the identification of places of interest is not always easy. He suggests the provision of a transparent overlay with a paper atlas of cartograms. The previous chapter demonstrated how ‘cdv’ uses interactive brushing techniques to overcome some of these limitations. This section provides additional observer-related behaviours to illustrate how abstract spatial representations, such as those used by Dorling (1995), can be deciphered with smart symbols.

A dynamic cartogram of the kind shown in Map Example 8 provides two aids to interpretation (see figure 4.5). Firstly brushing between a spatial view and a cartogram is of assistance in identifying
locations. This technique was demonstrated in the previous chapter. Additionally a scale bar is linked to a procedure that interpolates between the spatial and non-spatial views. This can be used to transform the locations of the graduated circles from their geographic locations recorded in the spatial data set (figure 4.5, lower image) to those computed by the algorithm in a more abstract topological cartogram space (figure 4.5, upper image). The 'canvasname tag coords' command is used to change the coordinates of the oval items according to the position of the scale bar, whenever it is moved. The first order contiguities are displayed with line symbols for reference.

This technique is analogous to the Grand Tour between alternative statistical views of data sets (Cook et al., 1995) and is a smoothed form of alternagraphics (Tukey, 1977). Graduated circle symbols can be dragged with the mouse as is the case in the 'cdv' circle maps, to see behind overlaps and investigate the adjacency structure. Dragging is implemented by combining the cursor action with a 'canvasname tag coords' or 'canvasname tag move' command to redefine or increment the coordinates of the currently selected item.

Figure 4.5 Map Example 8 - Dynamic views to aid interpretation of cartographic abstraction. (B5)

The map contains dynamic features including the ability to vary the locations of the symbols on the map that uses circles to represent population totals. The scale bar can be used to move the symbols between their geographic centroids (lower figure) and calculated locations that minimise overlap whilst maintaining contiguity – cartogram centroids (upper figure). First order contiguities are shown as a network and symbols on the circle-map/cartogram can be dragged interactively. The interface illustrated is that available in appendix B5. Documented scripts are available through the menus.
In figure 4.6 the contiguity probe introduced earlier in this section has been applied as a brush to linked views in order to reveal the spatial structure of both representations. The neighbourhood probe is a useful tool for relating cartogram spaces to geographic space and accounting for their flaws. The polygon map is probed for contiguous neighbourhoods at a variety of spatial lags. The cartogram is updated immediately and the success with which neighbourhoods are retained can be judged (it will of course be spatially variable). Equally the cartogram can be probed for neighbourhoods that are shown on the polygon map. The cartogram neighbourhoods can be visualized with transient symbolism as in figure 4.6.

Dorling (1994) notes that non-continuous cartograms can disrupt geographical topology. Figure 4.6 shows that due to the optimization procedure utilized neighbours in the cartogram are not always spatially contiguous and neither are spatially contiguous zones always adjacent on a cartogram. Figure 4.6 takes advantage of dynamic linking to explore and reveal the locational characteristics of the cartogram produced from the Leicestershire data set and can be used to assess its success in retaining the topological nature of geographic space.

The figure provides a complex snapshot showing a single view of a data set where many hundreds might be required when undertaking exploratory analysis. An interactive map would aid legibility as the map viewer could use the probe to request views that answer particular queries by showing certain aspects of the data. Developing views that reveal unknowns and confirm or refute the existence of detected patterns is at the core of the process of visualization (MacEachren, 1994).

4.2 Exploring Statistical Representation

'cdv' was developed as a tool for dynamic graphics rather than statistics, but for assessing local variability within distributions the concept of transient symbolism can be extended to include the
production of temporary local statistics, calculated when users interrogate locations within a view. Both transient symbolism and transient symbols can be used to provide analysts with information about attribute distributions and the statistical techniques used to measure them. Here an enumerated data set comprising of the counties of Wisconsin is used in most examples for illustrative purposes. The data show median household income for 1990 and were obtained online from the US Census Bureau (1997).

4.2.1 Statistical distribution within a locality

In 'cdv' the contiguity probe has been extended to show the local variation of a single attribute in two ways. Symbols within the defined neighbourhood can be symbolized depending on the distribution relative to a local mean (effectively showing local z-scores) or relative to the value of the probed symbol (thus showing local variation from the central zone). Each time a polygon is entered a procedure determines the neighbours and calculates a local mean. Neighbouring polygons are mapped in a divergent scheme (Brewer, 1994) with red or blue hues depending on whether they are above or below a value with which they are being compared. This is either the local mean or the central value depending upon which statistic is selected.

Figure 4.7 Performing local statistical queries with the contiguity probe. (C1)

Two sets of maps show mean household income in Wisconsin as recorded in the 1990 census. From the left these show an intensity-shaded map of the spatial distribution, a local z-score statistic for the probed zone, a local difference statistic. The colour scheme ranges from high to low using red through yellow to blue. More saturated shades represent extremes. The upper and lower maps show different results as the probe is moved across the state.
The lightness and saturation of hue are varied to produce a colour scale that reflects the strength of variation in terms of the local standard deviation. Darker, more saturated hues represent the extremes. Yellow polygons denote values that are within 0.25 standard deviations of the mean (figure 4.7). The probe shows variation within a locality that might not be detectable when using a globally defined colour scheme such as one based on intensity. The colour scheme is effectively varied depending on the data set. As with sub-set selection in 'cdv' this allows the full range of symbolism to be applied to the local data range and so local patterns are visually emphasised. MacEachren (1995) advocated this technique for enhancing ranges where contrast is required to explore pattern. Here the 'ranges' are regional, defined by a locality that provides spatial 'focus'.

The local difference statistic shows the difference between the value of the interrogated polygons and its neighbours, again in terms of the local variation. This is a good tool for detecting local form as a zone surrounded by blue indicates a local high, whereas a zone surrounded by red indicates a local low. Troughs, slopes and ridges in the data surface can be identified at a local scale.

Use of lists of contiguous neighbours at a variety of lags allows measurements to be visualized at a range of scales. In figure 4.8 a central county is selected, and the immediate neighbourhood shows a north-east to south-west 'slope' of medium household income. As the scale at which the statistics are measured is increased the detected pattern reverts direction, revealing a more general trend from south-west to north-east, showing the dominance of Milwaukee and its suburbs on the distribution of mean household incomes. The effects of processes operating at a number of different scales can be assessed with this flexibility.

Figure 4.8 Performing multi-scale local statistical queries with the contiguity probe. (C1)

The results of using the contiguity probe for a single zone is shown at a range of spatial lags. The maps show mean household income in Wisconsin as recorded in the 1990 census. A zone is selected and the local z-score statistic is calculated at a number of selected scales from the immediate neighbourhood (a contiguity lag of one zone from the selected case) to a spatial lag of four zones. The colour scheme is that used in figure 4.7.

4.2.2 Visualizing spatial autocorrelation

A spatial indicator of variation can be calculated from standardized local values. One of the best known indices of spatial association amongst geographers is Moran's I (Cliff & Ord, 1973; Getis & Ord, 1992). The statistic computes a measure of spatial autocorrelation based on the spatial...
distribution of variation around the mean. When measured globally the index may fail to detect non-stationarity in data sets where several regimes of spatial association might be present across a region (Hansen, 1996). Capabilities for interactive visualization, such as those demonstrated here, have generated a propensity for using techniques that explore these local patterns of association in spatial data (Anselin, 1995). Regional variation in spatial autocorrelation can be detected graphically by extending the contiguity probe to produce transient symbolism that reveals Moran’s I for some local region defined by the probe position. Additional interactive graphics can be used to explore the composition of the statistic, evaluate the features of the distribution that it is recording and assess its utility. The intricacies and foci of the index outlined in appendix A1 can be examined in detail with the type of probe provided here. These include assessments of the influence of the arbitrary decisions that are made concerning the weight function that is incorporated into the statistic, the influence of particular zones on a geographic neighbourhood and the suitability of the statistic as an index of spatial association/variation for descriptive purposes.

An autocorrelation probe is shown in use in figure 4.9. When calculating a single statistic factors such as the contribution of individual zones, the sensitivity of the statistic’s value to the inclusion of particular zones in the locality and the chosen weight matrix are often overlooked. In figure 4.9 an additional view has been produced that allows these factors and their influences to be assessed. Transient symbolism is used to show the covariance matrix for each probed locality.

![Figure 4.9](image)

**Figure 4.9** Using the contiguity probe and local statistics to produce a local covariance matrix.

Colouring on the map represents local z-scores around a selected zone. In the associated graphic each row and column represent a zone in the selected neighbourhood, each cell a combination of these. Coloured combinations represent immediate contiguity between zones with the colour representing the covariance. Red represents positive covariance, blue negative. The sum of covariance for each zone is represented by the length of the bar above each column in the matrix. Colouring is used as redundant symbolism to show whether the covariance is positive or negative. The sum of all values in any column of the matrix represents a local index of spatial autocorrelation. This is symbolised by the length and colour of the bars displayed above the matrices.

Moran’s I is calculated in the images of figure 4.9 at the same locations as the local z-score was calculated in figure 4.7. Moran’s I is a scaled product of local covariance, multiplied by a weight factor. If the distance weight factor is a binary indicator of contiguity, I is proportional to the covariance, the sum of the product of the variation about the mean for all neighbours. So
neighbours that are both above or both below the local mean have a positive effect on Moran's I whereas neighbours that traverse the mean have a negative effect. The size of the effect is dependent upon the magnitude of the variation.

The coloured map in the left image of figure 4.9 shows the local Z-scores, and the view to the right of the map shows the covariance matrix. Each row and column represents a zone in the defined locality, and where two zones are contiguous the corresponding square in the matrix contains symbolism. Red represents a positive value, blue a negative one. Once again the saturation or lightness of hue symbolizes magnitude. Each column is summed to produce a figure of 'local influence', which is symbolized by the length and colour of the bar above each column. For the central zone, which is compared with all of its neighbours, this is Anselin's local Moran's I (Anselin, 1995). The sum of the columns (and so the sum of the covariance matrix) represents Moran's I for the selected region, the figure for which is displayed numerically above the matrix. This provides a solution to the issues raised in section 1.5.1 and appendix A1. The images to the right of figure 4.9 show the same calculation and associated graphic for another probed zone demonstrating the interactivity of the feature. Dynamic graphics showing the probe moving across the county are provided in appendix C1.

Observer-related behaviour has been added to this abstract statistical graphic to enable the map user to explore the representation, relate it to the spatial data set and assess its meaning. Direct interrogation of the graphic provides information about the statistical representation. Figure 4.10 shows how a column can be clicked to show the zone that contributed the value, and the zone on the map and the constituent symbols on the covariance matrix are highlighted. Any individual cell within the matrix can be interrogated too, and the two zones that contributed the value are revealed on the map (figure 4.10, right image). Thus the dependence of a statistic upon a small number of values, its stability, and an idea of what it represents and why, can be determined visually.
A further figure shows that the autocorrelation graphic can also be extended to measure at a variety of scales based upon spatial lags. Figure 4.11 shows how second order neighbours can produce a positive value for Moran's I when the first order figure is negative. The contiguity matrix is interrogated and a pattern emerges where a single zone has a dominant effect. There are authentic geographical reasons for this occurring in the example provided. Menominee County, the dominant zone that is probed in Figure 4.11, has an extremely low median household income. The unique multivariate attribute behaviour of Menominee is a result of the high proportion of Native American residents living in a reservation within the county.

Figure 4.11: Interrogating a covariance matrix at a spatial lag of two zones.

Here covariance matrices are presented at spatial lags of one and two zones. Columns and cells in the matrix and zones in the map are dynamically linked to aid interpretation of the graphics.

The covariance matrix visualization can be extended to include an inter-zone weight function by using cell size to represent the proximity of each pair of zones. 'cdv' provides an equation interface through which the weight function can be varied, and the effects of different distance weightings on the local statistics assessed. Figure 4.12 shows weight functions of 1, 1/d and 1/d² respectively. As the weight function becomes increasingly significant the pairs of zones that have closest...
centroids dominate the index, as shown by relative sizes of the squares in the covariance matrix, and the changing lengths of the bar symbols, in the second and third images. The effect of the 1/d function in the example shown is to increase the value of the local indicator, as the two zones with the highest values are closest. The 1/d² function further attenuates this effect until the previously measured negative autocorrelation is eliminated. The graphical representations of the statistic allow its components and parameters to be assessed interactively and the sensitivity of the index, and utility of the measurement, to be observed in a manner mirroring that outlined above by MacEachren (1994) with regard to spatial, rather than statistical, graphics.

Figure 4.12 Adding a weight function to the index of spatial autocorrelation. (C1)

Here an additional parameter is introduced. A weight function can be selected from a GUI and the effect that it has on the statistic represented by the size of the cells in the covariance matrix. From left to right these images show covariance matrices mapping a binary, inverse distance and inverse distance squared weight function. Note that the local index changes from a negative to slightly positive value due to the increasingly dominant effect of the two closest zones, the central zone and that to its immediate north, which have similar high values. The interactive graphics contain the behaviours outlined in previous figures.

4.2.3 Visualization of distance weighting coefficients

Techniques such as those outlined above have led to an increasing interest in the derivation of local statistics to describe the multivariate nature of geographic areas (Unwin & Unwin, 1998). The final example introduced in the previous section provides evidence that small variations in relatively arbitrary weighting coefficients can have a significant impact on local statistics, and potentially conclusions derived from them and the decisions made as a result. Where distances are used to provide metric weightings that reflect the spatial significance of individual zones in spatial statistics alternative dynamic techniques become especially useful.

A large number of parameters must be considered when calculating a local statistic. These include the distance function, centroid definition, and location at which the statistic is measured. Each can have a major impact upon the index calculated, yet there is a degree of discretion in selecting an appropriate choice. An approach that recognizes this variation in potential outcomes is one where the stability of the index is determined when parameters are varied. This idea is statistically...
analogous to Muehrcke's (1990) concept of map stability, where confidence in representations is assessed by examining the extent to which small changes in the method of map creation effect the visual product.

Transient symbolism can be used to assess this stability by illustrating the distance weights of zones, the effects of varying the location of a point that represents an area and the location from which a statistic is measured. The tools that result permit experimentation with multiple statistic definitions, allow map users to explore the representations that they create and provide information about the sensitivity, stability and validity of measurements made from maps.

**Visualizing distance weighting between points**

The boundaries shown in figure 4.13 are counties in Illinois, derived from the Midwest data set for use as examples. Td/Tk code developed as an addition to 'cdv' provides the ability to digitize routes along the base map by moving the cursor and clicking the mouse. Localities can then be defined by creating a shaped brush and positioning it at each location on the digitized route in turn. This process is known as tracing. In the software implementation presented here brushes can be circular (basing localities upon distance inclusion) or irregularly shaped polygons.

Local statistics functionality allows indices to be derived for the computed locality with weightings based on a distance function for each point on the trace in turn. The function is entered through the keyboard and interpreted by a Td/Tk string parser.

![Interactive variation of distance functions in a weighted local statistic.](image)

The effect of variations in the arbitrary weight function used to calculate a local statistic is demonstrated here. A tracing tool is used to select a point location (black cross) and distance (orange circle) from which a neighbourhood of zones can be calculated. The bounding box of the symbol displaying the distance (black square) is used to search for likely neighbours for which distances between centroids are calculated. The resulting neighbours are shown in red with green outlines and blue crosses at their centroids. The saturation of colour shows the relative weighting of zones in a local statistic. The three images represent inverse distance, inverse distance squared, and distance subtracted from the maximum in the neighbourhood. The zones represent a selection of counties in Illinois.
The process is extremely graphic with distances, local zones and centroids symbolized. Figure 4.13 shows the results of changing the distance weighting function. Centroids are shown in light blue, and local zones are highlighted with green outlines. A red hue is used to colour zones contained in the locality, with the saturation of the hue dependent upon the strength of the distance function for each zone.

In the first example a $1/d$ function is symbolized. The two zones with centroids near to the sample point (indicated by the black cross) are dominant. This effect is enhanced with the $1/d^2$ function used in the second example. The third shows a function that subtracts the distance to each centroid from the maximum distance to the furthest centroid (which therefore has zero weighting), and demonstrates a less skewed distribution amongst the local weights. This alternative produces a local statistic with more regional input.

**Visualizing attribute location sensitivity**

A second use of dynamic graphics is in examining the sensitivity of attribute location. The first image in figure 4.14 shows the weightings associated with a $1/d^2$ function. In this example centroids have been calculated automatically as the average of all of the x and y points in the zone. The two zones closest to the sample point have a meandering common edge that is digitized with far more points than the edge opposite.

![Image](image-url)
Observer-related behaviour has been implemented in Td/Tk to enable the map user to move the centroids interactively and assess the impact of this variation on the local statistics. In the second image in figure 4.14 the two centroids have been dragged interactively to locations closer to the 'centre of gravity' of the zone. The same weight function has been used, but the different strengths of red show the relative effect of an alternative centroid definition algorithm, and reveal the sensitivity of a distance weight based statistic.

This interactive movement can be used to the researchers' advantage, as centroids can be placed at estimated centres of population. In the third example shown in figure 4.14 the polygons have been reconfigured to appear transparent and a Td/Tk image item has been used to display a geo-referenced map. The centres of population can be viewed in each zone. In the fourth example the dark blue crosses show that the centroids have been dragged to new locations where centres of population have been identified. The final image shows how the same 1/d² function weights the attribute values of each zone with the centroid distribution as defined by comparison with the population centres noted on the scanned map. The zone to the south west of the point at which the local statistic is measured now dominates rather than those to the north and/or south. This is the case as this zone contains the most local significant centre of population. Two conclusions can be drawn from this discovery. Firstly, it would be relatively easy to argue that such a statistic provided a more useful insight into the elements of the real world that the initial transformation into geographic information was designed to provide. And secondly, the degree of variation demonstrates that analyses using local statistics are prone to subtle variations in the parameters used for their measurement. Visualization can address these issues.

**Variation with measurement location**

In this third example the sensitivity of the location at which a statistic is measured is assessed with visual means. Figure 4.15 shows a trace, digitized interactively with the mouse, about the original point of measurement. The distance weighting function is again 1/d² and the adjusted centroids are used throughout. As the trace moves (shown by the black 'X') the locality, defined by zones whose centroids occur within a user-specified distance of the measurement location, varies and the weights attributed to the various neighbouring zones are shown by the saturation of the red hues.

Whilst a statistic should be expected to vary as the measurement location changes the seemingly violent deviations in attribute weighting produced by the 1/d² function with very little variation in location demonstrate that a traced index should be employed with due care and attention.

This is particularly true if the location of the index is defined in a relatively arbitrary way as is the case if the locations are digitized interactively on-screen with a mouse, or interpolated between selected points. Any such index should be tested for local variability and used with due
consideration of its sensitivity to change. Dynamic graphics such as those shown here can provide analysts with an indication of the validity and stability of a statistical representation.

Figure 4.15 Interactive variation of locations at which a weighted local statistic is measured. (C1)

The effects of minor variations in the location at which a weighted local statistic is measured are shown. The light blue crosses show zone centroids that have been moved to the locations of dominant centres of population in each zone. The black crosses show different locations at which a weighted statistic might be measured. The black arrows show the path of this trace that investigates the stability of a local statistic. Relative weights of the same inverse distance function are symbolised with saturation of hue in three figures. The zones represent a selection of counties in Illinois.

By investigating the criteria that go into the construction of local indices in a flexible environment with graphics containing dynamic behaviours some of the issues concerned with the sensitivity of local statistics to these factors will become more apparent.

4.2.4 Investigating the variogram cloud

Variogram clouds are suitable for assessing the spatial dependence in data sets and for detecting spatial outliers, and their utility can be increased and enhanced by programming them with observer-related behaviours such as techniques for brushing between the statistical view and a map (Cook et al, 1997). Variogram clouds consist of n*(n-1)/2 symbols representing all possible combinations of pairs of n cases plotted in a statistical space where the horizontal axis represents the Euclidean distance between the pairs of points and the vertical axis represents some variation in statistical values (figure 4.16). In data sets with strong spatial dependence, where Tobler’s first rule of geography applies, the variance in attribute differences will increase as the distance between locations increases. Locations that are close, but show large variations indicate spatial outliers.

The utility of dynamic links between geographical representations of data and the variogram have been demonstrated by Cook et al. (1997). In 'cdv' several forms of brushing are utilised to relate
the points on the statistical plot to the case locations on the map. Brushing single zones on the map reveals all n-1 points on the variogram cloud so that the variogram, and level of spatial dependence, can be identified for any individual zone. Brushing symbols within the variogram cloud results in the two zones represented by the symbol being highlighted on the map. The zone outlines are coloured so that the zone values, identified by the intensity shading, can be compared (figure 4.16, top).

One of the difficulties of the variogram representation is the way in which symbols overlap. It is impossible to detect the densest areas within the plot when symbols overlap. This is a problem that is common to many statistical plots that contain large numbers of symbols and is an issue that will require more attention as plots with larger numbers of symbols become feasible (Unwin, 1998). Interactive graphical techniques can overcome some of these limitations. One possibility is to produce a density map. This can be achieved by giving each symbol an intensity and blending symbols so that the intensities combine and the densest areas are brightest (Unwin, 1994). An alternative approach is introduced here, that uses the lengths of a series of bars to represent the density of symbols in user-defined areas. In doing so the technique for exploring the representation to assess densities facilitates the comparison of densities between selected areas, enables cross sections to be created and permits a degree of journaling in the interactivity. The dynamic behaviours relate representation and map very closely and so permit exploration of the relatively complex and abstract statistical representation.

The central and lower sets of images in figure 4.16 demonstrate. A user-defined brushing box has been created by dragging the mouse across the variogram cloud. When the box is clicked a rectangular symbol is created to the right of the variogram, the length of which represents the number of cases that it encloses or overlaps. Additionally the pairs of zones represented by all cases identified by the box are shown on the map with lines that join their centroids. The box can then itself be dragged across the variogram and re-clicked to produce an additional bar for comparison and reveal appropriate links on the map. A checkbox allows real-time probing for densities to be toggled on and off. The series of bars to the right of the lower two images in figure 4.16 show densities sampled as the box was dragged across the variogram cloud at a constant height, thus revealing the densities of pairs of zones with a range of low levels of variation at increasing distances apart. Peaks and troughs in this profile reveal information about the nature of spatial dependency at a range of distances. The bars themselves are linked to the variogram cloud and map. When they are touched with the cursor the box on the variogram cloud moves to the position represented by the bar and lines linking all pairs of cases relating to points on the variogram cloud within the box at that location are produced on the map (figure 4.16, bottom). The lines on the map can be brushed to show the pair of zones that they represent on the map.
and the relevant symbol on the variogram cloud. Additional and higher resolution graphics are provided in appendix C1 along with explanatory text.

Figure 4.16 Investigating the variogram cloud - interaction with a linked map and statistical graphic. (C1)

The data used in these figures show proportions of the population aged 60 and over for electoral wards in the Teign Valley, South Devon, UK as recorded by the 1991 UK census of population. The variogram cloud plots a point for all pairs of cases. The positions in the cloud represent the distances between locations (x axis) and the variation in statistical values between the pair of cases (y axis).

The view includes a number of interactive features through which the graphical representations can be explored. Any case on the map view can be brushed and related symbols highlighted in the variogram cloud. The top image here shows symbols representing the bottom of the two highlighted zones in the variogram with orange highlighting. This is achieved by touching the zone on the map. The zone contains a high value and so most of the highlighted symbols are high on the variogram demonstrating substantial statistical variation. Linking is two-way, and the case highlighted in green on the variogram cloud has been brushed to show the zones that it represents, which are identified by green outlines on the map. The middle and bottom images show how a draggable box can be created on the variogram cloud and used to assess the density of points. Pairs of zones represented by points within the box are linked by lines on the map and a bar to the right of the variogram cloud is created to show the point density. This is a transformation of the visual variables used to represent density from location on the variogram to size, which permits comparison. Densities at different positions can be compared by dragging the box across the variogram cloud, which creates a series of density bars in real time. Touching a density bar returns the selection box to the position that the bar represents and shows the relevant links on the map.
This additional view, that uses appropriate symbolism, permits interactive exploration of the complex structure of spatial independence in the variogram cloud. The bar chart showing the density of the variogram cloud can be scrolled through and added to at any time and new sized brushes can be developed. Additional interactive features and options could be developed. For example, an additional option could be added to scale the bars by the proportion of all zones at the selected distance so that the density of points on the variogram cloud could be related to the expected density given the number of pairs of zones at the specified distance. The approach utilised here makes these kinds of additional scripts, and the creation and examination of additional observer-related behaviours, very practicable.

### 4.3 An Example Script

A final example is included to demonstrate explicitly with documented scripts that outline the procedures and commands used in Tcl/Tk. It is provided in appendix B5 and integrates many of the features introduced above. Map Example 9 (figure 4.17) confirms that multivariate views can be combined with abstract spatial views in a dynamic, linked, environment for visualization in which the observer controls temporally varying symbolism to extract information about the data.

A dynamic parallel coordinates plot (Inselberg, 1995) that shows the relationships between all twelve example variables has been added to the views used in previous examples. The plot, shown in figure 4.17, is not as powerful as the dynamic parallel coordinates plot implemented in 'cdv' in which axes can be re-ordered and pairs of variables visualized as transient linked scatter plots.

The views provided do however contain the spatial smartness demonstrated in Map Example 8. Clicking the secondary mouse button on any symbol will produce transient symbols that link cases to neighbours in the polygon and scatter plot views. These symbolize attribute differences, where the arrow and hue show direction and line width represents magnitude (figure 4.17, bottom left, top left). In addition the example provides a means for visualizing the effects of varying distance weights parameters across the British regions, as outlined above. This is achieved by double-clicking the polygon map to set a location, holding the Shift key and clicking the 'X' symbol that identifies the location. Symbols will be filled with a red hue, saturated according to the relative distance weight of each zone in all views. The coloured lines on the parallel-coordinates plot therefore provide distance weighted multivariate information on the 'locality'. The equation used to calculate the distance weights can be varied by typing a valid expression in the text entry box (figure 4.17, bottom left) and pressing return. The effects of a new equation can be visualized by holding the Shift key and clicking the 'X' symbol once more. The equation parser will interpret the +,-,* and / operators as well as the following functions and variables:
So legitimate distance weight functions include:

- $1/\text{pow}(d,2)$ inverse distance squared
- $\max - d$ maximum zone distance minus zone distance

The dynamic visualization techniques introduced, explained and demonstrated in this chapter with simple examples and real data sets and applications have illustrated how maps can be programmed with behaviours to visually explore cartographic and statistical representations of spatial data. Software such as that provided and demonstrated here, containing the kinds of techniques outlined in this section enables users to explore and assess the statistical and graphical information that they use to represent and analyse geographic distributions, multivariate information and data about variation within and between localities. The examples provided here demonstrate that extensions to the approach outlined in previous chapters can be used to implement graphics that address research issues and have profitable application.
Interactive Maps for Exploratory Spatial Data Analysis: Cartographic Visualization Approach, Implementation and Application

5. Extending the Approach: Dynamic Mapping Examples and Applications

Jason Dykes

Department of Geography
University of Leicester
UK
5. Extending the Approach: Dynamic Mapping Examples and Applications

This chapter demonstrates the flexibility of the approach outlined here by providing examples of its implementation across a number of data sets to address a number of different issues. These range from relatively small scale ‘add-ons’ to ‘cdv’ where Td/Tk scripts were produced to provide additional requested statistical or graphical functionality, to extensions that apply the basic spatial display and interaction procedures to entirely different data sets. An assessment of the demonstrated flexibility, robustness and future proofing delivered by the scripting approach is provided. The following chapter extends these ideas still further by outlining the production of distinct software that develops the Td/Tk scripts used in ‘cdv’ to incorporate novel media types, that were not readily available when the research started, for an additional application. This enables richer, less abstract qualitative data sources to be introduced into the analysis. The theme that the chapters have in common is that all examples provided here result from requests made by academics who required map products suitable for ideation. The results therefore demonstrate a progression from theoretical demonstration of the methodology to direct application for use in academia, demonstrating the flexibility, applicability and utility of the approach.

5.1 A Day in the Park at Schwalm-Nette.

Researchers at the Centre for Recreation and Tourism Studies, University of Wageningen, The Netherlands, became interested in assessing the utility of the techniques to visualize their data following a demonstration of the methodology along with some early ‘cdv’ prototypes at the REGARD user conference in Augsburg, Germany. A large multivariate database had been collocated containing information about the locations and attributes of users of a recreational area. They had undertaken the classic first step and mapped their data to try and uncover patterns in the visual process of ideation identified by DiBiase et al. (1994). Several static maps of the data had been produced, but the richness of the data set and multiple means of slicing through it meant that the type of flexible mapping approach espoused here was regarded as an exciting solution. It was certainly believed to be potentially more productive than producing successive maps through their cartographic department, each of which took a matter of days to turn around. The application seemed a suitable test for the techniques advocated here. The data were geographic, with a temporal element and a whole series of categorical attributes. Some time was thus spent assessing the needs of the researchers, collating and manipulating the available data sets and subsequently modifying the procedures produced in ‘cdv’ to produce a GeoGUI for this particular application.
5.1.1 Tourist behaviour data

The data depict space-time behaviour in the Schwalm-Nette park on the German/Dutch border where detailed locational and personal attribute data were collected at hourly intervals for nearly one thousand individuals over a four-day period. The researchers identified a need for an exploratory 'first look' at the information so that patterns and particular spatial behaviour could be detected and potential avenues for more rigorous research identified. The interest expressed was focused on the pathways and open spaces of the park and an assessment of the amount of pressure on locations within the recreational area during a diurnal cycle, rather than the individuals for whom data had been collected. The activities and characteristics of those who used the spaces were also regarded as being of importance as they related to the likely behaviours of park users and identified groups within society that might not be making full use of the recreational resources that were available to them.

The data consisted of three components:

- A geometric data set much like that used in the 'cdv' software for enumerated data, but with linear as well as areal spatial units. Each unit contained a unique identifier or 'id'.
- A series of lists representing the space-time behaviour of each surveyed individual. Each list represented an hourly period of the day and contained the ids of the spatial units visited during that hour.
- A single list for each individual representing their characteristics. These included socio-economic indicators, their age, sex, home location, the composition of the group in which they were travelling, their mode of transport and the items that they brought with them to the park.

5.1.2 A dynamic mapping solution for visualization

Tc/Tk was used to map the geometric data and produce data structures that contained the collected information. Each item (lines were used for routes and polygons for areas) contained a unique id, and was programmed to exhibit a series of observer-related behaviours. The most fundamental of these involved a change in symbolism to represent the number of people utilising the particular spatial feature whenever a scale bar that determined the time was moved. This is demonstrated in figure 5.1 and shown more effectively in the dynamic version provided in appendix C1. Thus a form of dynamic comparison between different periods was implemented, rather than a canned animation of the changes over time. The scale bar allows the user to move quickly backwards and forwards through time. The software has advantages over a time-series animation for investigation in that the user has total control over the speed and order of the sequence.
The user performs dynamic comparison by assessing the distributions of tourists at different time periods by interactively moving the scale bar that sets the symbolism of each map element accordingly. The maps contain other observer-related behaviours for interrogation. The updates of the reconfigured symbols are instantaneous.

A number of forms of dynamic re-expression were incorporated. Lines and areas can be symbolised in a number of ways. A variety of colour schemes can be applied to the symbols, with the classification limits defined by the user through the interface. This allows, for example, areas with more than some critical number of people at any particular time to be highlighted. As an alternative to the filled areas, graduated circle symbols can be used to represent the numbers of people in any region of the park. These features are shown in the cartography window of figure 5.2.

The interface addresses the issue of multi-scale aggregation over time by allowing users to select and vary the time period at intervals other than by the hour as specified in the database. This was achieved by presenting the user with two scale bars, representing a start and end time for aggregation of numbers of people to be visualized. The scale bars can be moved independently to set a period of time, and the map will respond accordingly. The scale bars can also be set to a standard increment and successive periods are displayed at that increment by dragging the linked scale bars. The dynamic version of figure 5.1 provided in appendix C1 shows the distribution of population at two-hour intervals throughout a single day.

In addition to providing the facility to assess the geographic distributions and slice through the temporal dimension in a number of ways, functionality was added to set restrictions on the data mapped through the attribute space. In addition to the widgets created to vary the cartography a series of GUI windows were produced automatically in response to the data set. Researchers are able to control the individuals counted and mapped by specifying a series of selection characteristics via the interface.
The cartography GUI (bottom centre) allows the user to perform dynamic re-expression. Numeric classification can be varied interactively as can the colour schemes employed and the types of symbols used to map counts of tourists. Behaviours that fall in to the category of dynamic comparison include the multi-scale temporal aggregation widget (top centre) and the attribute selection widgets (right) that allow the inclusion/exclusion of tourists that have certain specified characteristics.

This functionality allows the distribution of distinct sections of the population to be visualized and assessed. For example, in figure 5.3 the attribute widgets have been used to select and map the distribution of German and Dutch people, in cars or on bicycles, who are visiting the park as part of a group and currently inhabiting their permanent places of residence. The map shows the distribution at the selected time - 14:00 hours. By clicking the scale bar, or attribute buttons, the corresponding maps are displayed instantly and time and attribute series can be produced to analyse the data set. Clicking the ‘Perm’ and ‘Temp’ check buttons immediately maps people with the same population characteristics other than their current place of residence and so the distribution of those in temporary accommodation would be revealed. Moving the ‘Year_Of_Birth’ scale bar would re-express the data as an attribute series, depicting spatial behaviour of ordered age groups at any time. Map views of different scales allow location within the park and personal attributes to be linked to, and compared with, the distribution of tourist points of origin on a national map (figure 5.3). Equally national locations could be selected and the distribution of regional tourists plotted in the park. These factors, along with tourist social groupings, are the main concern of the research into the effect that national barriers have on tourist behaviour in parks near European borders.

The examples shown in figures 5.2 and 5.2 demonstrate an important aspect of the scripting approach, that the software runs on a variety of operating systems, with interfaces that contain a
native look and feel. Figure 5.2 shows the software running under Microsoft Windows™ whilst figure 5.3 shows the same application running under SGI Irix 5.4.

Figure 5.3 Visualization of Schwalm-Nette Time-Space Data Set - Geographic brushing techniques.

Here brushing is shown between maps of two scales allowing the locations of tourists within the park to be compared with places of origin at a national/international scale. The national scale map can be used to select/de-select tourists for inclusion in the park map. Equally the map of the park can be brushed to show the places of origin of the tourists at any location at any time. An early version of the attribute GUI is shown on the right.

5.1.3 Exploring representation

Observer-related behaviours that allow map-users to interrogate dynamic maps for additional information about the representation were introduced in the previous chapter. In this example temporary transient graphics were used to explore the temporal attribute composition of any spatial unit. Figure 5.4 shows line and circle symbols that represent numbers of tourists at different locations throughout the day. Behaviours have been added to the dynamic map so that transient graphics can be summoned by the observer to provide a temporal overview of each location. When any symbol is clicked, additional items are produced to create a line graph that reveals the numbers of people at that location at each hour through the day. If subsets of individuals are selected, graphs are shown for each sub-set. The green line on the small graphs in figure 5.4 shows the total number of tourists, as does the map. The other lines show population sub-sets selected by the map user, in this case the light blue line shows the number of men and the purple line the number of women. The graphs are dynamically linked so that lines representing the same population sub-set can be highlighted in each view. As the scale bar is used to select different hours of the day, the symbolism changes, and a red indicator shows the selected hour on the
graphs of attribute composition. Figure 5.4 shows these changes with a sequence of three images, representing three hours chosen interactively, in real time.

Figure 5.4 Visualization of Schwalm-Nette time-space data set - Temporal querying.

Transient graphics provide a longitudinal view at selected locations. Time is shown on the horizontal axis with numbers of people with particular social characteristics on the vertical axis of the graphics. Here the maps show the numbers of people at locations within the park at three hourly intervals. The graphs show this total number (green line) and totals of men (blue) and women (purple). The precise social characteristics are user-defined and selected from a GUI window (see figure 5.3).

The temporal dynamism of maps and other representations underlies all of these techniques. Immediate update is essential for visual analysis, and the approach presented here provides it. The following types of questions can be investigated by visualizing spatial data in this manner.

- What are the behaviour patterns of locals in the park?
- Which areas are over-used by long-distance tourists?
- Which areas are free of people during the busiest periods?
- Where do wind surfers come from?
- When do people from the Venlo area use Schwalm-Nette?

By establishing answers, or appropriate new questions that may provide fuller answers, researchers and planners can use visualization to further their understanding of the social and spatio-temporal patterns of recreation.

5.2 Combining Statistical and Geographic Views

The flexibility of the approach to cartographic visualization introduced here is demonstrated by a series of techniques for adding elements of spatial information to statistical plots that are incorporated in to 'cdv'.

The ideas result from discussions with delegates at the second EUROSTAT workshop and conference on new techniques and technologies for statistics (NTTS) in 1995. Following a presentation at NTTS in Bonn, Germany, a series of discussions with graphical statisticians led to the development of a number of techniques that implemented links between spatial and statistical
views of data. As a result, methods of combining both spatial and geographical information into single views of data are suggested.

At the NTTS meeting methods were introduced that permit real-time cartographic visualization (Dykes, 1996), as documented in the previous sections of this volume. Having presented this work and introduced the approach as one that is open, flexible and capable of producing dynamic graphics to suit specific needs, delegates were interested to know whether it would be possible to incorporate geographic information into statistical views. This section reports on some techniques for adding such information to scatter plots. These ideas program observer-related behaviour to permit brushing. They use additional symbols and compound symbols known as glyphs, and were discussed and implemented both during the NTTS meeting and subsequently.

Whilst identifying the locations of cases in statistical views is essential for ESDA, Unwin (1994) points out that diverse contextual knowledge is vital for geographical analysis. It is essential to know not only where a data point is located but also what else is there, what it's near to, and what that means. Some contextual information can be provided with geo-referenced images plotted as a backdrop to the more abstract geographic views demonstrated thus far (see section 4.2). This is achieved by plotting image items on the canvas in the Td/Tk approach. Vector overlays can provide additional information. These can be implemented using line items on a canvas in Td/Tk (see section 5.1). They could be programmed to be configured with transparency or visible colour depending upon the way in which a checkbutton is toggled allowing them to be transient and switched on or off from a GUI widget. These are still locational guides. Each location can be identified individually, and significant symbols may be highlighted to aid recognition in software like 'cdv'. However, alternative methods can be developed to incorporate elements of the spatial structure of the distribution in a statistical view (and vice versa). At the NTTS meeting an interest arose in the definition of local neighbourhoods based upon zone contiguity rather than metric distance, such as those demonstrated in the previous chapter. Methods of identifying such neighbourhoods on dynamic statistical and geographical views, and ways of visualizing them were discussed. This provided an excellent test of the flexibility of the Td/Tk approach to dynamic mapping.

5.2.1 Identifying and visualizing geographic neighbourhoods

In section 4.1 techniques and procedures for producing lists of topological neighbours of any polygon were introduced. A series of probes were then described that identify neighbourhoods interactively with transient symbolism to highlight the relevant symbols. This is an appropriate way of symbolizing neighbours, but other options may be necessary. If values need to be read from the map as well as neighbours seen, then reconfiguring the colour of the polygon outline might be preferable (e.g. '-outline Red'). If localities need to be identified in scatter plot view then the point
symbols used may be too small to highlight by changing their colours. Additional transient symbols might be required, such as those included in Map Example 9. These can be desirable as they can exhibit observer-related behaviours to provide additional information about the neighbourhood. Transient symbols can display, or be interrogated for, information about the two zones that they join. For instance the difference in the two values might be represented by a hue or the magnitude of some flow between them by the width of the symbol. In figure 5.5 (left) a symbol has been brushed in a geographic view and symbolism used to show the first-order neighbours. Line items have been created on a statistical view (right) to show the neighbouring cases of the brushed symbol. The figure shows that whilst the zone interrogated in the map has the local median value for the x-axis variable (percentage of population with graduate or professional qualification) it exhibits a significantly higher value for the y-axis variable (percentage of population with Asian or pacific island ethnicity). Multi-view brushing can be achieved with a ‘bind’ command that associates the event whereby the cursor enters any item that symbolizes a data point with an action to fill polygons or draw lines to symbolise neighbourhoods in all views. Another one could clear polygons, or remove all lines when the cursor moves away from the item. The result is a ‘neighbourhood probe’ that can be used to identify geographic neighbours whenever an item is touched, as introduced in the previous chapter. However, in this instance additional transient symbols are used rather than transient symbolism. This form of representation can be toggled on and off and has the advantages detailed above of using additional line items to symbolize neighbourhoods.

Figure 5.5 Viewing neighbourhoods in statistical graphics with transient symbols.

Contiguous neighbours of a zone probed in the polygon map (left), are shown for the corresponding point on a scatter plot (right). Arrows link the point representing the zone to those of spatial neighbours.

The map shows the percentage of those aged 25 and over in possession of a high school diploma or higher educational award in the counties of Wisconsin. The scatter plot shows the proportion of population aged 25 and over with a graduate or professional degree on the ‘x’ axis and the proportion of the population with Asian or pacific island ethnicity on the ‘y’ axis.
In figure 5.6 neighbourhoods are shown at a user-specified number of lags, to identify spatial structure in a polygon map and a population cartogram. The data are identical to those used in figure 4.6, but whilst the polygon map uses transient symbolism to show the neighbourhoods the same relationships are displayed in the cartogram with transient symbols. These can be toggled on and off and allow other geographic information to be mapped in the polygons.

![Figure 5.6](image)

The 'neighbourhood probe' shows topological information in linked views of the wards of Leicestershire at a variety of user-defined lags using transient symbolism. The data are identical to those used in figure 4.6. The polygon map uses transient symbolism to show the neighbourhood. The cartogram displays the same neighbourhoods with transient arrow symbols. Colour and line width represent the lag.

### 5.2.2 Visualizing all locational information

Having created the neighbourhood probe, it became apparent that other aspects of spatial information, such as location, could be added to a statistical view by applying symbolism in a more traditional manner. The idea was to produce two views of a data set, one spatial and one statistical and use a visual variable other than location to symbolise the locational information of one view in the other (whether geographical coordinates or bivariate attribute values). Initially this was achieved by combining red and green colour components. Td/Tk allows colours to be defined as a hexadecimal string with values for red, green and blue (e.g. `fill #RRGGBB` where red would be #FF0000 and white #FFFFFF). This is a particularly useful technique for multivariate symbolism (see chapter 3). By colouring each symbol in every statistical view with red and green components based on the coordinates of zone centroids, the locality of zones on statistical plots can be determined from their colour, and zone proximity deduced from the similarity of the shades (see figure 5.7). There are limitations to this technique, particularly due to the fact that the RGB colour model is not based upon a perceptual scale (though neither is the grey shade one used throughout cartography and for much of the value mapping presented here).
A red-green colour gamut can be used to relay spatial information. Here the x coordinate is represented by the percentage of red, and y coordinate by the amount of green. When combined a unique colour represents each position. These colours can be added to symbols on the non-spatial scatter plot. The statistical data displayed in the scatter plot are those used in figure 5.5.

Alternative means of symbolism were attempted too. Black and white glyphs represent the geographical locations in the top pair of images in figure 5.8, using the 'location' visual variable rather than colours. This involved plotting specially created bitmap items at zone centroids on the polygon map and at the positions determined by the geographical locations of the zones in the scatter plot.

There are parallels with other work here where authors use Tufte’s (1983) principle of small multiples, where the similar form of a series of small graphics is used to compare and contrast information, with the location of each graphic being used to convey additional information. Dorling (1994) uses statistical glyphs in geographic space to show the distribution of industry and owner occupied housing on cartograms of parliamentary constituencies and population. Brunsdon (1996) could extend his selection of maps through correspondence analysis by plotting multiple maps in the statistical space used to choose between alternatives.

The top pair of images in figure 5.8 use glyphs that comprise of crossing horizontal and vertical lines. The position at which the lines cross represents the geographic centroid of the unit at which the data were recorded. So the ‘↑’ glyph indicates a location to the north-west (see figure 5.8, top-right) whilst ‘↓’ represents a south-eastern case, ‘←’ is central-western and ‘→’ just north-west of the centre. All locational information is thus added to the scatter plot, which uses the glyphs to locate statistical cases in geographic space. Three groups have been annotated on figure 5.8 (top-left) showing a spatial and statistical cluster (green), two statistically similar zones that are geographically distant (blue) and proximate zones that have similar values for the Y-axis variable, but vastly different values for the X-axis variable (red). The lower pair of images shows the same data using a combination of the red/green colour composites and small multiples. Glyphs
consisting of a square containing a dot at the relative spatial location of the case are distributed in geographical and statistical space, and coloured with a suitable composite. The example uses glyphs where dots in squares represent location. So here the 'EH' glyph indicates a location to the north-west, whilst 'O' is south-eastern, 'C' central-western and 'H' just north-west of the centre.

These two pairs of images show the same information with different symbolism. Both comprise of a scatter plot showing the mean per capita income of Native Americans against the mean per capita income of the population as a whole for the counties of Wisconsin and a base map of the state showing county boundaries and centroids. The top pair use monochrome cross glyphs to relate geographic locations to positions in the attribute space. The green, blue and red zones represent a spatial and statistical cluster, two statistically similar zones that are geographically distant and proximate zones that display a similar mean incomes amongst the Native American populations but widely differing mean incomes at county level. The bottom pair of images shows the same data using square glyphs in combination with the red/green colour composites introduced in figure 5.7.

For monochrome graphical reproduction the top pair of figures suitably combine geographical and statistical information, whilst the lower figures might be a more appropriate static representation where colour is available. In a dynamic environment for visualization re-expression would enable each of the alternatives to be viewed in turn when requested by the user and other interactive properties could also be made available. Zooming and scrolling would feature to aid interpretation.
In addition each glyph can be raised above all others, brightly coloured, and linked to the corresponding glyph/zone on the polygon map when touched with the cursor.

5.2.3 Combining locational and statistical information - A dynamic 'scatter map'

A progression from these two views is to combine the spatial and statistical information in a single dynamic graphic that uses the temporal dimension either to animate between a series of re-expressed views or to effect dynamic re-expression controlled by the user. Such an approach is analogous to the 'projection pursuit guided tour' (Cook et al., 1995), a dynamic tool where an overview of multivariate data is shown through a continuous sequence of low dimensional projections. A spatial example is provided by Hurley's (1988) 'Data Viewer' program that interpolated transformations between geographic and statistical views of point distributions. This can be achieved in Tcl/Tk by calculating the difference between coordinates for each glyph in the two views, and dividing it into a series of regularly spaced steps between the two known locations. In Tcl/Tk both the canvas 'move' and canvas 'coords' commands can be employed to change the location of each item by the relevant number of pixels between steps. The move argument to the canvas widget command has the form:

```
canvasname move tag x y
```

The coords argument to the canvas widget command has the form:

```
canvasname coords tag newcoords
```

Figure 5.9 The 'scatter map' - Glyphs representing geographic locations.

A scale bar permits smooth dynamic movement between the statistical (scatter plot, left) and geographical (polygon map, right) representations of the data. Clicking a glyph reveals an arrow that links the locations of the glyph in each view. Once again the data represent the mean per capita income of Native Americans against the mean per capita income of the population as a whole for the counties of Wisconsin.
Progression between the stepped locations can be controlled by the '-command' option of a scale bar. Figure 5.9 shows how the 'scatter map' can be implemented. The scale bar beneath the graphic producing smooth transition between the two views and allowing spatial and statistical information to be examined in a dynamic, user controlled way. Appropriate grey shades are applied to the polygon outlines and the scatter plot axes so that they fade or emerge depending upon the position of the scale bar.

Eight steps from the 'scatter map' sequence appear in figure 5.10. An additional interrogation function is illustrated in these figures. When symbols are clicked an arrow is added to show the route of the symbol between locations on the scatter plot and the map. Once again this transient symbol could be used to provide additional information about the case whose locations in the two alternative spaces it represents. The locations of the glyphs on both views can thus be traced at any stage. Clicking arrows removes them from the view.

The representation provides a dynamic interface to a combination of statistical and graphical information. Note the arrows that appear when a symbol is clicked to show the locations in geographic and statistical space. An animated sequence is provided in appendix C1.

5.3 Local Multivariate Views

The flexibility offered by the Tcl/Tk programming environment means that new views and ways of interacting with spatial data can be prototyped and tested. The combination of the wide range of
symbolism possibilities offered by Td/Tk symbols, their dynamism and inherent spatial information, make the environment suitable for computing local statistics that build on univariate local association statistics, such as Moran's I, and extend graphical symbolism to the multivariate visualization of such information.

Local indicators consist of a combination of locational information with multivariate attribute data. The need to investigate 'traces' was highlighted following meetings between researchers at the University of Leicester, UK, and statisticians based at the Institute for Mathematics, University of Augsburg. Tracing involves digitizing a route along which local statistics can be measured. The method is similar to that introduced using the contiguity probe, but can relate to discrete locations rather than zone centroids and usually takes place along a route with a neighbourhood defined by the user.

At a series of meetings between the two groups of researchers, funded by the British Council Anglo-German Fund and Deutscher Akademischer Austauschdienst, it was suggested that such traces could provide a useful means of investigating local variations in multivariate statistical information along linear geographic features. It seemed likely that collaboration between those with skills in the fields of statistics and geography might produce novel and useful new methods that spanned the fields.

5.3.1 The need for local statistics

Local statistics are an appropriate technique for the exploratory analysis of geographic information for a number of reasons identified by Unwin and Unwin (1998). Firstly spatial data can be expected to show spatial dependence, the essence of all geographical analysis, epitomized by Tobler's first rule of geography that all things are related, but near things will tend to be more closely related than distant things. Spatial dependence means that most of the classical methods used for the statistical analysis of data sets are inappropriate without some geographical modification. Secondly many analyses, particularly those using area data are subject and often solely attributable to the way in which the recognized geographical information is extracted from the real world. The modifiable area unit problem (Openshaw, 1984; Fotheringham & Wong, 1991) provides an example. Thirdly, stationarity cannot be assumed to occur in any process that operates over a real geographic space as variations will occur in phenomena that affect the process. Attempts to use classical statistical methods to perform spatial analysis on geographic data must overcome these problems. Additionally researchers must be aware that the larger numbers of geographic cases being processed by modern computers means that the chances of including regions with different geographic properties in the same analysis becomes greater, just as a shift to higher spatial resolutions means that effects relating to differing spatial scales may become superimposed. Alternative methods of analyzing geographic information are required. Unwin and Unwin (1998) note that this has resulted in a trend towards an assessment of what it is that gives specific
locations their unique character, rather than the identification of what places have in common and application of simple models that explain geographic variation over large areas. They note and recommend the use of local statistics in an exploratory framework, in order to learn more about each individual datum by relating it to neighbouring locations as a direct analytical procedure. They identify several reasons for this type of analysis including: region building by finding areas of similar value amongst a number of indicators; locating the boundaries between regions or anomalous areas within regions; identifying departures from regional norms when modelling spatial processes, especially where this gives an indication of scale-driven changes in the nature of these processes.

Unwin and Unwin (1998) also note the need to develop data structures that easily support the computation of local statistics, especially through the ability to flexibly define neighbourhoods and display the resulting statistics. The following examples demonstrate scripts added to ‘cdv’ that enabled the previously identified data structures and GUI techniques to be used to implement some of the suggestions, investigate their utility and assess the techniques. They use a data sub-set from the Midwest data set for clarity, focusing on the state of Illinois.

5.3.2 Locality selection - an interface

A GUI widget was added to ‘cdv’ to control tracing. Points at which local information is required can be digitized interactively on the base map and are linked with a series of arrows showing the direction of the trace. Delimiting the neighbourhood around each sample point can be achieved in a number of ways as requested by Unwin and Unwin (1998). Distance inclusion is one possibility, where distances are drawn on the map with an interactive draggable line. Any zone with a centroid within the specified distance of a point on the trace is included in the locality. Shaped brushes are another possibility and are defined with a ‘rubber-band’ box, or by drawing an enclosed polygon in ‘paint-brush’ style. Brush based selection can be set to include all zones that overlap the brush or those that are totally enclosed by it. The brushes can be constrained to remain at the orientation at which they are drawn or forced to rotate to retain their relative orientation to the direction of the trace (if for example a rectangular buffer around a linear geographic feature is being investigated).

Localities are defined when a trace has been digitized and a method of locality selection chosen. The map user then slides a scale bar to select locations along the trace in order. The software highlights the included zones at the selected point on the trace and adds cross symbols to identify the positions of their centroids. Once zones have been selected to create a locality, information about the locality can be calculated or visualized. An additional GUI is used to define a distance weighting for the local indicator. The magnitude of the distance weighting coefficient for each zone is then depicted by symbolizing the local zones with saturation of hue. The relative effects of varying the distance coefficient, the centroid locations, and the point at which a sample is taken
can then be assessed as demonstrated in the previous chapter (Dykes, 1997). A distance inclusion technique is used to define a locality, along a trace shown by the dark line across the counties of Illinois in figure 5.11.

Figure 5.11 Visualizing neighbourhoods and distance weighting coefficients from a trace.

Here a trace has been digitised across the counties of Illinois. A neighbourhood is shown around the third location on the digitised trace in the first three images. The fourth image shows a location around the eighth point on the trace. Zones within the neighbourhood are defined as those whose centroids fall inside the circle shown at an interactively determined distance. The square bounding box is used to minimise the search amongst polygons that overlap this symbol. Zones within the neighbourhood are shaded. In the final three images the influence of each zone within a neighbourhood on a local statistic is symbolised by the saturation of hue as defined by the selected distance weighting mechanism.

The crosses and zone shades reveal the locations of the centroids and the distance based weights resulting from the selected weight function respectively. The dark square shows that a Td/Tk bounding box search around the circle symbol depicting the selected distance is used to reduce the search for included zones to those overlapping the box. The potential efficiency savings of such a spatially restricted search based upon the spatial smartness inherent in the GeoGUI symbols are significant. Distances between the trace location and centroids for each overlapping zone are calculated and those within the specified distance included. The weight function is then used to assign a weight and colour saturation to each zone. By moving the scale bar provided on the trace GUI the user is able to see successive localities as defined by the current settings. These can be varied interactively at any time, as can the route taken by the trace.

5.3.3 Visualization of multivariate local variation

Once zones have been selected interactively, a variety of local indicators could be computed and displayed statistically or graphically. For exploratory purposes, and in accord with the graphic nature of the analytical environment, prototype multivariate graphics are produced here. Multivariate local graphics embrace the concepts of cartographic symbolism, that the information
processing capabilities of the eye-brain combination can be used successfully to analyse and assess complex geographic information, and apply them to interactive probes. Here the multivariate information is displayed in the form of local parallel coordinates plots (figure 5.12). These are a variation of the plot realised in ‘cdv’ and introduced in chapter 3. At their simplest they show the relationship between a number of chosen variables within the locality (figure 5.12a). The attribute data displayed in the plots represents data on educational attainment and proportions of the population living in conditions of poverty by age group. From left to right the axes show: the three educational attainment variables - percentage of those aged 25 or over with a high school diploma, any college degree, a higher or professional degree; the percentages living below the poverty level in the following age cohorts – 0-17, 18-59, 60+. The variation of the selected attributes and relationships within the locality can be assessed. A multivariate mode and two local outliers are visible in figure 5.12a. The graphics contain all of the dynamic and linking properties of the global parallel-coordinates plots and also incorporate circle symbols that show a selected additional attribute, zone population totals in this case.
Interactive local parallel-coordinates plots for multivariate spatial data query. Here parallel plots have been created from neighbourhoods selected by a digitised trace across the counties of Illinois. Data are shown for six variables for each of the seven zones in the selected locality. Two-way brushing is available to link related symbols on the spatial and statistical views. The circles on the first axis represent an additional selected variable, total population in this instance.

a) The grouping of the lines through the multivariate data set shows a local multivariate mode. Two statistical outliers are evident.

b) Adding the saturation of hue applied to zones within the neighbourhood to the line symbols on the local parallel plot combines the multivariate and geographic information. Here the dominant zone, that with the greatest weight, is seen to follow the local multivariate mode. A $1/d$ weight function is utilised.

c) Changing the weighting function to $1/d^2$ results in the map and graphic being updated. The relative saturation of the dominant zone is stronger, as is its influence on the local indicator.

d) The locality can be moved along the trace interactively. At a different location the zone with the greatest weighting is a multivariate outlier containing most of the population within the locality. This suggests a geographic core that is distinct in terms of its multivariate characteristics and a periphery with relatively homogenous attributes.

The local view updates to plot the information relating to each new locality as the sample location is moved along the path of the trace. Various visualization options exist to reveal more about the local distribution. The colouring applied to the included zones in the map view, based on the distance function, can be used to show the proximity of the zones symbolised in the parallel plot to the centre of the location defined by the trace (figure 5.12b). Closer zones dominate the graphic and in figure 5.12b the zone with heaviest $1/d$ weighting can be seen to follow the local mode. In figure 5.12c the GUI has been used to vary the weighting function. A $1/d^2$ function has been visualized with the effect of weighing the graphical indicator towards the zone containing the sample point. An interface that permitted variation of each of the three features identified in section 4.2 would provide further exploration of the sensitivity of the indicators and their validity.

As the sample location is moved along the trace lines representing spatial units fade in and out of the locality and so the graphic. The result is a multivariate, distance weighted, local, dynamic view, which contains map behaviours that allow the user great flexibility and control over the way in which their multivariate spatial information is transformed into a series of graphics for exploration. In the fourth image attention has moved along the trace to another point. Figure 5.12d

Figure 5.12

Interactive local parallel-coordinates plots for multivariate spatial data query.
demonstrates by showing that this location along the trace the zone with the greatest weighting is a multivariate outlier containing most of the population within the locality. This suggests a locality with a distinct geographic core and a periphery with comparable multivariate characteristics.

5.4 Assessment of the Flexibility and Extensions

As did the examples provided in the previous chapter, the three applications described here demonstrate that the combination of the methodology introduced in chapter 2 along with the example software outlined in chapter 3 provides a rich environment for producing interactive graphics for visualization. The difference here is that rather than developing techniques as a direct addition to the 'cdv' software in order to explore forms of dynamic representation, the solutions presented were produced as a direct result of the suggestions and needs of independent researchers or research groups.

The software produced to assess the tourism data set proved to be a success in terms of the exploratory analysis of this type of data. It was developed rapidly from the 'cdv' code (the initial prototype was delivered to Wageningen in less than a week). The institute used the software to assess these data and developed the techniques to accommodate other larger and more complex data sets for parks in the Netherlands (van der Knaap, 1997). The Tcl/Tk approach was also utilised to undertake further theoretical research whereby interactive graphical techniques were applied to the analysis of networks in order to assess tourist behaviour (Hillman, 1995). Additionally the combination of data set and GeoGUI outlined here are used in teaching in the Institute to demonstrate the complexity of tourist behaviour. A whole series of other techniques could be added with more time and investment. RGB colour composite maps such as those available in 'cdv' would allow researchers to combine and contrast groups with selected attributes. Using symbols to map individuals, rather than spatial features would provide insights into the spatio-temporal behaviour of tourists and produce an interactive 'movie' of recreational activity during a day in the park. Individuals could be mapped as items with shape, colour, texture, and outline width conveying any of the personal attributes, much in the way that Chernoff faces (Chernoff, 1973) have been used in static maps to portray multivariate spatial data (Dorling, 1994).

The views that accommodate both spatial and statistical information were also produced for exploratory spatial analysis. They demonstrate the utility of Tcl/Tk as a tool for creating specific dynamic graphics to represent individual data sets, and exemplify the power of cartographic symbolism for ESDA in a dynamic computer environment. Those techniques that used colour and location visual variables to add spatial information to statistical views demonstrate the cartographic functionality of the approach. The type of highly interactive example that morphs between
transformations of a data set that employ the location visual variable to show geographic and statistical information is analogous to the ways that brushing have been identified as a means of combining locational and attribute data. The implementations presented here demonstrate the flexibility of the Td/Tk language for cartographic visualization and illustrate the way in which transient views of data can be effective. The prototypes were developed in a single evening from existing ‘cdv’ procedures whilst at the NTTS workshop with graphical statisticians. They allow a series of projections of restricted aspects of multidimensional information to be synthesised, as Tobler’s (1979) second transformation from geographic information to map is varied continually in order to take advantage of the benefits of using simple and effective symbolisation techniques to elicit information from rich data sets.

The local multivariate GUI developed and demonstrated here provides an extremely flexible locality selector that enables the user to explore the local variation within a geographic region and to assess the stability or otherwise of the local indicators produced. The tool reaffirms the utility of the approach and provided the research group with both an analytical device to assess the production and sensitivity of traces and an initial expression of multivariate local graphics with a geographical component.

The approach used in these applications has proved to be capable of addressing research issues in representation and visualization with a high degree of versatility and considerable robustness. The initial implementation was a useful source of cartographic functionality upon which to build more specific routines for particular application and the scripting approach provided the flexibility to add functionality as and when required. It is possible, however, to take the approach further. The forms of representation available in a dynamic mapping environment for cartographic visualization can be extended to incorporate less traditional means of presenting information than those noted up until now using the approach outlined here. These possibilities are reported upon and implemented in the chapter that follows.
6. Extending the Approach: Virtual Environments and Multimedia

Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization Approach, Implementation and Application

Jason Dykes
Department of Geography
University of Leicester
UK
6. Extending the Approach: Virtual Environments and Multimedia

The high level programming features and graphical capabilities of Td/Tk have been utilised thus far to produce highly interactive maps that use relatively standard forms of abstract symbolism to convey information through accepted visual variables (Bertin, 1983). The high level control of external processes that is available within Td/Tk provides an opportunity to address some of the research issues associated with utilising new data formats and representing data in novel ways as outlined by the ICA Commission on Visualization (MacEachren, 1998). In addition to the variety of means of symbolism, excellent facilities for dynamic linking and support for data reading/writing provided by Td/Tk, the scripting language supplies a number of image read/manipulate/display capabilities and means of interacting with external programs. Consequently it provides a rich environment for developing software maps that use less abstract representations of reality than those demonstrated up until now and/or link existing software with bespoke Geo-GUIs. It is thus suitable for prototyping a range of highly interactive maps that run external processes and geo-reference, organise and provide access to information from a whole series of digital media. These multimedia maps can use photo-realistic representations of the real world to provide a navigable and virtual environment. Such capabilities have been implemented in a piece of demonstration software that is available on the accompanying CD in appendix B7. The ‘panoraMap’ software shows how the approach advocated in the previous chapters can be extended to provide dynamic maps for visualization that incorporate exogenous information and novel media. These data sources can be linked through a Geo-GUI that provides a virtual environment in which visualization can take place in context.

6.1 An Application Specification

The software was developed in response to a particular application with a relatively specific set of requirements. It was produced to address the use of virtual environments and information technology in teaching fieldwork and is presented here for a number of reasons:

Firstly it demonstrates that the approach used has real application and can be used as a successful solution to a visualization issue. The tenet that the advent of software maps allows dynamic cartographic products to be designed for a specific map use underlies much of the argument outlined here. The software is an instance of the transfer of the approach presented in chapter 2, into example implementations and then applications as shown in figures 1.6 and 1.7. This transformation is achieved with a very specific and well-defined objective. The aims, objectives and rationale behind the software are thus presented in detail.
Secondly it addresses important research issues that are currently of concern to the academy. Methods of representation using new forms of interface and novel media, the integration of data views and links between maps and data are key to the ICA Commission on Visualization research agenda (MacEachren, 1998). The combination of quantitative statistical information with 'exogenous' data that provides context in which informed analysis can take place is being advocated by leading statisticians (Cleveland, 1998). The software responds to these issues.

Thirdly the use of novel media and production of a non-traditional interface that provides a virtual environment within which multimedia information can be located and assessed through external software reinforces the conviction that the methodology used here for visualization is transferable and has some degree of longevity and future-proofing.

Finally whilst the software addresses research issues and provides a cutting edge interface that is in many ways suitable for undertaking academic research it is designed for performing relatively small-scale rudimentary analyses in undergraduate education. By exposing a non-expert user-group to techniques for visualization a preliminary assessment of the suitability of this method of exploratory graphical analysis can be made. In addition, the range of visualizers is extended from geographic researchers to non-expert students who perform exploratory analysis in the private domain, revealing information that is unknown to them with highly interactive maps. This application of visualization reveals geographic knowledge that is known to the community as a whole, to individual map users, as part of the process of education. The distinction between visualization and communication is evidently blurred by applications that utilise visualization outside of the research realm by providing dynamic maps for user-centred knowledge acquisition.

6.1.1 Software rationale

This section outlines the software requirements and rationale behind the development. A fuller explanation of the educational reasoning behind the application is provided in appendix A5.

The aims and objectives of fieldwork are manifest, and vary between subject area, institution and teacher, but the ability to perform comparison, to develop observation skills in order to achieve this and to analyse collated information are key (Williams et al., 1997). Comparison might be between geographic locations, models (including maps) and reality, or primary and secondary data sources. Observation involves interpreting the natural environment. Analysis is analogous to the process of academic research and involves the processes of exploration, confirmation, synthesis and presentation outlined in figure 1.4.

The comparison objective can be achieved in the traditional way, by visiting a number of locations, assessing models into the field and taking measurements for analysis in the laboratory (Gardiner & Unwin, 1986; Warburton & Higgitt, 1997). An alternative approach is to provide virtual...
environments that are inherently spatial, permit 'observation' and provide a rich data set that includes a series of models. If such an environment were to permit data analysis, an aspect of fieldwork that has often been neglected (Haigh & Gold, 1993), substantial benefits to fieldwork teaching would be likely to occur (Williams et al., 1997).

'Virtual Reality', or VR provides an attractive user interface for achieving such educational objectives (Winn, 1997; Boyd-Davis et al., 1996; Laurillard, 1997). By immersing the user within the representation, and providing them with familiar real world affordances data recording a complex three-dimensional environment can be navigated and interpreted successfully. In terms of the schematic representation of the cartographic process VR can be regarded as an interface that is sufficiently immersive that a users map image operates in such a way that the map user performs and responds as though they were within the representation (see figure 6.1). VR can provide an appropriate interface for undertaking some statistical tasks that involve higher dimensional data through visualization (Symanzik et al., 1997; George Mason University, 1999).

The schematic diagram (adapted from Robinson et al., 1995) shows a range of virtual environment interfaces that provide the user with varying senses of immersion and virtual reality. Each of these contain elements of reality or realistic affordances. The actual sense of reality will depend upon a number of issues including the data, hardware, and projection device used, the levels of interactivity, the behaviours embedded in the virtual environment and the cognition of the user.

In the situation outlined here, where comparison and synthesis are key, ideation and student-centred learning are objectives and time is of the essence, visualization is an appropriate method to use for the analytical process. This is particularly so as minimising the time between data collection and analysis improves the performance of students (Gardiner & Unwin, 1986).

Thus a dynamic map was required that extended the type of GeoGUI for cartographic visualization presented up until now to provide a virtual reality interface and to incorporate additional media types. Specifically the software was required to:

- Support visualization by providing a navigable virtual environment for the interactive exploration of spatially referenced information.
- Maximise utility by using affordable and accessible software, equipment and data resources. Particularly an avoidance of 'GIS', reliance on more prevalent data formats and focus on the development of a usable interface were advocated.
- Permit comparison by providing realistic world views and incorporating dynamic re-expression of spatial information.
- Include contextual information gleaned from multimedia data types and new data formats such as data from digital collection and logging devices.
Figure 6.1 A Transformational View of Cartography (Tobler, 1979): Virtual Reality.

The schematic diagram adapted from Robinson et al. (1995) uses the location of the symbol depicting the map user to represent the possibilities for immersion in cartographic products. 'Virtual Reality' can be regarded as a continuum containing a whole range of environments with different levels of immersion and use of real world affordances to support and facilitate the third transformation.

The four examples here show, from left to right, map products where:

- The interface takes advantage of interactive cartographic techniques where the user can interact with the data to vary the second transform
- The interface uses some real world spatial affordances
- The interface relies upon real world spatial affordances and takes advantage of a strong sense of immersion
- The user is fully and physically immersed in the model and responds as if operating in the real world

By implementing this design software not only satisfies a direct application, but additionally addresses issues that are of current consequence in the research areas of cartographic visualization and graphical statistics. The aims are thus twofold. An application-based aim is the production of a beneficial and effective mechanism for enhancing fieldwork that is simple to set up and use for a variety of tasks in a number of geographic areas, thus applying a new technology to yield a transferable, usable and popular resource. The research objectives are to demonstrate how
the approach introduced here can be extended to utilise new forms of representation and incorporate novel data types, and applied successfully to a genuine map use and set of map users.

6.1.2 Method of implementation

Realistic digital representations of geographic areas are an attractive proposition for numerous tasks, both within research and education. A prevalent approach to such virtual environments is the use of continuous information such as a digital elevation model (DEM) to produce navigable surfaces that encourage familiarisation with large-scale features. These can be used extremely successfully to provide an overview of topography. A lack of spatial resolution however often means that the information rich distant is more appealing than the detail lacking local. This is particularly problematic when ‘observation’ is an objective whether to aid in learning or to add high-precision context for ESDA. Methods exist to address this problem such as variable resolution triangulated irregular networks (TIN) and level of detail (LOD) algorithms (Reddy, 1998). A particularly exciting development is the view dependent progressive mesh, where the resolution of a TIN changes according to the observers view frustum, and the orientation and distance of points on the surface (Hoppe, 1997). The result is a far better mapping between the screen pixel density and data density across the generated view. At best this removes the problem of low detail at close locations. At worst it ensures that whilst locations that are close to the viewer are shown at the highest resolution of the data model distant and unseen locations are mapped at lower resolution. Speeds at which navigation through and manipulation of the virtual environment can occur are thus increased as is the local detail.

High fidelity models utilising these techniques can involve huge amounts of data and still omit the kinds of local details that make virtual reality such a potentially attractive interface. A complimentary approach takes advantage of a media type that has only become easy to produce relatively recently. It generates the same effect with discrete data items collected at a finite number of point locations by using the increasingly popular medium of 360° panoramic imagery. Panoramic digital photographs can show the view from a chosen location continuously through 360° in the horizontal plane. Such imagery provides a realistic representation in a relatively immersive manner and is an excellent source of current micro, meso and macro scale information (see figure 6.2). It is also relatively easy to collect and geo-reference and so provides the means for high levels of data currency and richness that are currently difficult and expensive to achieve with continuous models. When presented through a spatial interface the power of combining panoramic and planimetric views of spatial information used by Imhof (1957) in his “vue perspective et carte orientée réciproquement” (plates 32, 33) becomes apparent.
Accordingly a panorama-based virtual environment is an attractive proposition, particularly in order to compliment the DEM/TIN surface-based approaches. Despite its current potential to provide higher levels of detail a discrete environment such as this is unlikely to be as immersive as a continuous world in which the user can specify their position and direction of view at any location. The verity of this statement, and sense of 'virtual reality' experienced by an individual using either of these approaches to VR will of course depend upon the quality of data, form of projection and cognition of the user. This reinforces the supposition made above, that any definition of VR must accept that a variety of degrees of immersive interface exist. Figure 6.1 illustrates this schematically where a broad range of potential relationships can be seen to exist between the map user and the representation that they are using. This range will depend upon data collection techniques, the data model used, the mapping, the technology used to perform this mapping and the users perception and cognition. It spans a whole series of immersive and semi-immersive 'virtual environments' that do not contain all of the immersive properties of some VR applications. Software relying upon discrete geo-referenced panoramic imagery can provide a navigable virtual environment by linking between a spatial network of images and maps. Such software is not as immersive as many 'VR' applications, but can achieve the objectives and addresses some of the issues identified above. This is particularly true if the panoramic virtual environment were to be flexible enough to enable other new media types to be mapped and complimented by an analytical component or 'collected data view' with dynamic graphics for exploratory spatial data analysis. This chapter demonstrates that the approach presented here provides a suitable vehicle for developing such a solution.

6.2 Interactive Panoramas in Tcl/Tk

The VFC panoraMap software combines these qualitative views with other media and quantitative information mapped dynamically in a fashion appropriate for visualization map use. It uses the image reading, display and manipulation functions available in Tcl/Tk. Currently '.GIF' and '.PGM' image types can be read into the high-level image format for display and manipulation. Images are created with a command of the form:

```tcl
image create image-type name [option1 value1 ... optionN valueN]
```
To create a '.GIF' a command such as the following would be appropriate. It creates a photo item called 'photol' from the file 'photol.gif':

```
image create photo photol -file photol.gif
```

Once created, images can be used on many of the widgets, including canvases. Tcl commands exist to zoom, sub-sample, and copy and paste sections of images. These actions can be readily linked to Tk widgets to control and display the image manipulation.

To create panoramic imagery that can be panned interactively a 360° image is required. These are assembled using multiple successive photographs from a (digital) camera, which are stitched together with appropriate software. In order to record 360° of panoramic information approximately nine overlapping images are required when using the equivalent of a 28mm lens. The growing availability of high resolution, low cost, digital cameras is thus making the production of panoramic images more popular and an increasing number of packages are available to enable photographers to stitch successive images into seamless 360° panoramas. Some models even advertise the panoramic image as a sufficient reason for purchase (Olympus, 1998; Casio, 1998). A variety of formats and associated panorama players exist. For example, Live Picture Inc. and Apple Computer produce the Live Picture and QuickTime VR formats with which players can zoom, pan and hot-link (Live Picture, 1999; Apple Computer, 1999). Alternatively the VRML 97 specification (The VRML Consortium, 1999) contains a background node through which panoramic images in the horizontal and vertical can be combined to allow panning through 360° in both the vertical and horizontal planes. Whilst commercially available players can be used to tour through areas in which panoramic images have been taken, they do not take advantage of the kind of flexible dynamic interface that the modern map can provide. They are not explicitly geographic and nor are they sufficiently flexible environments to allow modification so that a geographical interface can be developed in order to incorporate these digital media into new forms of digital map.

Plenty of cheap and accessible panorama 'stitching' software exists for both PC and Mac platforms. PhotoVista from Live Picture appears to do a good job and is available online for $60 at the time of writing (Live Picture Corporation, 1999). Figure 6.3 shows the way in which a series of digital images are combined to produce a seamless panorama. Spin Panorama (PictureWorks Technology Inc., 1999) is an alternative that appeared on the free cover disk of a number of UK computing magazines recently and QuickTime VR is also available on the Internet (Apple Computer, 1999). The SkyPaint software (Wasabi Software, 1999) is an example application that combines panoramic imagery with stills taken of the view above and below in order to create the six square images required by the VRML Background Node. It runs as a plugin to popular photo enhancement software. Ongoing research at Microsoft Inc. (Szeliski & Shum, 1997) demonstrates that future products with additional functionality are likely. These programs usually allow the user to stitch
their images into seamless panoramas in standard graphics formats that can be used by programmers to create panoramic image viewers.

Figure 6.3 Creating panoramic imagery using a standard digital camera. (C1)

Top Left - A series of nine images captured using a digital camera. The images show the field of view in 360° at a single location. They were recorded manually by taking a photograph, then aligning the left edge of the view through a camera viewer with the right edge of the previous view, and continuing in a clockwise direction until the starting point was re-photographed.

Top Right - Image stitching using PhotoVista (Live Picture Corporation, 1999)

Bottom - The resultant 360° panoramic image

This is the approach taken here, where a navigable virtual environment using panoramic imagery with analytical capabilities and flexible links to external multimedia players was required. In order to be of widespread use to fieldwork teachers the types of data and associated metadata used needed to be minimal, readily available and easy to collect. The intended user-group included first-year Geography students, and teaching staff with little computing experience and no knowledge of geographic information software. Accuracy and precision were thus less important objectives than the ease with which data could be incorporated into the software and the functionality provided. Demonstration software was developed in Tcl/Tk with multiple (possibly conflicting) objectives. These were:

- an assessment of the technique
- an appraisal of its utility for use in fieldwork
- simplicity - in terms of both the use and production of the virtual environment

To implement an environment that relies upon panoramic imagery, CompuServe '.GIF' images were created as shown in figure 6.3 using PhotoVista (Live Picture Corporation, 1999). These satisfy the stated objective of simplicity and avoid GIS data formats. Tcl/Tk was then used to read
the 360° imagery and create a canvas widget that contains a second image item that is of a suitable width to show a proportion of the panoramic image equating to the current field of view. Cursor interaction with this image can then be bound to a procedure that cuts and pastes appropriate sections of the full image into the one representing the field of view with a 'bind' command. With a series of checks in place to account for the joining of the ends of the panoramic image a continuous panorama is produced. This functionality is documented and demonstrated by the 'simplePanZoom' software illustrated in figure 6.4 and provided in appendix B6 on the accompanying CD.

```
bind fwin(cBot) <Button-1> "panScrollImage ?win(cBot) pHed"
bind fwin(cBot) <BI-Motion> "panScrollImage ?win(cBot) pHed"

The offset is retained, so that next time image cropping occurs it can be added to position and the image rotated continuously.

An additional procedure, 'keepScrolling', includes a loop that repeats the image copying and pasting process and increments the location of the cropping within the image by the most recent amount scrolled, to continue panning.

bind fwin(cBot) <Shift-Button-1> "keepScrolling fwin(cBot) pHed"
```

Additional checks are required to copy two sections of the image when the ends of the panoramic image are joined.

Figure 6.4 'simplePanZoom' - Panoramic image viewing in Tcl/Tk.

Top Left - The main window shows the 1:8 sub-sampled panoramic image (top) and a section of the 1:2 sub-sampled image (bottom). When the bottom image is dragged with the cursor a new section of the 1:2 sub-sampled image is revealed to 'pan' the view. The section of the image that is revealed is displayed with the orange lines showing the edges of the field of view on the 1:8 scale image.

Top Right - Clicking on the 1:2 scale image reveals an appropriate section of the original image in a separate window, permitting 'zooming'.

Bottom - Tcl/Tk scripts outlining the use of the 'image' object and 'bind' commands in this way are provided within the software that is located in appendix B6.
The software allows the user to load a panoramic image, which is then subsampled twice to provide a representation of the full image at 1:8 scale and a section of the image at 1:2. The larger image can be dragged to the left and/or right with the left mouse button and set spinning by clicking with the primary mouse button and holding the 'Shift' key. The subsampled image above the main view in figure 6.4 shows the section of the full image that is being copied and pasted into the main image. Appropriate sections of the full image can be viewed by clicking a location on the larger image with the secondary mouse button. This combination effectively provides zoom and pan functionality. The series of higher level commands required to produce interactive seamless panoramas in Tcl/Tk are outlined by the software. These consist of image reading, sampling and copying functionality and bindings of the type that have been documented earlier in this volume (see section 2.4.2). They can be viewed by clicking the 'Code' menu on the program and selecting an option for further details.

6.3 A Navigable Virtual Environment in Tcl/Tk

By adding links from this kind of interactive panorama to other panoramas and displaying the locations of all images on a map a spatial and navigable environment can be produced. Links between the image and a map are achieved in 'panoraMap' by recording the location of the image, the initial bearing and the angle of view displayed in the image. An appropriate transient symbol can be produced on a map to show the angular extent of the view shown whenever it is changed. Rectangular GIF images of the sort that can be downloaded from the Internet or cropped from a screen dump form the basis of the map views. These must show planimetric information such as a thematic map or rectified air photograph and have known extents to the north, south, east and west so that locations within the region displayed on the map can be calculated. Panoramas are of the same format and require meta-information about their locations, so that they can be mapped, the bearing of the left-hand side of the image and the field of view covered by the full panorama (up to 360°). The position and direction of view can then be calculated for any section of the panoramic image.

These two types of image, and four pieces of metadata for each type are sufficient to determine the angle of view at any position on a panoramic image and to calculate bearings between all points at which images are available. The direction in which one is 'looking' with a panoramic image can thus be displayed on a map and the relative directions of all other images can be shown on an panorama displaying the view from a single point. A navigable interface that maps panoramas can thus be developed.

Consider two locations, represented by the points p1 and p2, with co-ordinates (x1,y1) and (x2,y2) as shown in figure 6.5. A panoramic image I1 taken from point p1 has field of view f1 and length
in pixels $L_1$. The bearing of the left-hand side of the image in degrees from north is $b_1$. If $v$ is an appropriate horizontal section of $I_1$ with length $L_v$ then the angle of view displayed in the section can be determined as $f_v = f_1 \times L_v / L_1$.

Sections of a panoramic image, such as $V$, can be displayed in a viewer and arrows produced on a map at the point relating to $p_1$ to illustrate the extent of the view shown in the image section. If the left-hand side of image $I_1$ is shown in a viewer then the angle of an arrow representing the bearing of left-hand side of view will be $b_1$ and the angle of the arrow displaying the right-hand side $b_1 + f_v$. When other sections of the image $I_1$ are selected for display in the viewer the angle of the arrow representing the left of the image will be $b_1 + (f_1 \times L_v / L_1)$, where $L_v$ represents the number of pixels along $I_1$ at which the left hand side of the image section shown in $v$ begins. Once again the angle of the arrow displaying the right hand side of the view is $b_1 + f_v$.

![Figure 6.5](image.png)

**Figure 6.5** Relating locations on the ground to positions on geo-referenced panoramic images.

The initial bearing, field of view, and length of a panoramic image and corresponding relationships between bearings from the point at which the image was taken and horizontal positions on the image are shown. The calculation of the relative bearing of another location from the specified point is also illustrated. A full explanation of the figure and calculations undertaken is provided in the text.

The process of selecting a section of the panoramic image to view, $V$, and representing the angle of the view with arrow symbols can be programmed to occur interactively when the cursor is moved to the left or right in a viewer. If the calculations are made every time the section of an image shown in a viewer is changed then the symbols on the map displaying the current view can be updated as the image is panned. Continual mouse movement and update of image sections and symbols results in a draggable or spinning panoramic view. The reverse process can be programmed too, so that dragging the symbols on the map with the mouse changes the section of the image shown in the viewer. In a 360-degree view a check is needed to paste from both ends of the image when the viewer contains the direction of the bearing at which the panoramic image...
is split. This functionality is demonstrated in the 'simplePanZoom' software provided in appendix B6 with sample code that formally describes the process using the scripting language.

Showing the directions and distances of a number of panoramic images that are linked to a map in this way provides the ability to navigate across the virtual space and between recognised features. This is particularly so when the locations of other panoramic images are represented by hot-linked symbols within panoramas that display the appropriate image when clicked. These features provide a real sense of spatiality and immersion that are the essence of virtual environments and to achieve them the angles between the locations of all panoramic images must be calculated.

The angle \( \theta_{12} \) between the two points \( p_1 \) and \( p_2 \), in degrees clockwise from north, can be determined using basic trigonometry as shown in figure 6.5. The process is as follows: horizontal distance between points, \( dx = x_2 - x_1 \); vertical distance between points, \( dy = y_2 - y_1 \); distance between points, \( d_{12} = \sqrt{dx^2 + dy^2} \). The angle \( \theta \), representing the angle between the vector from point 1 to point 2 and the vertical, is calculated with \( \theta = \arccos(-\frac{dy}{d_{12}}) \).

Converting this angle into degrees clockwise from North vertical requires action dependent upon the position of \( p_2 \) in relation to point \( p_1 \). The relevant information can be deduced by referring to the distance \( dx \). If \( dx \) is negative then \( p_2 \) lies to the left of \( p_1 \) and so angle \( \theta_{12} \), the bearing of \( p_2 \) from \( p_1 \), is more than 180 degrees, and so \( \theta_{12} = 180 + \theta \). If however \( dx \) is positive and \( p_2 \) lies to the right of \( p_1 \) then \( \theta_{12} = 180 - \theta \). The bearing \( \theta_{21} \), from \( p_2 \) to \( p_1 \), is equivalent to \( \theta_{12} + 180 \) if \( \theta_{12} \) is less than 180 degrees, or \( \theta_{12} - 180 \) if \( \theta_{12} \) is greater than 180 degrees.

It follows that \( L_{1,p2} \), the number of pixels along the image \( I_1 \) that relates to the bearing of \( p_2 \) from \( p_1 \), is \( L_1 \star (\theta_{12} - b_1)/f_1 \). In the example shown in figure 6.5, if \( b_1 = 45^\circ \) and \( \theta_{12} = 135^\circ \) then as \( L_1 \star (135-45)/360 \) can be reduced to \( L_1/4 \), then \( L_{1,p2} \) is known to be a quarter of the way along image \( I_1 \) from the left. This is the location at which where any symbol that hotlinks to a panorama representing \( p_2 \) should be placed. When the section \( v \) is displayed in a viewer then if \( 0 < L_{1,p2} < L_{1,v} \) and \( L_{1,p2} - L_{1,v} < L_v \) the symbol should appear in image section at a position \( L_{1,p2} - L_{1,v} \) pixels from the left edge of the viewer. Relating horizontal locations along an image to bearings from a point in this way means that cursor movement in either the map or the image can be linked to appropriate symbolism in the other view. Interactive tools can thus be developed to relate bearings on the ground to the imagery. The configuration, which is easily set up due to the minimal data and metadata used, permits panning around the highly detailed and realistic landscape and touring across it, the essence of a virtual environment (see figure 6.6).

The levels of immersion provided are relatively low and so 'panoraMap' is an application considered to be positioned to the left of the schematic continuum of virtual environments shown in figure 6.1.
Digital panoramic imagery, maps and associated metadata are integrated by the software to produce a virtual environment that can be navigated by selecting views from a map interface from which additional maps and locations can be selected.

The orange symbols on the map represent the locations of panoramic images, the orange arrows show the fields of view. Orange arrow symbols within the images show the locations of other panoramas. The red arrow on the bottom panorama shows interaction that results in an arrow being produced on the map view, from the location at which the panorama was taken at the bearing specified by the red arrow. Two-way brushing between symbols on the map and the images relate the two types of view, and clicking on either symbol opens the appropriate panorama meaning that a degree of navigation is possible. Multiple backdrop maps can be loaded of different areas and at a variety of scales to permit zooming and panning.

6.4 Incorporating Multimedia and Running External Processes

Once the basic virtual environment has been created the advantages of using a scripting language to develop demonstration software come to the fore. 'Multimedia' is an overused term with a variety of conflicting meanings. Here, the term is used to identify the variety of developing digital data types that might be usefully utilised to represent, or provide, qualitative or quantitative
information of interest to geographers. A number of 'multimedia' data types can be displayed internally in Td/Tk. These include imagery, text and hypertext. The 'exec' command can be used to launch external applications that deal with other media types. This means that the scripting language can be used in its role as 'glue' between software components to produce multimedia maps that contain a loose coupling with other applications. These are easily achievable with Td/Tk and so new media types can be used to extend the representation.

'panoraMap' is extremely flexible in its ability to geo-reference multimedia data files that have been collected in the field for synthesis and analysis. Multimedia information can be geo-referenced interactively and any datum seen/heard/played in a suitable external application. Qualitative digital data are geo-referenced with the software by clicking the dynamic map. This opens a 'File Select' dialog that produces a symbol for any selected media file at the clicked location. Touching the symbol reveals details about the file, and clicking plays it in the external application selected for that media type. A GUI interface allows any executable program residing on a machine to be associate with any selected media data type, determined by its file extension. This linking permits a whole array of possible uses including the annotated 'sketching' functionality described by Fernandes et al. (1997) by linking to text, increasing detail by linking to a more detailed image and 'ground truthing' (Cartwright, 1997) by linking to a WWW page containing a WebCam image. It provides the opportunity to fully geo-reference any digital media residing on a machine and interact with it using the appropriate software and means that the types of digital information that can be used are limitless. The utility of a whole series of multimedia data types for representing geographic information can be assessed and a degree of future proofing is ensured. In the teaching context effective examples are the use of encoded video dips from lectures, hypertext providing historical information about an area and geo-referenced data in a spreadsheet located at relevant points of interest. In the research context text about the data collection techniques and data quality, reports on the topic of interest, interviews with local experts and data relating to previous studies could be made available. The functionality is illustrated in figure 6.7. The 'panoraMap' software is designed as a Geo-GUI for data input, synthesis and analysis. The data available for each location are stored in a hierarchical structure that is easily replicated. The metadata associated with each datum can be displayed and changed interactively with coordinates generated by clicking on the map and locations changed by dragging map symbols. Multimedia data are selected with familiar file selection devices as illustrated in figure 6.7.

6.5 Visualization for Education

It is postulated here and elsewhere (Dykes et al., 1999) that visualization is a particularly suitable technique to help students generate ideas, synthesise information and understand the spatial
character of collected data which satisfies the identified requirement for analysis and synthesis of information on returning from fieldwork. 'panoraMap' contains features that allow collected data sets to be integrated, displayed spatially and compared with secondary information extremely rapidly in order to return information to students immediately following their work in the field. They can then utilise the collected information while the exercise is still fresh in their minds. It also contains functionality to load a series of alternative backdrop maps and automatically creates menus to list those that are available. Unseen themes such as the geology, vegetation, population density or land-ownership can be used in conjunction with the panoramic images.

Figure 6.7 Extending the forms of representation available to the map-maker and map user.

Multimedia data selection, geo-referencing and display are available in 'panoraMap'.

Here yellow symbols on the map represent added multimedia items that can be played in appropriate software when clicked. A geo-referenced digital image (bottom right) and a hypertext document (top right) are shown. Clicking the map reveals a file selection interface through which multimedia data files can be selected and geo-referenced at the clicked location (top left).

The software supports quantitative data in a number of ways. It takes advantage of many of the analytical techniques utilised in the 'cdv' software (Dykes, 1998) for the cartographic visualization of enumerated data sets, providing dynamic choropleth maps, interactive dot-plots and parallel coordinates plots. Secondary data sources can thus be assessed using the software, data sampling and collection strategies prepared, these strategies executed in the field and the primary and secondary data analysed within a two-dimensional virtual environment on returning from the field.
Figure 6.8 shows how a series of observations were recorded by students using a digital camera in the Teign Valley, South Devon, and then compared with secondary data from the UK census of population.

This link between the virtual environment with its secondary information and the field is a vital one relating to the 'comparison' fieldwork objective and providing users of the software with insight into the first cartographic transformation, that of data collection, which creates a model of the real world (see figure 1.1). Cheap global positioning systems (GPS) receivers with accuracy levels in the order of 50ft provide an excellent means of reinforcing this. 'panoraMap' can be used in conjunction with the Garmin GPS 12XL so that any data collected in the field is marked by a GPS geo-reference known as a 'waypoint'. Collected waypoints can be downloaded from the receiver on returning to the field centre and displayed in 'panoraMap'. Clicking a waypoint will produce a file selection box, and the multimedia file chosen is geo-referenced at the location recorded in the field by the GPS. In addition, waypoints can be used to map quantitative data during fieldwork. A GPS waypoint file, downloaded from the receiver, will create symbols at each recorded location. A simple text file containing a row for each waypoint and columns containing the waypoint name and...
then numerical values representing measured attributes at that location will map the distribution of the data and provide the kinds of visualization functionality described with reference to area data sets in the preceding chapters. By combining these features multimedia information can be collected at each data point and accessed through the software to provide exogenous information at all sample locations. These features are fully illustrated in figure 6.9.

Figure 6.9 'panoraMap' - Visualization map use in a virtual environment.

Here parallel-coordinates plots are used to assess the relationship between recordings of slope and soil moisture taken at a field location on Holne Moor, Dartmoor, Devon. The distributions are compared with secondary information gleaned from a geographical information system. The georeferenced photographs provide additional qualitative information about the environment, discharge and a vegetation transect taken across the stream that was the focus of this study.

Additionally, as GPS receivers log positions automatically, routes taken in the field can be plotted in the virtual environment, along with the times at which locations were visited (see figures 6.9 and 6.10). By combining these routes with the multiple backdrop feature, thematic information can be integrated with field observations and data, providing a powerful tool for adding context to analysis. 'panoraMap' can also be used to plan and digitise suitable routes through an area for data collection prior to leaving for the field. These can be uploaded into a GPS, which is used for navigation. On returning from the field, the recorded routes and collected data can be plotted in the software. Areas that have been recently visited and experienced can be investigated and their
characteristics assessed as described by the available data sources. This clearly addresses the 'comparison' objective identified in the reasoning behind the application.

The data illustrated here concern footpath erosion and were collected and used in a study of environmental management and recreational pressure on Haytor Down, Dartmoor, UK.

Data points are recorded in the field with GPS receivers and loaded into the software. Associated quantitative data are loaded from an ASCII file and mapped by filling symbols at each location with an intensity symbolising the data value. Menus provide access to all attributes stored in the data file. Here the distributions of, and relationships between, vegetation cover, slope and soil moisture are assessed and compared with secondary sources. A parallel-coordinates plot shows a multivariate outlier, highlighted in green on the map. Imagery taken in the field at each data point provides contextual information as do the panoramic virtual environment, geo-referenced imagery and hypertext.

6.6 The Application in Use

A variety of different applications of the software have already been alluded to and illustrated during the explanation of its features. It is, however, worth reporting on some recent tests of the software in fieldwork to demonstrate the success of this form of representation that provides a
virtual environment for visualization. The suitability of the approach towards visualization taken in this thesis is also demonstrated by reporting upon direct applications of the methodology.

6.6.1 Counter-urbanisation in the Teign Valley

'panoraMap' has been used in a number of fieldwork projects that fall broadly under the Human Geography umbrella. First year students from Leicester University undertaking a study of counterurbanisation used 'panoraMap' as a way of visualizing and organising data collected in the Teign Valley, South Devon, in April 1999 as shown in figure 6.8. Groups of students visited a number of villages and conducted interviews and used a digital camera to record observations before accessing data from the 1991 and 1981 UK censuses of population in order to compare the villages, their locations and the socio-economic composition of their residents. The project involved three groups of students visiting two villages on successive days. The interviews and observations formed an attempt to compare the villages and evaluate the effects of counterurbanisation. Evidence of the processes of gentrification and/or geriatrification was sought and the likely causes and effects considered. The software provided students with a rich data source with which to back up their observations and the ability to share primary information collected in the field and so develop a regional impression that extended beyond the two localities that they visited. The project was run over two days giving the students time to collate and analyse their data on the first evening before returning to the field. 'panoraMap' was used as an analytical tool to assess the patterns produced by data provided from the UK census of population, to geo-reference the data collected in the field and to synthesise these two data sources and other secondary information in order to generate hypotheses about the processes taking place in the area under study. Additionally the software was used as a presentational tool with which the students demonstrated their findings to staff.

6.6.2 Environmental management on Haytor Down

An assessment of erosion and tourism management was undertaken using 'panoraMap' by a group of first year Geographers from the University of Plymouth. Their selection of locations using the software prior to leaving for fieldwork and ensuing data-collection at these locations demonstrates the utility of the software and approach in fieldwork, and the utility of visualization in a virtual environment with contextual information.

For example figure 6.10 shows a series of locations at which environmental information about footpath erosion was collected. Different appropriate locations were selected by three groups using the software on the morning of the field trip from the illustrated air photograph and other information about erosion provided by the Dartmoor National Park and presented as geo-referenced hypertext documents. The identified positions were loaded from the software into GPS
receivers and located in the field using the GPS. Whilst at the locations the students took images of
the slope, erosion and vegetation and made measurements of the slope, soil moisture, pH and
percentages of agreed types of land cover. On returning to the classroom the data were loaded
into 'panoraMap', which enabled the entire group to assess the variation across the surveyed area,
synthesise the primary and secondary information and perform preliminary visual analysis (see
figure 6.10). A series of interesting spatial distributions resulted. The effects of management
strategies, identified from geo-referenced imagery showing paths that were being strengthened,
allowed to regenerate and under renovation were discussed, as were issues of data quality and
data collection strategies. A number of initial conclusions about the relationships between slope,
erosion and car park locations were drawn.

6.6.3 Catchment hydrology on Holne Moor

'panoraMap' was used by first year Geography students on non-residential fieldwork at the
University of Plymouth to visualize information about a river catchment on Holne Moor, Dartmoor,
as shown in figure 6.9. A range of information about the hydrology of the catchment was recorded
and geo-referenced with GPS receivers. This included images showing vegetation transects,
discharge measurements at locations along the main stream and values of slope and soil moisture
at 20m intervals along a transect from the stream to the edge of the catchment. Photographs of
the vegetation found at 5m intervals along a transect are shown at the bottom right of figure 6.9.
The shaded circles to the left of the map in figure 6.9 represent the locations at which soil
moisture and slope were recorded. Each group of students was given an initial starting point for
this series of measurements, which was located using the GPS receiver. Each group's readings
were loaded into the software for analysis on return to the University and relationships between
slope, aspect and soil moisture assessed, as shown in figure 6.9. As the day in the field was
repeated a rich qualitative and quantitative data set was amassed that covered the catchment and
which will be of use to geographers from Plymouth and elsewhere in subsequent fieldwork.

Each of these projects used 'panoraMap' as a key element of the fieldwork and demonstrates the
use of the visualization techniques proposed in this volume in a real and relevant application.
Whilst teething problems and the usual issues to do with using information technology in teaching
were bound to surface, each project resulted in positive feedback from staff and students alike and
has evident application. The proof of concept, utility and generality sought by the fieldwork tests
were successfully attained. Some modifications will be necessary, and some lessons were learned,
but in synopsis the software seemed to provide an exciting and enhancing perspective on the
projects with which it was used and achieved the stated objectives.
6.7 An Assessment of the Application

The prototype software presented here demonstrates how effectively a representation that constitutes a navigable and recognisable virtual environment can be created from modern digital media with minimal data and metadata. The software map that results is realistic and inherently spatial and provides an excellent means of synthesising information from a whole range of traditional and novel spatial data types. Visualization map use achieved through the Tcl/Tk approach appears to have been used extremely successfully.

Empirical evidence is provided by the responses of student users who were asked to complete a rudimentary feedback form having used the software as part of their fieldwork. Students indicated that the software provided a focus for the day's work and enhanced the projects where it was used. Encouragingly responses indicated that students saw potential benefit from using the software prior to carrying out the experimental work, both during and prior to the field course, and the notion that it was beneficial to see collected data displayed in the evening returned a strong consensus. Students were asked whether they could see the utility of the software elsewhere in the curriculum and in other fieldwork projects, which they could, and for criticisms and possible additions, which can be addressed. When asked to identify the most positive aspect of the software many of the initial objectives were identified. Comments included:

- "the amount of information"
- "the ability to compare data"
- "seeing what you have achieved that day"

Pleasingly, students responded well to the application of 'visual thinking' paradigm with the following 'most positive aspects' being identified:

- "enables us to see patterns and changes in relationships very well"
- "simple to use, shows spatial variation"
- "very good representation of information"

The immediacy of loading information that students have recorded during the day into a graphical system on returning from the field certainly appears to stimulate interest. Mapping the routes recorded by GPS and associating them with secondary data was a particular success. This kind of virtual environment seems to enable the user to make direct links between their map image and the real world. By enabling students to interact with their field data immediately they can process their information and integrate primary and secondary data sources whilst making use of the geographic context that is fresh in their minds. Comparisons are readily made between the two, satisfying an identified fieldwork objective, and the kudos granted to digitally mapped primary data encouraged students to question the validity of the secondary data and consider their origins.

Students using the software also appeared to make more of their time and efforts in the field, and
to appreciate the fact that they could combine the group's data as a whole. This resulted in the production of a variety of organisational structures to ensure efficient data collection and quality control within groups.

An additional factor of considerable value is that the environment can be set up with relatively little effort, an enormous benefit for teachers. The panoramic images are collectable in a day in the field and the maps easily screen dumped from geographic information software or a web browser. The figures presented here are a little cluttered, and as 'panoraMap' is a dynamic piece of mapping software the only way to really appreciate the nature and functionality of the software is to use the copy provided in appendix B7. The software is sufficiently flexible that it will detect folders containing maps, panoramas and metadata when added, meaning that the opportunity exists for users to load and assess their own virtual environment and analyse their own data.

The flexibility that the Tcl/Tk language provides was important in allowing the software to be modified and extended to suit particular projects. By adding required analytical functionality to the virtual environment as it was needed a whole series of forms of analysis became possible. In essence this is the ultimate in student led exercises as students can request functionality as they need it if the time and resources exist. This method of analysis is a close approximation of the graphical scripting approach to visualization and provides a useful case study of the visualization approach to analysis.

### 6.8 The Potential of the Approach

The 'panoraMap' software demonstrates that the approach taken here has real application and can provide a flexible environment for the development of new techniques and incorporation of novel data types in which issues of representation can be assessed. Whilst the positive experience with and feedback from the users of 'panoraMap' is pleasing and confirms many of the arguments put forward above, the virtual environment and multimedia representations incorporated into the software have potential use far beyond that addressed here. Possibilities exist for adding contextual or exogenous information to more rigorous analyses. Such information is being identified as having increasing significance in the process of analysis and knowledge acquisition (Cleveland, 1998). These data might originate from a number of developing sources and useful meta-information for analyses that take advantage of statistical graphics could come in the form of video, imagery or hypertext. Such sources can provide important context to analyses of the kinds of complex phenomena studied by spatial scientists (Unwin, 1999). Immersing oneself in a virtual representation of the environment under study and assessing qualitative and quantitative information about it whilst performing rigorous visual and statistical analysis has great potential. It
enables researchers to make the transition back through the cartographic transformations that have taken place in order to provide them with mapped information for analysis, and to embed their research in real-world context. For example, Unwin and Unwin (1998) assert that stationarity cannot be assumed to occur in any process operating over real geographical spaces due to likely regional variation in characteristics that are favourable for that process. Providing additional exogenous information in abstract form and contextual information through a realistic virtual environment can help orientate the researcher and provide them with insight into the characteristics and operation of the part of the real world that they are studying.

Figure 6.11 Interactive techniques for assessing the spatial nature of multivariate distributions.

Visualization of qualitative and quantitative information concerning footpath erosion used in a study of environmental management and recreational pressure on Haytor Down, Dartmoor, UK.

Interactive techniques for ESDA can be incorporated into this kind of virtual environment. The shading of lines on the interactive parallel plot (bottom centre) corresponds to the distance of each case from a probe that can be moved around the map (top left). A spatial cluster of cases with a similar multivariate signature is evident as the darkest lines, those closest to the location of the probe shown by the position of the cursor, follow a similar pattern. These cases occur around the ‘Haytor Rocks’ feature that can be seen by clicking orange symbols on the map that link to panoramic images (bottom right). ‘panoraMap’ enables users to assess exogenous information such as the imagery and hypertext (top right) provided here in order to explain the distribution and assess the processes that might be taking place. An additional image, showing the way in which the shaded parallel plot varies, is provided in appendix C1.
Additionally more advanced techniques for visual ESDA can be incorporated into the software and/or developed. For example, functionality demonstrated in section 4.3 for symbolising the proximity of cases to a spatial probe on statistical views of data was added to 'panoraMap' for exploratory purposes. The results are shown in figure 6.11 where the grey shading of lines in the parallel-coordinates plot corresponds to the distance of each case from a probe on the map. A spatial cluster of cases with a similar multivariate signature is evident. These cases occur around the 'Haytor Rocks' feature (figure 6.11, bottom right). 'panoraMap' enables users to evaluate additional contextual information in order to explain the distribution and assess the processes that might be taking place. In the example shown in figure 6.11 this additional information is in the form of imagery, hypertext and maps showing data collected by the Dartmoor National Park which have been loaded into the software.

The type of synthesis of geo-referenced secondary data and qualitative data that take advantage of new digital media that 'panoraMap' supports may provide a means of resolving some of the issues concerned with analysing geographic data. At worst it can enable us to address them by incorporating information about some of the unknowns and spur us on towards the search for better analytical tools identified by Unwin and Unwin (1998). Such techniques certainly have huge potential for use in more rigorous exploratory spatial data analysis than that undertaken by students undertaking fieldwork as 'panoraMap' demonstrates at a number of levels. At a generic level the application confirms the utility of visualization and uses novel digital media to provide a unique interface to contextual information. At a more specific level the implementation presents the kind of functionality that a virtual environment for ESDA might contain. In terms of the approach it illustrates that Td/Tk is a convenient implementation strategy that provides a good vehicle for developing cartographic visualization applications and is sufficiently flexible to be able to provide potential solutions to real problems.
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

7. Discussion, Context and Consequences

Jason Dykes
Department of Geography
University of Leicester
UK
7. **Discussion, Context and Consequences**

7.1 **Results, Outcomes and Significance**

The work described in this thesis has been presented in a relatively linear way with an introduction defining the need for a particular approach to dynamic cartography for visualization, which was described and then used to implement working software. The software and environment were then in turn applied both for the achievement of, and research into, cartographic visualization. The relationships between these elements of the research were represented schematically in figure 1.6. The methods used subsequently to address these aspects of cartographic visualization are shown in figure 7.1, which uses the graphical structure introduced in chapter 1.

![Diagram showing the relationships between approach, implementation, and application in cartographic visualization.](image)

Figure 7.1 Research undertaken – Cartographic visualization has been addressed at three 'levels': Approach, implementation and application.

The methods used to address these elements of cartographic visualization are shown.

Whilst there are strong and evident links between the approach, implementation and application of the research they can also be considered and assessed independently. Visualization can evidently be successfully achieved without the software, environment or approach introduced here. Alternative means of producing smart maps and undertaking geographical scripting exist too. The real 'results' of the research presented here are thus the implementations. They are the embodiment of the approach, and have been widely used to substantiate that both it and cartographic visualization as a whole are appropriate. By applying them geographic analysis can be
undertaken and evidence presented to support the argument made in the introduction that visualization is a useful form of ESDA. Examples of some of the issues associated with this method of exploratory map use can also be addressed and novel techniques for achieving it outlined. Each of these aspects of visualization research has been featured in the preceding chapters using the implementations.

Confirmation that these results and examples are valid is implicit in the empirical evidence provided. Recognition of their general significance however is found not only in the ability to apply the techniques but also in the more widespread acceptance and use of the implementations, the approach that underlies them and their application within the academy.

Each of these three stages is inter-linked as demonstrated by the structure in which this thesis has been presented. For example, 'panoraMap' has achieved tangible accomplishments at each stage. It is an example of application not only because it is shown to permit cartographic visualization which has been used successfully as a technique for data synthesis and analysis at three UK Universities, but also because the open-ended approach utilised can accommodate novel media, allowing research into visualization. As it is presented as a working piece of software it is evidently an implementation – a demonstration with appropriate example data is available to the community on the Internet. 'cdv' falls into multiple categories in similar ways. The implementation demonstrates that the approach is suitable for visualization of a certain type of data model. The functionality is applied to address research issues in visualization and the scripting language is used to demonstrate how a flexible geo-graphical environment for visualization can be utilised. In effect the structure used in this thesis relies on the implementations to show that the approach is useful and how it can be applied.

Despite these dependencies, it is possible to detect that the research is significant in each of these three areas by recognising:

- Use of the approach for dynamic mapping
- Use of the implementations for visualization, demonstration and education
- Development of the applications for further visualization

Overall these factors substantiate the significance of the research presented and confirm the proposition made in chapter 1 that visualization is a useful form of map use.

7.1.1 Use of the approach for dynamic mapping

Tc/Tk has been used as a means for the high-level description of dynamic cartography for visualization by over eighty graduate spatial information scientists at Leicester University. The scripting language has been taught with an emphasis on its potential for cartography, visualization and spatial information modelling and processing for four years. A number of M.Sc. theses have
successfully addressed issues in Geographical Information Science using the language. Mapping has been undertaken using Td/Tk in the wider domain also. Examples include the Historical Atlas of Industrialising Britain (Southall & White, 1998) and the Apoala Project (Pennsylvania State University, 1998). The former of these consists of a major effort to integrate historical information from a number of sources and present it as an integrated dynamic atlas. The Td/Tk scripting language has been used to produce interactive maps that exhibit observer-related behaviours and can be accessed by researchers. The atlas consists of 31 chapters containing interactive maps and historical text from a wide range of sources. The latter is a project based at Penn State University that is aiming to develop an integrated spatio-temporal geographic information, visualization, and analysis system with the goal of supporting complex exploratory analysis of environmental data (Pennsylvania State University, 1998). It uses Td/Tk to implement a dynamic parallel coordinates plot that serves as an interface tool for exploratory analysis. The Historical Atlas Project uses dynamic maps that are suitable for communicative purposes, whilst the Apoala Project is implementing observer-related behaviour in Td/Tk for more exploratory map use.

7.1.2 Use of the implementations for visualization, demonstration and education

The 'cdv' demonstrator software has been used in a number of modes. It has been the focal part of presentations of cartographic visualization at a large number of meetings and to a range of bodies in institutions across the World. These include the International Cartographic Association (ICA), the Association for Geographic Information (AGI), EuroStat (New Techniques and Technologies for Statistics), the American Statistical Association (ASA), Eurographics UK, the British Cartographic Society (BCS), the National Centre for Geographic Information and Analysis (NCGIA), the Computing Section of the Royal Statistical Society (RSS) and representatives of the Office of National Statistics (ONS) and the Rutherford Appleton Laboratory (RAL). The software has also been used to demonstrate ideas and visualization map use. In a published review of the software the Royal Statistical Society described it as 'technically dramatic' (Swan, 1999). In 1995 'cdv' and the techniques for visualization that it contains were demonstrated to representatives from the UK Office of National Statistics (ONS). The ONS subsequently released a section of the UK census of population for 1991 into the public domain, specifically for use with 'cdv', under the impression that the analytical capabilities would lead to more use being made of the census with the development and popularisation of visualization techniques. The ED-Line consortium were similarly impressed with the functionality and potential and allowed a generalised set of boundaries to be released into the public domain as a demonstrator of their data holding. These generous and unprecedented actions meant that through the functionality of 'cdv' a pertinent spatial data set was made available to the academy for visualization map use in research and teaching in the UK. The software and data are available to the UK Academic Community through the Manchester Information Systems and Associated Data Sets service (MIDAS, 1996).
In addition to this static repository for software download the Joint Information Systems Committee (JISC) saw fit to fund a further project to support ‘cdv’ and build upon the demonstrated potential. This is a further and considerable measure of the success enjoyed by the software and the approach that it embodies. The JANUS Visualization Gateway Project (The Manchester Metropolitan University, 1999) has appointed a National Census Visualization Officer to assess and promote the techniques used in ‘cdv’ in order to support visualization of the 1991 and 2001 censuses. The project has a number of objectives that support ‘cdv’ and the kind of visualization it performs. These include housing a version of ‘cdv’ on the MIDAS mainframe for remote data visualization, producing documentation of the software and techniques and providing training courses at a national scale to publicise the ideas to the academy and commercial concerns. In addition a national generalised geometry has been produced of ward boundaries at county level, specifically for the kind of map use advocated here and particularly with the software implementation presented in chapter 3. The provision of a national cartogram base at county level is due to follow.

This promotion has resulted in the software being used in research and teaching programs at a number of institutions. These include the aforementioned use of the ‘cdv’ application to analyse spatial information about tourist behaviour at the University of Wageningen, the Netherlands, and the use of ‘cdv’ to teach visualization and census analysis at the Universities of Keele, Leeds and Manchester as well as Leicester. The Centre for Tourism and Recreation Studies in Wageningen describe their successful utilisation of the software for visualization as follows:

"the first phase [of the research] focuses on analysing the spatial structure and the connection between different variables and time space behaviour. This connection can be traced by visual means... Some hypotheses on tourist behaviour and the use of information and environment can be developed. Also an impression on possible tourist recreation complexes might be formulated" (van der Knaap, 1997).

The techniques have been further developed for exploratory analysis of network complexes within the Centre (Hillman, 1995). Additional data sets for other locations, such as the Valls-Rijn, with different attribute sets have been analysed using the software.

North American users have taken advantage of the software too, with key academics from a number of fields incorporating ‘cdv’ into their teaching. These include Luc Anselin (University of West Virginia), Fraser Taylor (ex-Chair of the ICA at Carleton University) and Terry Slocum (EMAP designer, visualization software expert, University of Kansas). In the commercial and private domains the Office of National Statistics use ‘cdv’ for demonstration purposes in their ‘roadshows’ and County and District Councils in Shropshire and on Merseyside have been provided with copies having requested the software for analytical purposes. ‘cdv’ is also included and described in "The
Census Data System’ a collated volume detailing suitable resources for use with the 2001 census (Rees, Martin & Williamson, in press). A ‘panoraMap’ demonstrator was also included in a recent volume that used text and a CD to collate, assess and demonstrate the state of the art in multimedia cartography across the globe (Cartwright, Petersen & Gartner, 1999).

7.1.3 Development of the applications for additional visualization

Other notable outcomes of the research include projects that aim to take advantage of the functionality and implementation by scripting extensions to the ‘cdv’ software so that the approach can be applied to other research areas and data sets.

For example, the JANUS project is producing an interactive Internet-based ESDA browser that will provide an exploratory and interactive map-based interface to the 2001 census specifically for visualization (Carter & Dykes, 1998).

The award of £87,000 to the School for Policy Studies by the ESRC to undertake a spatial analysis of crime and criminal justice in England and Wales is another notable success. The research aims to examine and interpret the spatial relationships between crime, victimisation and punishment over recent years in England and Wales. The data will be initially analyzed using ‘cdv’ and then multi-level and logistic modelling will be employed to assess patterns that are detected. The techniques available in cdv are described in the research proposal as being "ideal for exploring and analysing the multiple spatial associations in English and Welsh victimisation and crime data" (Hillyard, Gordon & Pantazis, 1999). The proposed customisation of ‘cdv’ to reveal information from the multivariate spatio-temporal data sets that are being analysed will constitute a significant test for the software and approach.

7.2 Assessment of the Approach

The approach, implementations and their application have evidently met with a degree of success, at a number of levels. The scripting language identified is a suitable strategy that underlies the implementations and their application to undertake, and perform research into, cartographic visualization. Having outlined its main advantages and the way in which the scripting language and the software environments that can be achieved with it, it is worth assessing the limitations, flexibility, future and possibilities of this particular vehicle for cartographic visualization. This allows us to consider whether and how the approach can be further formalised and how the future of visualization might fit in with it.
7.2.1 An environment for software development

Learning the syntax required to write Tcl/Tk scripts and developing prototype software is a rapid process compared with personal experience of using other interface building software and lower level programming languages. Maps with explicitly defined symbolism, observer-related behaviours and elaborate GeoGUIs with a speedy interface and impressive functionality are readily achievable as demonstrated in previous chapters. The smart symbolism, scrolling and scaling provided by the Tk canvas are extremely appropriate for mapping and providing zoom and pan functions, meaning that basic dynamic map functionality is almost a default of any symbolism.

The development environment is excellent too, with clear and consistent debugging messages speeding up the development process and commands for tracking changes to variables and trapping and reporting upon errors.

A major advantage for higher level programmers is that the underlying scripting language is developed independently from the applications that use it. Tcl/Tk users have experienced this advantage over the last few years as the language was taken on by Sun Microsystems between 1994 and 1998 when substantial development occurred. This included ports from X-windows to a number of platforms, including Macintosh and Microsoft Windows™, the production of a browser plugin and a byte-code compiler. The cross-platform nature of the scripting language means that the approach advocated here has wide applicability and is extremely useful for demonstration purposes, as installation is possible on a wide range of combinations of machine and operating system. The plugin means that certain Tcl/Tk commands can be issued and run in scripts over the World Wide Web. The byte-code compiler provides speeds approximating those of applications compiled from systems programming languages through a flexible interpreter. On its release the speed at which the 'cdv' software operated doubled overnight with no additional work from the software developer. This improved response speeds and increased the interactivity that could be achieved. Moreover the complexity of the maps that could be satisfactorily visualized and the observer-related behaviours that could be associated with each map and symbol was extended.

7.2.2 Limitations and drawbacks

Some limitations and drawbacks associated with using the approach for visualization have been identified previously. Whilst the flexibility of the language has been advocated as a major advantage for visualization, it can cause problems from a software development perspective, and the lack of functionality that addresses geographic concerns specifically is an additional disadvantage. The relationship between specific product and generic flexible visualization solution is a difficult one to achieve and manage across visualization. With such a flexible, high-level approach as that promoted here the relationship between quickly developed prototypes and robust
software for use by a mass audience to demonstrate and evangelise was difficult to manage in practice. The former of these address specific issues and require expert knowledge and scripting skills to run them from the command line and enable user-led ideation where the view or technique required to extract information is not known at the onset of the research. The latter perform a communicative objective by providing sample visualization functionality to a novice, or less, expert audience. These modes of operation are represented by the ‘implementation’ level of figures 1.6, 1.7 and 7.1 and are analogous to the joint communication/visualization objectives of static maps. Difficulties in managing the relationship between these roles are always likely where an environment is required to both demonstrate and perform visualization.

Additionally, whilst Td/Tk is a powerful medium for defining two-dimensional dynamic cartography with strengths for linking GeoGUI objects and applications, a number of limitations to the cartographic functionality and inconsistencies with cartographic practice exist. For example, whilst the language has impressive capability for discrete symbolism in two dimensions and interaction with symbols, continuous information cannot be easily represented graphically and three-dimensional functionality is omitted entirely. The interfaces have a native look and feel that determines the aesthetics over which the scripter has limited control, and some cartographers may feel bound by the ‘Paint Brush’ type vector graphics. It has already been noted that the ‘focus’ and ‘orientation’ visual variables are not supported and dynamic visual variables are difficult to control effectively. The imagery functions are useful but the types of image that can be read and the ways in which images can be transformed and manipulated could be improved, as could the range and types of multimedia object that can be read and manipulated directly. Td/Tk cannot, for example, match the multimedia functionality of recent developments in Java such as the Java 3D API (Sun Microsystems Inc., 1999) or the Java Media Framework (Sun Microsystems Inc., 1999).

A direct limitation is related to the fact that the speed at which dynamic map behaviours occur is greatly reduced as symbol numbers and complexities are increased. The actual limitations depend upon the hardware and software combination used and the response time required.

### 7.2.3 Map complexity and map dynamism

The trade-off typical in scripting languages between simplicity, clarity and processing speeds does have an effect on the data magnitudes that can be incorporated into Td/Tk applications and these are assessed here. Successive releases of the interpreter and improvements in the processing speed of hardware have improved the speed at which the graphical commands operate during the period of research. Using a Td/Tk compiler to run binary code rather than interpreted scripts can increase the speeds at which commands are processed. Visualizers then lose the openness, flexibility and interaction provided by a command line through which the software can be modified and interacted with in real time. And the most recent versions of the language use a byte-code
compiler to reduce this difference to such a degree that it is barely significant. This section assesses the speed with which Tcl/Tk manages graphics on a number of systems and so assesses the opportunities for the visualization of large data sets.

### Figure 7.2

Time taken to produce maps for a single platform.

The graph shows the speed at which maps were produced on a single platform, the SGI Indy running Tcl/Tk 7.3/3.6 under Irix 5.1.1 that was used in 1994. Maps were produced with different number of symbols and a range of symbol complexities. The vertical red lines show 1, 2 and 5 second time periods that might be considered the bounds of real-time dynamism. The case highlighted in orange has been brushed, data details for this case are shown at the bottom right of the graph. All other cases from this configuration with the same number of symbols are highlighted with a black outline, all other cases from this configuration with the same symbol type are highlighted with an orange outline.

In order to assess the levels of dynamism that can be achieved with Tcl/Tk under a variety of conditions software was developed to record the time taken to produce maps with a range of different levels of complexity. The software takes advantage of the Tcl 'time' command that returns the time taken to execute a command or procedure. A number of versions of the Tcl/Tk core were used along with a series of platforms with varying operating systems, processors and memory configurations. These were selected relatively arbitrarily, but reflect the range of platforms used during the completion of the research presented here. Each combination of platform and Tcl/Tk version was used to create maps of $10^1$ through to $10^4$ items. Four types of map were produced with symbols of different complexity. In order of complexity, rectangles, 5-point polygons, 10-point polygons and 25-point polygons were plotted. The process was undertaken five times for each
combination of map size and symbol complexity on each selected platform. The experiment aimed to demonstrate the changing nature of the field over the course of the research undertaken here and enables an assessment of likely future trends to be made. The data that resulted from these tests can be analysed using dynamic graphing software provided in appendix B8 on the accompanying compact disk.

Some results are shown in figure 7.2, which plots the number of symbols against time for a single platform, the SGI Indy configuration that was used in 1994. This involved running Td/Tk 3.6 on the Indy under irix 5.1.1.

The linear nature of the relationship is strikingly evident, with the time taken to create a map directly proportional to the number of symbols created. This emphasises the trade-off between map complexity and levels of interactivity that make principles of graphic design and high data:ink ratios so important in dynamic cartography. The four levels of symbol complexity are represented by the four parallel lines of points. A point is brushed in figure 7.2, representing the map with 1000 rectangles (the full details are provided at the bottom right of the graph). The median time taken to produce this particular map under the Td/Tk 7.3/3.6 and SGI Irix 5.1.1 configuration was 5.384 seconds. All other points that represent the same map type and map size for this symbol are highlighted: the 1000 symbol maps with a black outline, the rectangle maps with an orange symbol outline. The linear relationship between number of symbols and time taken to draw the map is apparent, as is the fact that more complex shapes, with greater numbers of points, take more time to map. Visualization relies upon map behaviours that respond quickly to the user to provide graphical interactivity that occurs in line with the researcher's train of thought. There is no specified limit to the rapidity of response that is required: a satisfactory time will depend upon the way in which the user operates, but essentially the quicker the better. Unwin (1999) notes that interaction in an EDA context requires 'immediacy' of response as if a result or feedback does not appear immediately the train of thought is broken. The red lines on the graph represent arbitrary times of 1, 2 and 5 seconds which might be considered satisfactory periods of time to wait for data views by certain dynamic map users for visualization. It can be seen from figure 7.2 that with the computing platform available at the onset of this research, the most generous of these permitted a map containing 1000 rectangles or 200 25-point polygons, whilst the most stringent permitted a figure in the order of 200 rectangles or 40 25-point polygons.

Things have moved on however, as figure 7.3 demonstrates. The graphed data shows the times taken to produce between ten and ten-thousand rectangles for five hardware/software configurations. These are as follows:

- Td/Tk 7.3/3.6 - SGI Indy, Irix 5.1.1
- Td/Tk 8.0 - SGI Indy, Irix 5.1.1
The upper horizontal red line representing five seconds shows that under the initial configuration, represented by the lilac points, a map containing 1,000 rectangle items could be produced in five seconds and the highlighted case shows that a 10,000-rectangle map took over a minute to draw. The series of light green points show that a PC with relatively standard specification can currently draw 10,000 in just over three seconds and be expected to produce up to 20,000 rectangles within the arbitrary five second limit. This is a twenty-fold increase in speed, and thus a twenty-fold increase in the complexity of map that can be considered dynamic and used for visualization.

Figure 7.3 Comparison of times taken to produce maps on variety of platforms. (C1)

The graph shows the times taken to produce maps with between 10 and 10,000 rectangles for a variety of platforms and versions of the Tcl/Tk software that were used as research progressed. The hardware/software configurations are, from slowest (first) to fastest (last) Tcl/Tk 7.3/3.6 - SGI Indy, Irix 5.1.1; Tcl/Tk 8.0 - SGI Indy, Irix 5.1.1; Tcl/Tk 8.0 - SGI Mainframe, Irix 6.5; Tcl/Tk 8.0 - P200, 64MB, Windows 95; Tcl/Tk 8.0 - PII400, 256MB, Windows NT. The vertical red lines show 1, 2 and 5 second time periods that might be considered the bounds of real-time dynamism. The graph shows that dynamic mapping speeds have increased by a factor of between ten and twenty during the period of research. For example, the case highlighted in orange has been brushed, and the data details shown at the bottom right of the graph reveal that under the initial configuration a 10,000 rectangle map took over a minute to produce. The green points show that the same map takes three seconds to produce on currently available machines.
When the current and initial configurations are compared for each of the symbol types it is evident that current computers outperform the visualization workstations of five years ago so totally that even maps comprising of the most complex form of symbols used here can be produced more quickly on modern machines than the simplest maps could on their predecessors (see figure 7.4). Indeed response times vary by an order of magnitude. Currently a PC running the Tcl/Tk scripting language can expect to produce 20,000 rectangles in a 5 second response time (an array of 140x140), 3600 polygons with 10 vertices (a 600x600 symbol map), or 2,000 polygons with 25 vertices (a 44x44 symbol map).

If this ten-fold increase in computing speed that has occurred in five years continues we can expect to be able to produce even more complex interactive maps. In 1995 Tcl/Tk 7.3/3.6 running on an SGI Indy visualization workstation produced 10,000 25-point polygons in 284 seconds and the same number of rectangles in 60 seconds. These graphics can be produced on a 400MHz Pentium with 256MB RAM running Tcl/Tk 8.2 for Windows in 29.7 seconds and 3.2 seconds.
respectively. Indeed the NT machine can produce 100,000 25-point polygons in 325 seconds and a million rectangles in just 369 seconds. Figure 7.5 graphs these numbers.

Figure 7.5 Assessing future trends - Speeds at which maps are produced on a PII 400.

The graph shows the times taken to produce maps with between 10 and 10,000 symbols with a variety of symbol complexities for Tcl/Tk 8.0 on a Pentium II 400 with 256 MB RAM running Windows NT. The horizontal pink lines show 10, 20 and 50 second time periods. If a 10-fold reduction in map response times occurs then these lines can be used to show the type of map that would be possible within 1, 2 and 5 seconds which could be considered the bounds of real-time dynamism. The highlighted zone shows that 40,000 10-point polygon can currently be created in 50 seconds, equivalent to the supposed future 5 second limit. The symbols to the right of the graph show that maps consisting of 100,000 rectangles are possible with the current configuration.

This leads us to guess what might occur in the near and distant futures. If the assumption is made that speeds continue to increase at the rates noted over the past five years in the next five years then the types of dynamic map that will be possible through scripting will be much more complex than those presented here. The kinds of mapping that currently take in the order of 50 seconds might be achievable in five seconds in the near future. The pink horizontal lines on figure 7.5 show current times of 10, 20 and 50 seconds. The types of operation that are currently achievable in under 50 seconds and would be possible to produce in a dynamic scripting environment in the near future if technological improvements progress at a constant rate include 200,000 rectangle maps (450*450) rasters, 60,000 5-point polygons, 35,000 10-point polygons and 20,000 25-point polygons. If speeds were to accelerate by this amount in two successive five-year periods then dynamic maps containing huge numbers of symbols might be feasible. Using the evidence collected
here it is suggested that dynamic maps containing 1,500,000 rectangles, 600,000 5-point polygons, 350,000 10-point polygons and 200,000 25-point polygons might be achievable.

These types of products will evidently require different mapping techniques to those presented here, and may well result in a degree of synthesis between statisticians who are aware that techniques do not scale up (Unwin, 1999), cartographers and remote sensors who have skills in dealing with large spatial data sets. Software for visualizing the data presented here, from which the graphs and figures were derived, is provided in appendix B8.

7.2.4 Future support and development

In addition to these factors the suitability of the approach must be assessed in the context of its likely longevity if it is to be advocated as a method for creating dynamic maps, performing visualization and continuing research. The utility of the Td/Tk scripting language for tasks beyond dynamic cartography is illustrated by the large and growing user community and commercial platform-company. These provide a degree of stability in the rapidly changing world of computing due to the investment of effort and capital into the language.

The foundation of the Scriptics Corporation by John Ousterhout, author of Td/Tk, was a major event in the summer of 1998. The foundation has placed the support and development of Td/Tk in the commercial domain whilst maintaining the source and Td/Tk interpreter and 'wish' application in the public domain. The speed with which successive versions of the interpreter have been released and the breadth of additional functionality that they have contained have been more impressive than ever since Scriptics took over the research, development and support of the language. The interpreter now includes a Td/Tk compiler that creates byte-code on the fly and improves performance significantly. In addition the network of Td programmers appears to have expanded with the customer-base of commercial clients, with financial firms enthusiastic adopters of the technology (Simms, 1999). A number of products and services have been released by Scriptics. The most significant is TdPro, a commercial development aid that has a number of features. These include a compiler that assembles Td/Tk scripts into an operating system-specific executable, such as those presented in appendix B on the accompanying CD. This removes some of the flexibility required by researchers for visualization but increases the likely audience for wrapped map products as the investment in installation is reduced. Researchers can take advantage of the debugger, wrapper and checker provided by the software to improve their scripting and prototyping efficiency. Increasing the speed at which visualizers can respond to their graphical needs makes the environment an even more suitable one for visual thinking. A free evaluation version of Td Pro is available on the Internet at the time of writing (Scriptics Corporation, 1999).
In terms of longevity the future for Tcl/Tk is not assured, as trends in the computing industry are notoriously difficult to predict. However, users of the methodology outlined here can gain confidence from a community numbering between 500,000 and a million Tcl scripters worldwide (Scriptics Corporation, 1999). The source runs on a number of operating systems and so is resistant to any sudden shifts in the technology battles between Microsoft Windows™ and Linux. In addition the Association for Computing Machinery named Dr. Ousterhout as the recipient of the 1997 ACM Software Award for software with a lasting influence. If Tcl/Tk is as enduring as previous winning systems such as UNIX, the WWW, TCP/IP and Postscript then when the evidence provided here for the utility of the language to create dynamic maps is assessed, it must be concluded that investment in this high-level language is worthwhile. Plans for further developments are extremely appropriate and make this assertion particularly true. These include a Tk GUI Builder through which user-specified GUI widgets can be created, better Java integration tools, support for component frameworks and protocols such as ActiveX, Java Beans, DCOM and CORBA and support for additional hardware platforms (Scriptics Corporation, 1999).

7.3 Extending Flexibility and Functionality

Such developments will facilitate the production of functionality and graphics that are more explicitly cartographic and can be, indeed are being, used to overcome some of the limitations identified above.

Up until now the discussion has aimed to demonstrate that the Tcl/Tk scripting environment is at an appropriate programming level for spatial analysts to produce two-dimensional maps for visualization and for prototyping and investigating the utility of dynamic representations of data that exhibit observer-related behaviour. In both cases the types of cartography required will vary between individuals, data sets and throughout the research process. The flexibility of the language and efficiency with which high level commands can be used to produce user-defined dynamic cartography are thus key features of the approach. Tcl/Tk utilises the flexibility and development speed of the higher level scripting language and the high level commands that represent dynamic cartographic syntax to achieve this. However close couplings to systems programming languages provide the ultimate flexibility, and available extensions to Tcl mean that new higher level Tcl commands can be created to perform cartographic visualization. These approaches sacrifice the productivity and explicitly specified cartographic functionality for the ultimate flexibility if it is required. This can be used to overcome some of the limitations of the approach and provides an opportunity for cartographers to develop a language and environment specifically for visualization.
7.3.1 Additional flexibility: Systems programming languages

Currently the language is extensible, as the source code is distributed and can be modified directly. This approach requires considerable low-level computing skills and significant systems programming effort, and does not provide the clear model for cartography of the higher level language. However, tighter integration with the Java systems programming language is being developed by Scriptics through Td-Blend and Jad (Scriptics Corporation, 1998). These extensions are a Td package that permits interaction with the Java Virtual Machine and a Java implementation of parts of Td respectively. The products will facilitate integration with an important and functional systems programming language that has impressive capabilities for graphics, Internet communications and processing power. The aim is to "create a tight synergy between Tcl and Java" (Johnson, 1998). Java will be used for creating core technology whilst Td will be used to arrange and link Java software components and for tasks that require dynamism and high level control. The developer will be able to communicate seamlessly between the two languages resulting in a more powerful environment than either of the languages provide individually (Johnson, 1998). For example, the iterative cartogram algorithm (Dorling, 1993) currently requires the speed of a systems programming language. Td-Blend would allow a Td procedure to be created from code that is written in Java such as that produced by Herzog (1998) for creating population cartograms. Once the procedure is defined and run in Java it can be called, visualized and interacted with by a Td/Tk script or program and so take advantage of the high level commands presented here for the symbolism and interface. These capabilities provide enormous potential for expanding the interface for cartographic visualization. The ability to extend the graphical scripting approach to cartographic visualization advocated here by adding reusable code that performs standard graphical and statistical procedures could lead to the kind of synthesis between the use of compiled code, GUI and command line that Unwin (1999) appears to champion. This prospect of integration is extremely exciting as visualizers could reuse code to overcome numeric problems and programming inefficiencies that are disadvantages of the scripting approach. It will enable standard procedures to be used by scripters if they are available as reusable software components in a systems programming language.

7.3.2 Extending and formalising cartographic language

These levels of flexibility provide potential solutions to many of the limitations listed above as code can be created to provide some of the required functionality and to formalise the cartography so that it accords with current and developing cartographic theory and practice. These improvements can be achieved in at least three ways involving systems programming languages, the proposed Td/Java integration and standard Td scripts.
At the lowest level extensions can be written in a systems programming language as the Tcl/Tk source code is distributed. A huge number of additional packages exist and extension of the toolkit's capabilities is both encouraged and common amongst the user community. Indeed, as new widgets and functionality are produced, successive releases of the Tcl/Tk source tend to incorporate the most useful of these features. The image processing and manipulation functionality is an example, as indeed is the canvas that is relied upon for the cartography provided in this volume. Programmers can thus develop additional functionality and cartographic items for incorporation with the Tcl/Tk source in a systems programming language to compliment the current cartographic functionality. Currently this is achieved in C++, but the creation of a working version Jad will mean that Tcl extensions can be programmed in Java.

The types of functionality that Tcl does not support currently that might be added includes:

**New Widgets**
- An extension of the 'canvas', the 'map' that provides planar geographic coordinates system
- An extension of the 'canvas', the '3D-space' that provides the item and symbolism functionality of the 'canvas' but based upon a three-dimensional coordinate system.
- An incorporation of these two features to produce a "3D geographic space"

**New Canvas Functionality**
- An item inclusion option that supports linear distance checking rather than that based upon the bounding box
- An option that uses the extent of a polygon item to determine item inclusion rather than its bounding box

**New Item Symbolism**
- A focus/crispness option value pair
- A texture option value pair
- A transparency option value pair
- An orientation option/value pair
- Option value pairs for hue, saturation and value
- Option value pairs for dynamic visual variables

**More Elegant Cartographic Language**
- Each of these features could be incorporated into a more formal high-level cartographic language that follows cartographic convention and procedure even more explicitly than the current syntax. Items could be programmed with option-value pairs that match cartographic visual variables exactly, and interfaces produced that provide alternative symbolism techniques and guidance stemming from the cartographic literature, depending on the data
type. This might take the form of some form of generic cartographic XML (Web Publishing Inc., 1999) mark-up that is useful across map-making and visualization. This could then be specifically interpreted by Td using a parser such as TclXML (Ball, 1998).

- Ultimately the results of efforts being undertaken to formalise the theory of interactive graphics (Wilhelm, 1998) could be incorporated into a language for dynamic cartography.

Researchers are already producing three-dimensional extensions to Td/Tk suggesting that the environment may be capable of incorporating some 3D surface, object and vector visualization in the future (Paul & Bederson, 1996; Esperanca, 1998).

In addition to the approach that uses the Tcl source and a systems programming language to address these issues, some advances can be made at a higher level by writing procedures with Tcl scripts. This approach takes advantage of the prototyping speed and higher level programming skill requirements of the scripting language.

**The 'Focus' Visual Variable**

Figure 7.6 New item symbolism - Incorporating the 'focus' visual variable. (B9)

This figure shows the 'Focus' software, a GUI to the 'makeSymbol' procedure. The GUI on the right is used to specify option values to the 'makeSymbol' procedure. Here rectangle shapes are specified of a selected size with varying locations, colours and degrees of focus or crispness. The window on the left is used to choose a location to display the selected symbol, and the command that would be issued by the current settings is displayed in the window to the left. Clicking the canvas in the GUI on the left creates the symbol. Those produced here demonstrate the possibilities of using this extension for mapping with the focus visual variable. The software is available in appendix B9.

An example script is presented in appendix A6 that defines a procedure 'makeSymbol' that allows a symbol to be specified with location, shape, size, colour and focus. Software that provides a GUI to specify symbol characteristics and creates symbols at selected locations is shown in figure 7.6. The
script uses image items to create symbols as requested and then applies a filter to the symbols in order to achieve the required blurring. It has the following form:

\[
\text{makeSymbol canvasname x y -type shape -width w -height h -focus f -fill col}
\]

Rectangles, circles and images can be used to symbolise information. Due to their raster nature image items do not contain many of the option-values of vector canvas items and so reconfiguring directly is difficult and much of the interactivity is lost. Re-configuring image items requires the production of a new image which is a relatively prolonged process for large images and becomes more so if filtering is required. However, figure 7.6 shows that the script has extended the language sufficiently to enable maps to be created that use shape, location and colour to symbolise geographic information and focus to reflect the certainty of the attribute information.

An additional example uses glyphs that were introduced in section 5.2 to represent locations on statistical graphics. The 'focus' software GUI to the 'makeSymbol' procedure contains an interface element that allows these glyphs to be selected and placed on a canvas interactively. A degree of burring can be specified and applied to each. Figure 7.7 illustrates.

![Figure 7.7](image)

Varying the focus of glyphs that show locations to represent geographic certainty.

This figure shows the 'Focus' software, a GUI to the 'makeSymbol' procedure. Here the GUI on the right has been used select image items. By moving the cursor on the raised square a symbol that identifies the relative location is selected. The focus with which the symbol should be plotted is chosen using the scale bar. As the cursor is moved across the canvas on the left the resultant 'makeSymbol' command and symbol are displayed. The items shown here demonstrate that 'makeSymbol' and the variation of focus that it sanctions permit statistical graphics to be created that contain glyphs symbolising location and confidence in that location. For example, the location might be the modal spatial coordinate of an aggregated data set with the focus representing the standard deviation around that mode. The software is available in appendix B9.
In figure 7.7 the window on the right shows a selected glyph and focus option. When the cursor is moved in the window on the left the arguments to the 'makeSymbol' command that result from the current option settings and location are displayed. Clicking the canvas executes the specified command and creates the symbol. This results in a display that could be used to create a statistical graphic in which the confidence in geographic locations were represented by symbol focus.

The 'Orientation' Visual Variable

An additional script using the same approach addresses symbol orientation and is also presented in appendix A6. It allows polygon shapes to be defined and then located on a canvas with options and values specifying the size, colour and orientation of the symbol. The command has two forms, the first identifies a polygon by a list of coordinates, the second by the unique tag of a previously defined polygon:

```
makePoly canvasname x y -coords coordlist -orient r -scale s -fill col
makePoly canvasname x y -tag tag -orient r -scale s -fill col
```

This figure shows the 'Orient' software, a GUI to the 'makePoly' procedure. The GUI on the right allows a polygon to be digitized and values for the scale, colour and orientation options to be selected. The digitized shapes can be rotated interactively by dragging them with the cursor. Moving the cursor over the canvas on the left reveals the selected options and values to the 'makePoly' command. Clicking the canvas issues the 'makePoly' command and the specified symbol is created. The rectangles and arrows shown here constitute a simple map and could be used to show variation between categories, direction and magnitude. The functionality provided by the procedure can be used to create large and complex maps that symbolise with the orientation visual variable. The software and script for varying symbol orientation are available for use in appendix B10.

Figure 7.8 New item symbolism - Incorporating the 'orientation' visual variable.

This figure shows the 'Orient' software, a GUI to the 'makePoly' procedure. The GUI on the right allows a polygon to be digitized and values for the scale, colour and orientation options to be selected. The digitized shapes can be rotated interactively by dragging them with the cursor. Moving the cursor over the canvas on the left reveals the selected options and values to the 'makePoly' command. Clicking the canvas issues the 'makePoly' command and the specified symbol is created. The rectangles and arrows shown here constitute a simple map and could be used to show variation between categories, direction and magnitude. The functionality provided by the procedure can be used to create large and complex maps that symbolise with the orientation visual variable. The software and script for varying symbol orientation are available for use in appendix B10.
Moving the cursor over the canvas in the window on the right shows the command that would be issued were the canvas to be clicked, with the option values to 'makePoly' reflecting the options selected using the GUI. Clicking the canvas issues the command and produces the symbol. The shapes shown in figure 7.8 can be regarded as a simple map and show how the Td extension presented here can be used to enhance the cartography that is achievable with the language.

This technique also provides a means of formalising the cartographic syntax that is used as it is possible to produce procedures that define cartographic symbols such as:

```
createSymbol polygon coords -hue h -sat s -val v -texture texture -scale scale id data col n col n
```

Dynamic properties might trigger updates to the configuration by issuing commands such as:

```
change id opt val opt val
```

These commands could be issued to update all items with the specified identifier in all open views and thus enable brushing to take place automatically.

### 7.4 Conclusion

The degree of acceptance of the approach, popularity with which it has been greeted and variety of documented outcomes and ongoing work strongly suggest that the work presented here is of consequence. The high levels of flexibility provided by the systems programming interface to Td/Tk and the benefits for dynamic cartography outlined here mean that investment in the approach is appropriate and the kind of development and extension proposed and demonstrated here is both feasible and potentially profitable. The assessment of the types of map product currently possible and potentially achievable in the near future if rates of improvement continue reinforce these conclusions.

The extensions implemented here by way of example are not particularly ambitious. A more significant additional step would involve the production of smart cartographic items that are designed to represent geographic phenomena such as streams, roads, towns, wards and other objects. These could be attributed with options for expected data fields as well as for scale, generalisation, forms of symbolism and other elements of cartographic display and so encompass established cartographic theory in a truly object-based model and integrate data model and representation more fully. This possibility is made useful and more likely by the proposed support of formal connections between the Td/Tk scripting language and the object-based Java systems programming language.
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

8. Conclusion and Futures

Jason Dykes
Department of Geography
University of Leicester
UK
8. Conclusion and Futures

8.1 Synopsis of Thesis

This volume began by outlining a series of new types of map behaviour and an old set of cartographic quandaries exemplified by issues concerned with the transformation of geometric and statistical geographical information into a visual representation of enumerated data. The new types of map behaviour have become possible during the period of research due to the technological advances that have been made in the second half of this century and the acceleration of innovation and impetus in the recent past. These technologies and the type of graphic that they permit were noted to be having an effect on the way in which science was being undertaken and were both changing the types of visual representations of information being employed in research and the ways in which they were being used. The techniques were adjudged to have potential application to geographic data by extending the types of cartography with which we are familiar for use as exploratory tools early in the research cycle, during the process of ideation. A need to develop an environment for visualization from which flexible and responsive maps could be produced with suitable observer-related behaviours that provided an interactive and geographical interface to spatial information was established. It was postulated that these dynamic maps for visualization should be different to those produced for other map uses and might benefit from the synthesis of series of design criteria from a number of fields in addition to traditional cartography and through the generation of novel representations of geographic information.

An approach was introduced that made novel and imaginative use of a programmable graphical user interface builder with a scripting language as its interface. This dynamic environment for interactive visualization was presented using simple data sets and examples to demonstrate the syntax and the ways in which cartographic symbolism and observer-related behaviours can be achieved, along with the relationships between traditional interface elements, smart cartographic symbols and encapsulated data.

Working software with significant novel and interactive functionality was then presented with online guidance and fully documented source code. The software redefines the map as a dynamic interface to spatial data and shows how the approach empowers spatial scientists with exploratory data analysis techniques, such as transient symbolism and linked views of data, in a flexible mapping environment. The software explicitly demonstrates the approach in terms of the embodiment of the ideas presented, the incorporation of a number of techniques and influences and the scripts which describe the functionality and model expressly.

A series of extensions to this initial software prototype were then presented that provided interfaces for the exploration of cartographic representations, the combination of spatial and
statistical information and the incorporation of additional forms of spatial information. Alternative pieces of software that take advantage of the same approach for visualization were also introduced. A description of their application was provided to demonstrate the flexibility and aptness of the vehicle used, its relevance and the suitability of scripting for incorporating new technologies into cartographic representations as they become available.

The context and consequences of the research were then outlined with a report on the use of the approach, the implementations that demonstrate it and the development of these pieces of software for further visualization. Some aspects of the approach were discussed in order to assess its applicability and future and the relevance of the methods used as well as the types of dynamic map that they permit. Despite some current limitations the flexibility and robustness of the environment and its likely continued development in a rapidly changing field suggest that additional investment for visualization is appropriate. A number of examples of additional functionality and formalisation of the cartographic language were suggested and demonstrated along with ways in which they might be accomplished.

8.2 Review of Assertions

This format has addressed the assertions made in the opening chapter and verified their authenticity:

- The techniques for visualization introduced in chapter 3 and utilised in subsequent chapters have demonstrated that spatial enquiry can take advantage of techniques being developed in a variety of academic fields.

- The production of a number of highly interactive GeoGUIs that combine these approaches has undoubtedly re-defined the "map" as an extremely dynamic interface to information, a tool for visual thinking, rather than a single proclamation of the best estimate of some truth. The numerous implementations of dynamic comparison and re-expression provide examples, as does the work in chapter 4 that allows users to explore representations for information and assess map and pattern stability.

- The creation of maps that are as dynamic as a graphical user interface has been outlined in full (chapter 3). The flexibility of such an environment for exploratory dynamic cartography at the command of the enquirer has been demonstrated by developing scripts for additional functionality throughout chapters 3 to 6.
Appropriate novel analytical and representational techniques have been produced and demonstrated. These include the dynamically linked cartograms (chapters 3 and 4), the exploration of local statistics (chapter 4), the local multivariate graphics (chapter 5) and the incorporation of qualitative multimedia data and virtual environments into an analytical arena (chapter 6).

The 'cdv' and 'panoraMap' software developed and provided in appendix B demonstrate that attractive/working/usable/ software can be created using the approach.

In combination the work presented has shown the approach to be suitable for use with a number of data types and in a number of applications. Chapters 3, 4, 5 and 6 illustrate this with numerous examples.

In light of the success of the approach outlined here, the corroboration of these assertions and the belief that ongoing research is appropriate using the approach identified it is worth considering a number of recent and likely future developments that influence these statements. The following section outlines and assesses advances made in parallel to those presented here and considers the near future of cartographic visualization in order to provide context to, and assess the applicability of, the contribution made here.

8.3 Assessment of Recent and Future Trends

The potential offered by the combination of cartographic functionality, flexibility and an environment for visualization that permits rapid prototyping is extremely exciting and offers great potential for formalising our cartographic past and present. It has, however, been stressed throughout that all of the work provided here lies within a rapidly changing and developing field. Hence the 'wave' metaphor used at the outset. Our cartographic futures are uncertain and one can only speculate about their likely direction. A number of recent developments however, have been of particular significance. Their potential consequences in the medium-term future for visualization, scripting and the approach used here are discussed briefly in this section, along with some thoughts on directions that would appear to be of potential benefit having undertaken this research.

8.3.1 Combination of technologies

The work reported here is one of a number of efforts to combine useful techniques from a number of fields to achieve some specific graphical objective. Chapter 1 noted that at the onset of this research a number of cartographic developments were integrating techniques from other fields.
The trend towards combining methods from previously separate technologies has paralleled much of this work. When the research began MacEachren et al. (1994) offered the sentiment that the lack of appropriate software for exploratory analysis meant that the visualization process required a 'boot laces and sticky tape' approach. Different software systems were used throughout the process to perform computations and achieve different types of graphic and interaction. Links between each system required significant effort and data transformation. Research that used visualization thus suffered from a reduction in the response time, levels of interactivity, and types of graphic that could be produced. A number of developments have attempted to bridge these gaps more smoothly with close couplings between geographic information software, cartographic functionality and statistical packages. Good examples include the work of Cook et al. (1997) whose connections between Arc/Info and xGobi are impressive, and the links between Arc/Info and S-Plus for statistical analysis with a geographic base. The geographical functionality and cartographic representations used in MANET (Unwin et al., 1996), an extremely powerful and impressive tool for graphical statistical analysis have also progressed considerably.

As new types of media, data and analysis develop additional links will need to be achieved. Scripting languages provide a means of achieving this, as outlined in the previous chapter and demonstrated in chapter 6. They are often referred to as 'glue' for combining components written in lower level systems programming languages (Ousterhout, 1998). They certainly provide a more seamless, elegant and closer coupling than the 'boot laces and sticky tape' approach, which is precisely what MacEachren et al. (1994) requested and furnish sufficient flexibility that additional, as yet unknown, software can be incorporated. The time for scripting may well have come.

Certainly commercial GIS vendors are taking account of the research and changes that are going on and are moving in the direction of the kinds of dynamic graphics and flexibility that are suitable for visualization. The work of Hansen (1996) demonstrates the point. Whilst it could be argued that there are some advantages in using proprietary GIS for visualization, the flexibility and control available when using a graphical toolkit mean that the approach outlined here is more appropriate for development work even now, despite the provision of higher-level macro languages in Arc/Info. This notion is supported by the range of examples presented here and by considering the 'potential enhancements' wish-list for visualizing enumerated data sets provided by Egbert and Slocum (1992) in their epilogue to EMAP and noting that the majority of these are eminently more feasible with the graphical tools afforded by Td/Tk than with those available in Arc/Info (ESRI, 1999).

It remains to be seen whether commercial GIS vendors address visualization map use seriously. Bivand (1998) notes a trend amongst commercial off-the-shelf products to permit extensive customisation through little languages for scripting and interface construction and a continuation of this trend will be extremely useful for visualizers. If the scripting language platform company is to be believed it appears that many commercial companies are already investing in scripting for 'glue',
integration and software development (Simms, 1999). The evidence provided in this thesis would suggest that some synergy between these vendors could be of commercial and practical benefit.

If a map-maker is focusing on links, and the combination of existing software components to perform visualization with uniquely specified views and functionality, then a scripting language is currently an appropriate development solution. If the thrust of development is on the data or the content then it may be less so, with systems programming languages being more suited. We have seen that these are not mutually exclusive options, and that techniques are being developed to integrate them closely. Fourth generation programming environments such as scripting and visual programming languages are resulting in a division between communities in the world of software development between 'component developers' who use traditional third generation systems programming languages and scripting centric 'application assemblers' (Johnson, 1998). The former build large reusable software components with tools such as C, C++ and Java that focus on performance, memory management, protocols and other low-level system characteristics. The latter connect these modules or applications together and so are less concerned with low-level system characteristics. They look for tools with high degrees of flexibility, efficiency and dynamism as the focus is on relatively small amounts of code that specify the connections between existing components written in third generation languages. The development of tools such as Td-Blend and Jad will result in a dual language environment that can support both communities of developers. The more maps become interfaces to data, providing access to user-defined data views that exhibit observer-related behaviour, the more that the concentration in cartography will focus on the links rather than the content, and the more scripting languages are likely to become relevant for mappers. Those with cartographic potential will become increasingly attractive to multimedia cartographers who are endowed with the ability to use and incorporate powerful software components into their maps. Tcl/Tk is an excellent example with impressive dynamic mapping capability and exciting additional functionality planned in the near future.

8.3.2 Advances in communications and sharing of data and software

Recent developments and advances in communications technology and the way in which software is written in an Internetworked computing environment have an impact on this work and cartographic visualization as a whole.

The Internet is an essential dynamic medium for maps that communicate and provides a whole host of possibilities for mapping in the future. Maps are becoming a popular way of presenting and accessing information, and their prevalence in the software medium of the World Wide Web means that an increasing proportion of map users expect the products that they use to be dynamic. This trend is a further pressure on the visualization/communication continuum as large numbers of people use the Internet to access, assess and analyse geographic information with dynamic maps.
In addition to promoting visualization amongst the masses these capabilities mean that spatial
scientists can perform collaborative visualization, share data and techniques and make the
methods as well as the results of geographic enquiry available to the community. Several authors
in the special issue of Computers & Geosciences on exploratory cartographic visualization
(MacEachren & Kraak, 1997) presented their techniques interactively in a WWW supplement to the
journal (e.g. Mitas et al., 1997).

The approach outlined in this volume can be used to produce interactive maps with programmed
observer-related behaviours on the Internet in a number of ways. These presented a significant
opportunity for visualizers as code and graphics can be readily distributed for communication and
verification (Bivand, 1996). The release of a Td/Tk plugin for Netscape and Internet Explorer from
Sun Microsystems on July 10th 1996 made it possible to view and interact with Td/Tk maps
through a number of popular WWW browsers. A number of limitations exist with this environment.
As is the case when running Java applets, ‘tdets’ are scripts that download to a local machine from
a WWW server and run a program locally. Evidently a series of restrictions must be made on the
functionality of these scripts if security is to be maintained and so some of the attractive features
of Td/Tk such as the file opening and manipulation procedures and the ability to produce any
number of linked windows are not available. Thus the capabilities of the plugin are a subset of the
functionality of the full source. Safe Td is being developed to address these issues and ensure
security so that additional functionality can be incorporated into the plugin and permit encryption
so that data can be distributed (Scriptics Corporation, 1999). The Historical Atlas of Industrialising
Britain (Southall and White, 1998) has already distributed interactive maps on the Internet by
taking advantage of the Td plugin.

The Internet isn’t just about browsers and plugins however, the most important feature is in the
ability to transfer information and communicate. Sun promoted Td/Tk as “a universal scripting
language for the internet” (Sutherland, 1998) and Scriptics have continued to provide impressive
capabilities within Td/Tk.

Internet communications can be developed using the scripting language (Scriptics Corporation,
1998; Ball, 1997) and the ‘socket’ command makes servers and clients programmable in a few
lines of code (Hopkirk, 1997). By using the GUI to call application components written in systems
programming languages external servers and clients can also be used to transfer information
across ‘the Web’. The ‘panoraMap’ software can access the Internet in this way, by running Java
applications that have been developed to communicate with a remote spatial database (Dykes et
al., 1999). The Td application runs a local Java client invisibly to the user with an ‘exec’ command.
The client communicates with remote servers and returns textual information provided by the
server software to the application as a series of strings. This text identifies the names of files that
comply with certain search criteria specified by the user through the application interface and
additional metadata. The software can then use these metadata to update the interface, and
download the appropriate file for inclusion in the local application if it is requested by the viewer.
Spatial searches of a remote database can thus result in symbols being plotted on a local map to
identify geo-referenced panoramas and multimedia. The client downloads the appropriate files,
which are then incorporated into 'panoraMap' and accessible for viewing in the same way that local
files can be seen (see chapter 6). So if a user makes a request to a remote server for all maps with
the same bounding coordinates as the currently displayed map these will be returned to the
software by the client which issues the specified series of criteria to search the database behind
the server. The software will then issue additional commands through the client to download the
maps and will respond in a suitable way by updating internal data structures and producing GUI
widgets from the returned textual metadata that allow the data to be viewed and incorporated
appropriately.

In 'panoraMap', when a user makes a request for a map from a remote server then panoramas
located within the bounding coordinates of the map are downloaded into the hierarchical data
structure required by the software and mapped automatically. Appropriate links between
panoramic images are computed as demonstrated in section 6.3. The software is thus Internet
enabled and extremely flexible in terms of the data that can be displayed within it. Requests can
be made to a number of different servers and data cached and combined from a variety of remote
databases. The result of these Internet links is a two-dimensional geo-graphical Internet browser
that is based upon a navigable virtual environment and has wide application for communication
purposes (figure 8.1). The combination of this networked functionality with the analytical
procedures contained in the software means that 'panoraMap' supports the visualization of remote
information. The described architecture provides two-way communications functionality meaning
that the 'panoraMap' interface can be used to add data to a remote database as well as to
graphically define criteria by which a remote server is searched for data in order to extract a
relevant data set for visualization (Dykes et al., 1999). The backdrop map features a draggable
selection box to achieve this latter functionality that returns selected data within the specified
region from remote Hub clients. The software is thus effectively a map window to the WWW
implementing the functionality that permits visualization and provides visualizers with metadata
about their data set and enables them to search remote databases for relevant exogenous
information and additional quantitative data. These sources provide relevant auxiliary information
to support the transformation between maps that are produced, the data model and the real world
issues that are under analysis and to guide conclusions drawn from them. Such an open ended
architecture is extremely appropriate for informed visualization and is akin to the open ended
scripting model that ensures flexibility in terms of the view that is presented and the types of
interaction that it exhibits.
The loose coupling described here could be tightened with the Tcl-Blend extension that would allow the client to be called as a Tcl command. Opportunities such as this for blending developments that utilise the processing speed and power of systems programming languages with the development speed and flexibility of scripting are likely to be beneficial in the future.

Figure 8.1 Further extending the approach - 'panoraMap' and the Internet.

External applications can be run from the GeoGUI. These include software that displays the georeferenced multimedia data, like the Netscape browser shown here (top right), which allow the forms of data included in the map to be extended. In this case imagery and text from the Dartmoor National Park Moor Care database on erosion are shown along with links to additional remote information.

Additionally networking software can be run and incorporated into the application. Clicking a symbol that represents a file produces a metadata GUI (bottom left) that can be filled with appropriate information about the multimedia file. The 'Hub' menu on the main application window (top left) then enables users to select a remote machine and account using the VFC Hub model and software (Dykes et al., 1999) and make a connection. The multimedia data and metadata can then be sent to the remote database at the click of the 'Submit button.'

'panoraMap' can download data from a remote database and automatically provides a spatial interface to these data and metadata using the same communications mechanism.

The issues concerned with distributing data sets in any form (such as copyright, liability, data value and personal investment in data collection and enrichment) are likely to be considerable hurdles to
the advancement of this form of visualization. The utility of highly generalized spatial information for visualization and increasing use of non-traditional data sources for representation may counter these effects.

8.4 Some Effects of The Internet

These trends towards networking are having effects beyond those of Internet-ready software and data sharing. An opportunity exists for the sharing and joint development of software products and new forms of geographic data models are being developed to account for the distribution of data across the Internet that may change the way we encapsulate certain types of spatial information.

For example, GeoTools (McGill & Turton, 1999) presents a set of reusable cartographic components that provide a potential basis for collaboration amongst cartographers using Java classes. These could be used to produce a standard set of techniques and a cartographic syntax and are a viable alternative to the Tcl/Tk approach. Potential exists for the rapidly developing area of streaming technologies to provide location dependent geographic information as the user navigates across a map or through a virtual world. By sending information within a geographic vicinity across the Internet as the user changes their location an apparently seamless representation can be produced. This functionality forms part of NetGIS (Kahkonen et al., 1999).

Entertainment companies are investing heavily in streaming and the technology may overcome some of the issues concerned with the sizes of data files and numbers of calculations required when currently manipulating three-dimensional representations. Abadjev et al. (1999) stream meshes of a sort that could easily be used to represent geographic landscapes, whilst Naka et al. (1999) use a Real Player plugin to stream 3D geometry, suggesting that multi-scale models could be used to address these issues. Applications could learn likely user behaviours and add and remove elements of the surface as and when required to produce an Internet mounted seamless virtual world. If a user were travelling constantly in any single direction coarse spatial information would be streamed in to the model at the horizon, populated with detail as the user approached and removed from the model as the user moved away. By extending the data stream to account for any movement of the viewers' location and the view frustum (Hoppe, 1997) a continuous, seamless, streamed and manageable view-dependent mesh could be produced that replicated our real-world experience of observed detail density decaying with distance. These developments must be considered and could form our cartographic futures. Making sure that relevant forms of representation are possible within these types of modelling environment that are being produced is an important job of computer cartographers (Dykes et al., 1999) if they are to ensure that their cartography can take full advantage of these exciting advances and vice versa. It is to be hoped that the powerful players in the digital media arena can be persuaded of the relevance of...
8.5 New Issues to Address

A whole series of new ways of collecting, storing, representing, displaying and accessing data have become available recently and the range of types of cartography that are feasible is steadily swelling. Continuing advances in computing are also likely to present new issues with which cartographers will need to grapple. These will include having to find suitable forms of visualization with new types of data, hardware, and interface. For example, the changing face of cartography is exemplified by the case of a technology that has developed immeasurably since this research began - that of global positioning systems (GPS). In the early 1990's receivers were expensive, bulky and had such limited graphical output that the production of dynamic cartography from GPS output was barely an issue, let alone the provision of highly precise interactive maps within the receiver. Quality receivers with real time ‘You are Here!’ graphics are retailing for under $100 in the US at the time of writing with a number of effects. Newer models incorporate geographic databases and real-time maps presented on 4cm square colour LCD displays. The type of interface that is required in this situation, for the type of map-use that GPS owners undertake, presents a whole series of additional issues involving map design for specific purposes with a particular series of technological possibilities and restraints. A response to this new data type was presented in chapter 6 where the combination of GPS receiver, computer and dynamic mapping software was utilised to associate the real world and the map model intrinsically. Resolving these issues with additional constraints will require some carefully and skilfully applied cartography for dynamic mapping and additional hardware constraints and features are likely to test and extend our cartographic techniques as developments occur.

GPS technology is just one example of a whole series of new spatial data sources that are emerging. The resultant masses of data mean that the need to visualize information is more relevant now that it was at the time of the McCormick Report (McCormick et al, 1987) with greater computer processing power and more accessibility to it.

A whole series of real-time digital data sources are becoming available and representations that permit their visualization are being developed in parallel. Examples include real time locational information from cellular telephones and car navigation systems, the speed sensors that constantly collect information about traffic flows that are appearing on motorway bridges and data collected about access to computers that house communications software such as WWW servers. The WWW is an enormous data source in itself that can be analysed with cartographic visualization. The relevance of documents can be mapped to show their relative proximities in an abstract space (NewsMaps.com, 1999). Virtual worlds such as Alpha World (Dodge, 1999) provide data that...
permits geographic enquiry and the analysis of virtual geographic and sociological processes that
are inherently spatial (Dodge, 1999). Data concerning telephone calls is collected in enormous
quantities with gigabytes of data being streamed into telecommunications companies by the day
(Wills, pers. comm.). Interactive graphics can be used to assess the conversations that take place
as well as the geographic and other attribute information associated with each call in order to
detect patterns, explore markets and detect phone crime from enormous data sets (Wills, 1995;
Wills, 1998).

Store cards are an additional data source that provides multivariate spatio-temporal information
which demands appropriate visualization tools in order to search through the data and relate it to
additional geographic information for commercial gain. Additional examples from the field of
geodesy demonstrate how complex spatial information can be generated from existing historical
sources with computational techniques (Fairbairn & Taylor, in press). Applications in geodesy and
other sciences provide maps in real time in the field and demonstrate the rapidity with which
Tobler’s (1979) transformations can take place, and decisions be made based upon interaction
between the maps, users and data in real time in the field (Pascoe & Ryan, 1996). Field
computing, and the assessment of complex multivariate data for decision making in the field is an
important future focus for visualization.

8.6 A Short-term Example

It seems likely that visualization that relies on interactive maps will become an increasingly
important technique for making sense of the escalating quantities of spatial data from which we
have to make decisions. The types of application presented here are likely to remain appropriate in
the short term. It is, however, impossible to assess how well the approach will account for the
types of issue introduced above.

How well we can use developing technologies and how successful we are in assessing our
accumulating data resources and thus gaining knowledge about the world that those data
represent may well depend upon the ways in which the strands of research identified here are
synthesised. Environments that enable developing technologies to be integrated with a common
approach are likely to lag the developing technologies, thus there is a tension between
implementing new views and using new data and media and addressing well-known problems by
synthesising familiar techniques.

For example, the approach to cartographic visualization introduced, implemented and applied here
has aimed to provide a balance by presenting novel interactive maps that use traditional symbolism
to extract information from a familiar data type. Methods for displaying, analysing and integrating
novel media types such as those identified above will follow in the wake of the accepted use of
those media types. The approach taken here has been shown to be able to support some of these and produce forms of representation at the cutting-edge of multimedia cartography and cartographic visualization and apply it. The discussion presented suggests that continued investment in this approach is suitable. The types and magnitudes of data that it can encompass and the ways in which it might be used to produce effective interfaces are evolving.

As techniques develop and digital data resources become the norm the series of cartographic transformations outlined by Tobler (1979) and referred to throughout this thesis will become increasingly flexible. For example, the type of enumerated data model used by way of example in this volume may seem relatively rigid. It will however be available in a number of forms following the 2001 census. It is possible that data will be input and stored at an individual level, with enumeration occurring as specified by the user when data are output. New means of integrating data sources such as personal data reported on loyalty card application forms, shopping habits recorded by self-scanners and Internet sales and census information are likely to be developed. The data models from which we map in order to conduct geographic research may well be very novel and are likely to be extremely flexible.

As the 2001 census approaches far more ambitious forms of visualization can be attempted using novel views from existing data models. Wood et al. (1999) use population density surfaces and derivatives calculated from them for visualization. The approach outlined here can be used to incorporate more symbols, and add specialist functionality required to interpret these new forms of representation. Figure 8.2 shows how the increasing speed at which Td/Tk operates means that a national cartogram can be produced showing all 14,000 wards of Great Britain concurrently. This can be programmed to exhibit dynamic behaviours for real-time interaction. When displayed on a computer screen a number of issues arise that contrast with the publication of such an abstract map on paper. Whilst the levels of interaction enable users to relate geographic locations with areas on the abstract projection the mass of data cannot be displayed as effectively at 72dpi on screen as it can on paper at 600dpi. Techniques are thus required to zoom and pan around the abstract space in order that it might be interpreted. Figure 8.2 show methods that have been developed for doing so in advance of the 2001 census.

Whilst a cartogram with 14,000 symbols provides high levels of detail required to perform visualization on such a rich spatial data set, scatter-plots and parallel coordinates plots of the sort outlined in previous chapters are likely be less useful tools for eliciting patterns with such large numbers of cases. Techniques need to be developed and applied to address the issue of large numbers. This will even more necessary if the graphical capabilities at our disposal in the more distant future permit us to visualize all 144,000 enumeration districts in Britain concurrently.
One outcome of these more specialised interface operations is that usability becomes a key issue and it may become desirable for software to provide more guidance as to the way in which visualization takes place. The ICA Commission on Visualization has identified the potential role of intelligent agents in this process. In a parallel response Andrienko and Andrienko (1999) have produced software that utilises many of the views implemented in 'cdv' and takes advantage of cartographic research by guiding users into appropriate data views and symbolism. A more general repercussion is the ability of the user to perform and control numerous parallel cartographic transformations from the real world to the map image and vice versa. Such capabilities provide the potential for more informed and successful geographical analysis and confirm that the role of the cartographer and extent of cartographic issues extend well beyond a single one-directional transformation from data model to representation and into the specification and provision of appropriate and flexible dynamic cartography.

Figure 8.2 Scaling the approach - Cartogram visualization at the national scale.

Here two pairs of images demonstrate functionality required and implemented in order to display the 14,000 electoral wards of Great Britain in a single map. The data are represented by linked views showing symbols of equal size at zone centroids (right image in each pair) and a non-continuous population cartogram (left image in each pair). Real-time brushing and other forms of observer-related behaviour are now feasible with dynamic maps that contain this number of symbols, using the approach outlined here. The images show how zooming into a new viewing region in one view results in the cases at the edge of that view being highlighted in the other, so that areas within the geographic and abstract spaces can be compared and assessed.

8.7 Conclusion

This thesis has reported on the near past and present and so provided the kind of way-marker described in the introduction. The integration of a number of techniques and data types has been demonstrated. The original use of a scripting language has been outlined and the synthesis of some relatively abstract and simple maps, GUIs and data has carefully detailed an approach that
has a number of significant advantages for cartographic visualization. These have been applied to address issues of interest to the visualization community and to perform visualization in order to analyse sample data. Whilst the body of the work presented focuses on the links between approach, implementation and application the three elements can be considered in isolation. The successful accomplishment of useful research into any one of these would be a worthy effort in the field. Alternative strategies could have been employed to address any of these elements. Imaginative use of other software vehicles might have led to similar implementations, which could in turn have incorporated various other dynamic behaviours data sets and data views, and these could easily have been applied to undertake and assess visualization relating to issues other than those presented here. By undertaking research at each stage, linking the stages closely together and applying them to a particularly problematic data model it is hoped that the utility and value of a unified but flexible approach to visualization has been established and a significant contribution made. The underlying model is particularly advocated to perform visualization due to:

- the high level and explicitly cartographic nature of the language
- the flexibility that it accommodates
- the capacity for rapid development
- the large potential user-base amongst spatial scientists
- the fact that development of the approach and changes in the field appear to be making the vehicle a more suitable one than ever

Evidence has been provided to demonstrate these features and to show that the approach, implementations and their application have been endorsed by members of the academic community.

But the waves of technological change are frequent and of great magnitude, and no way-marker lasts forever. The example provided above looks at the near future, whilst much of this chapter has attempted to assess the mid-future. Evidently we will continue to need to develop techniques for incorporating the types of functionality and flexibility demonstrated here with ever-larger data sets consisting of a range of ever more diverse spatial information and media in order to address a whole spectrum of issues. Ultimately this will be achieved by continuing to focus on research priorities such as those identified by the ICA relating to representation, interfaces and visualization/database links. Some of these might be addressed with Tcl/Tk, others with scripting. Dynamic maps are always likely to be useful and visualization appropriate. It is to be hoped that the kind of holistic and flexible approach demonstrated here can be built upon in order to achieve an equally elegant environment for mapping in the future. And that the cartographic visualizers of the 21st century demand the types of environment that are becoming available at the end of the 20th century to visualize the data that they use to address the issues with which they are concerned. Efforts such as the research presented here, that demonstrate an approach and make
such flexible and interactive tools available to the community, can improve the likelihood of this happening as we move forward.
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

Appendices

Appendices
A - Text Appendices
B - Software Appendices on Accompanying Compact Disk
C - Support Materials on Accompanying Compact Disk
D - Academic Papers in Cover Pocket

Jason Dykes
Department of Geography
University of Leicester
UK
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

Appendix A: Text

A1 - Spatially Optimised Classification for Choropleths
A2 - Notes on Taking Panoramic Imagery
A3 - Selected Scripts from the 'cdv' Software
A4 - Scripts that Extend 'cdv' Functionality
A5 - Virtual Reality and Visualization in the Fieldwork Context
A6 - Scripts that Specify Focus and Orientation

Jason Dykes

Department of Geography
University of Leicester
UK
Appendix A1:

Spatially Optimised Classification for Choropleths

Jason Dykes

Department of Geography
University of Leicester
UK
A1. Spatially Optimised Classification for Choropleths

In section 1.5 a number of possible responses were noted to the mapping of enumerated information for visualization. They fall into the three main categories identified by Fisher et al. (1993).

1) Production of Static Maps with Optimised Design Criteria
2) Novel Views
3) Increased Interactivity

This appendix provides a brief summary of work that was undertaken into optimised design criteria for static choropleth mapping. They aimed to extend work that had been undertaken on choropleth classification by optimising in order to maximise spatial characteristics of the mapped data. The objective was to take greatest advantage of the abilities of human vision and visual processing by classifying to maximise spatial association. This was attempted by measuring a spatial statistic from a classified map and re-classifying continually under constraints in order to maximise the chosen index.

A1.1 Rationale Behind Spatially Optimised Design Criteria

Tobler (1973) argued that the most useful representation of a choropleth displayed as many grades of shading as data values, but others (Dobson, 1973) counter this by noting that choropleths without intervals do not model or provide insight. The argument for each alternative may depend on the maps location along the visualization/communication continuum. In communicative terms, regrouping data items decreases the communicative bandwidth, and imposes standards on the information, reducing the potential for individual personal vagaries in interpretation. Regarding ideation, classifying with a specific goal renders a model of an aspect of the data set that might reveal information that is not apparent in other views. These reasons for classifying are summarised by Muehrcke (1990, quoted in Buttenfield & MacKaness, 1992) who notes that "it is abstraction, not realism that give maps their power".

Once free from the constraints of being accurate data inventories, digitally produced maps can be used to display spatial pattern at the expense of other map functions. The use of maps to model pattern is termed 'map analysis' by Muehrcke (1978) and 'intermediate map use' by Bertin (1981).

It is postulated that the incorporation of spatial associations as classification parameters instead of statistical goals such as 'map accuracy' (Jenks and Caspall, 1971) would achieve this. Traditional classification techniques categorise a series of values in a single dimension and map them, yet if space, the fundamental of geographic study, is given sufficient status in solving a geographical
problem then maps may be produced that both reveal spatial associations and thus provide insight into dominant underlying spatial processes and communicate spatial distributions effectively. This latter suggestion is supported by the recognition that map communication effectiveness is dependent upon map complexity (Muehrcke, 1972; Gatrell, 1974). The goal of internally homogenous classes (Jenks & Caspall, 1971), or fitting classes to the statistical distribution (Evans, 1977) can thus be subordinated to the objective of revealing spatial relationships (Monmonier, 1972, 1974). This is an important recognition, and one that has frequently been overlooked. For example, Evans (1977) acknowledges 'contiguity biased' class selection, but does not advocate it for reasons relating to accuracy, and Egbert and Slocum (1992) have no spatial association classifier in their interactive choropleth visualiser, EMap, and regard the provision of measures of spatial structure as no more than a 'potential enhancement'.

Several features of the choropleth negate against statistical accuracy, and so add strength to the argument for the inclusion of a spatial pattern component in classification. For example the data are masked by aggregation and the effects of zone size and shape as filters and the imposition of unnatural boundaries has a considerable effect on aggregate values.

The production of a spatial classifier satisfies a further demand. Dawsey (1990) refers to the need for aesthetically pleasing maps without continuous user interaction, and Wiebel and Buttenfield (1992) suggest that pre-specified map designs are supplied for non-proficient users of interactive systems. The incorporation of a spatial classification algorithm would help fulfil these requirements.

A1.2 Indices of Spatial Association

Numerous coefficients have been applied to area value data in order to measure spatial association and map complexity (e.g. Muller, 1976; Olson, 1974; Muehrcke, 1973; Monmonier, 1972, 1974). In order to quantify the spatial arrangement of areal values so that this feature of the map can be optimised in the classified representation, a suitable spatial statistic is required. The most mathematically rigorous indices for quantifying spatial association are those of autocorrelation. According to Goodchild (1986) "...if one were to summarise a spatial distribution of unequal attributes in a single statistic, one would in all likelihood choose a spatial autocorrelation index".

In order to qualify the potential for varying map autocorrelation by class delimitation Moran's I, a standard measurement of spatial autocorrelation was thus used. The index is the weighted sum of the product of separate data observations centred on the expected value of the observations, standardised by their variance and normalised for the total sum of the weights (Cliff & Ord, 1981). A $1/d^2$ weight function was used to calculate $I$. In order to evaluate the effects of choropleth
classification on autocorrelation and determine the potential for using it as a map classification parameter, a series of data distributions with varying degrees of spatial dependence and statistical characteristics were created. These included a regular grid of units and an irregular set of polygons. To keep the classes in realistic realms thirteen classification algorithms were implemented, and each set of zones classified into 3, 5, 7, and 9 classes for the variety of distributions.

A1.3 Comments on the Effects of Varying Classification Strategies

The regular grid illustrated that varying class delimitation strategies can alter indices of autocorrelation. The range of effects were data specific. Random distributions, or those that varied in a spatially uniform manner displayed less fluctuation in autocorrelation when class boundaries were altered than those which combined spatial trend with a random element to produce a complex, but autocorrelated surface. Informal analysis of the coefficients illustrated that for a simple grid, the visually interpreted association matches relatively well with Moran's I.

The coefficients returned for the irregularly zoned data sets, however, proved impossible to estimate in visual terms. This finding epitomises problems basic to the tenets of the visualisation methodology, which seeks to make use of our pattern recognition capabilities and the proficiency of the eye to absorb large amounts of information very rapidly. It concurs with the opinions of Lloyd and Steinke (1976), Muller (1976) and Peterson (1985) who suggest that individuals have very different bases for identifying pattern in irregularly zoned choropleths, and Openshaw (1992) who notes that the brain is easily tricked and misled whilst the patterns that do occur are often far too complex for visual recognition and interpretation.

The way in which an autocorrelation index rightly biases outcome toward variation between very close neighbours directly counters the way in which mapped trends are recognised and perceived through areas of contiguous shaded blocks. For example, a homogenous series of values, surrounding an extreme deviation will produce a low (not autocorrelated) coefficient despite the evident homogeneity in a high proportion of the mapped space. This effect is especially acute where the values represent small zones, as the $1/d^2$ weight function is very sensitive to exceedingly close centroids, whatever the values of the visually dominant larger zones.

In addition to these failures of the utilised coefficient to match perceived patterns, the indices are reduced in calibre by the use of simple Euclidean geometry. Employing physical distances between centroids generated from irregular zones is naive both in terms of map pattern quantification and the measurement of spatial processes. Further restrictions of the spatial autocorrelation index
involve uncertainty surrounding the definition of a standard weight function meaning that Moran's I is no different to many other weighted spatial statistics in that it may be manipulated to 'prove' that a data set behaves in a particular manner, for instance Dacey (1965) and Chou (1991) advocate distinct complex separability statistics. An additional constraint concerns the inability to examine the spatial relationships for neighbours other than those immediately adjacent with this technique, which is required when examining associations and spatial processes within enumerated information at a range of scales (Gatrell, 1974).

A1.4 Conclusion

These findings suggest that Moran's I and distance based autocorrelation coefficients do not describe pattern parameters that can be used successfully as a basis for illustrating visually recognizable data trends when classifying irregularly zoned areal data sets. This is a notable conclusion, as many maps that have been used previously to measure complexity involve fewer zones of substantially less size variation than those used in this analysis (e.g. Muller, 1975; Lloyd & Steinke, 1976; Jenks & Caspall, 1971). The influence of the weight function may not have been fully appreciated by studying irregular zones that approach a regular lattice, highlighting the need for continued research into identifying methods for the extraction of the information concerning spatial associations contained in a complex area value data set.

These empirically derived findings result in several conclusions:

- The initial supposition made here that that a spatial autocorrelation index can be used to measure map complexity as perceived by a map user is refuted.

- This means that the dual aims of improving the effectiveness with which maps communicate and maximising a statistic that measures spatial association are not achievable with through single map with this approach. The visualization and communication objectives require different solutions.

- An important index of spatial autocorrelation can be detected from regular matrices, but not from complex sets of zones that contain units of significantly different shape and size. As a result, new techniques are required if map users are to be able to compare the spatial association within and between geographic data distributions, investigate the stationarity of spatially recorded phenomena and identify the strength of association within localities.
The difficulties in relating a measurable statistic to a complex map, which conflict markedly with the visualisation ideology, have long been recognised by thematic cartographers. Muehrcke (1972) stated that "visual map reading does not permit the recognition or differentiation of subtleties in configuration, and rapidly deteriorates in quality as control gets sparse, pattern grows complex and the number of variables increases. For these and other reasons we suspect that the results obtained from visual map reading are often imprecise, inconsistent, or more apparent than real. If these deficiencies were, in fact, real, they would distract significantly from the use of maps in scientific enquiry, especially if they went unrecognised." The need for additional prompts to reinforce patterns, provide alternative numerical and graphical information and specific aids to assist in the interpretation of mapped information are apparent from this statement which questions the most basic tenets of the visualization methodology. It is corroborated by the evidence presented here where spatial data presented on a regular grid proved interpretable, yet the variation in shape and size of a sample enumerated data set added sufficient confusion to the information to make the measured spatial relationships extremely difficult to decipher.

This conflict between statistically measured and visually perceived trends has an important bearing upon the visualisation of irregular area value data. If relatively sophisticated autocorrelation indices do not reflect visually perceived trends then questions arise concerning the appropriateness of the choropleth for communicating geographical pattern through irregular areas for ideation. Single statistics do not provide adequate information from the mapped data and yet an irregular choropleth is too complex for visual analysis. There is consequently a demand for the development of techniques that reflect the strength of spatial associations within complex data sets so that area value data can be analysed effectively. The need for visual aids to interpretation to be simple, bold, clear and adhere to principles of graphic design and communication is re-emphasised by these findings which suggest that the distraction effected by the shape of irregular zones is not insignificant. Dynamic maps, which contain behaviours that respond to requests made by the map user, may provide elegant and pragmatic solutions.
Appendix A2: Notes on Data Assembly for 'panoraMap'

Jason Dykes

Department of Geography
University of Leicester
UK
A2. Notes on Data Assembly for 'panoraMap'

As one of the objectives of panoraMap is the ability to produce virtual environments quickly with minimum effort and few technical skills this section outlines the data and metadata types that are used in creating the virtual environment included on the accompanying CD in appendix B7. It provides some ideas and suggestions about how those data and metadata might be collected.

A2.1 Planimetric Views

The image format used for the base map in 'panoraMap' is 256-colour CompuServe GIF. Any number of these images can be created and viewed in turn for a specified location determined by a common bounding box. All images used in any location should be of the same dimensions and show data of precisely the same areal extent. Other regions within a location are shown with symbols on 'panoraMap' maps. These can be loaded by clicking the symbols, and so geographic zooming and panning between a variety of locations and scales is achieved. Suitable images showing planimetric views are often available on the Internet, or can be purchased from data suppliers. Alternatively they can be scanned or grabbed from spatial information software.

LandSerf (Wood, 1998), which is freely available on the Internet, is particularly good for producing clear and detailed images of elevation models and their characteristics. It reads a whole variety of spatial data formats and the exact spatial extent can be specified in order to crop to a desired location. Under the Windows™ operating systems pressing the <Alt> and <PrtScn> keys together will record the selected window in the Windows™ dipboard from which the required map can be cropped and resized using image/paint software. The dipboard saves information as a 32-bit image. It is essential that users resize and crop their imagery with the full colour table before converting to 8-bit imagery as a final step for use with panoraMap.

A2.2 Notes on Taking Wide Angle and Panoramic Imagery in the Field

A2.2.1 How to take panoramic images

Panoramic images can be collected in the field in at least three ways. Two involve ‘stills’ cameras and software to combine successive images into a panorama, the third creates a single wrapped image from digital video. Here are some thoughts on each.
**Analogue Stills Camera**

Analogue photographs taken at successive angles from a location in the field can be processed, scanned and combined into panoramas. A 28mm lens is preferable, but a 35mm lens will also work. Even with a 28mm lens a 360° panorama requires a lot of film and scanning. About 9 or 10 images are required for a full 360° panorama with the camera held in 'landscape' orientation using a 28mm lens. With an analogue camera it is thus often preferable to photograph smaller sections of the surroundings, as figure A2.1a demonstrates below. The camera should be kept level throughout the panorama as vertical motion will result in a non-rectangular image (see figure A2.2b). Images can also be combined if portrait photographs are taken. These are often more appropriate where the subject of interest or landscape varies in height throughout the panorama such as the image taken of Grasmere from Loughrigg Terrace shown in figure A2.1b. When taking photographs that are going to be combined into a panoramic image be sure to identify the film and record the numbers of the photographs that relate to each panoramic image. Ensure that 10% overlap occurs between successive images.

*Figure A2.1 Panoramic images from an analogue stills camera.*

High definition and resolution images can be obtained by scanning prints from a digital stills camera with a wide angle lens. The effort required to develop and scan images, and the amount of film required to create a 360-degree panorama mean that analogue cameras are best suited for taking pictures where smaller segments of the full field of view are required and resolution is important. These examples show panoramic photographs taken on steep slopes, with near vertical topography to the rear. The top and bottom photographs are of Becka Vale and Haytor Down from Greator Rocks (Dartmoor, UK) and Grasmere from Loughrigg Terrace (The Lake District, UK) respectively.
Digital Stills Camera

A digital camera with a reasonable CCD and large amount of digital storage is the preferred way to record imagery. Low-resolution images such as 512*384 pixels will usually suffice. My own experience with the Casio C-800L Camedia has been very positive. The camera stores 120 images at 512*384 pixel resolution, meaning that 13 full panoramas can be collected before returning to log data. More recent models contain removable storage media reducing the need to download data to the laptop at regular intervals. It is advisable to record the numbers of the images that will go into creating each panorama as they are taken for future reference. As is the case with the analogue camera it is important to keep the digital camera level throughout the panorama and to ensure that an overlap of about 10% occurs between successive images for stitching (see figure 6.3).

Analogue Video Camera

Figure A2.2 Panoramic images from an analogue video camera

Panoramic imagery can be recorded by rotating an analogue video camera smoothly through 360 degrees and then using hardware and software to capture the video digitally. Additional software can then be used to stitch the video into an image strip. These panoramas were taken from the same location in Wistow, Leicestershire, UK, an show successively from top to bottom:

- the perils of capturing moving objects
- that moving the camera vertically during the pan can confuse the stitching software and result in a lot of redundant space in the stitched image, and that changes in light during the pan can produce unwanted contrast levels
- a usable panoramic image from a smooth steady sweep

Images can also be created from analogue video by panning the camera smoothly across the horizon from a fixed position. Software such as 'Panorama' (VideoBrush Corporation, 1999) will digitise the signal from a tape through a video input card and stitch the pan into a single panoramic image (see figure A2.2). The panning must be smooth, as the stitching software can't
cope with sudden movement or changes in direction. Record a 'static' image for two or three seconds at the start and end of each pan that will make the endpoints easier to detect. Do not change focus or zoom whilst recording a panoramic image with a video recorder. As with the stills cameras it is important to avoid vertical motion and recording moving objects as demonstrated in figure A2.2.

A2.2.2 When to take panoramic images

Software that creates panoramic images tends to rely upon edge detection to position the images for stitching. Images where there is a high degree of variation between successive images are thus easier and quicker to stitch than those of homogenous panoramic scenes as often occur on the barren moorland of Dartmoor. The software also expects the edges of the successive images to be similar so try to avoid scenes with much movement or with significant variation in brightness or contrast. This often involves collecting the images that will be wrapped into a panorama rapidly before weather conditions change or elements of the scene move. Quickly moving cloud cover can be a panoramic nuisance!

A2.2.3 What to take

Panoramic imagery can be a useful medium for providing exogenous information at a variety of spatial scales. The size of the images means that it can be difficult to record detail at distance and so a concentration on close features or general trends at distance is recommended. 'panoraMap' uses links between images and other data types to provide detail, so the model for recording greater detail at distance from the location of a panoramic image is to record additional imagery of higher resolution to depict features of particular interest. The software will then produce links between the two collected images. Links between panoramic images work best when panoramas are taken at features or points that are identifiable in other images. The panoramas need to be quite close to one another for this to happen. An area of between three and four square kilometres is a useful benchmark for a 'panoraMap' location. A series of panoramas can be collected to cover such an area on a well planned and busy day. Figure A2.3 shows locations at which panoramic images were collected in a 3.5km square area on a single day with three laptop downloads at the car park located in the north-east. Locations were selected using panoraMap, loaded into a GPS receiver, located in the field and then photographed.
A useful trick is to take a panoramic image whilst standing on a slope, as shown by the top and middle images in figure A2.4. If the camera is held level this results in high-resolution information up-slope from which ground cover can be ascertained and information about larger scale features down-slope. A useful combination of small-scale information for ground-truthing or sampling secondary data sets and large-scale topographic information for navigation and orientation results. Figure 6.2 was taken whilst standing right against Haytor West and includes a section of the granite tor at Haytor West and Haytor East. The view provides information at three spatial scales: micro, with the rock crystals visible in the tor; meso, as the granite feature is clearly visible; macro, with the form of the landscape and distant tors on the horizon.

Figure A2.3 Panoramic image collection - A 3.5km by 3.5km area covered in a single day. (C1)

This figure demonstrates that panoramic imagery can be collected in reasonable time. The 'panoraMap' software is used to illustrate with an aerial photograph showing a 3.5km² area of Holne Moor, Dartmoor, UK. The software was used examine the terrain, features and geography of the area in order to plan suitable locations for panoramas. The Olympus C800L Camedia camera used has a storage capacity of 120 images, enabling 13 panoramas to be collected before logging data to a laptop. Positions at which a car containing a laptop could be parked were also identified. A suitable route that took in each of the selected locations and permitted data logging at the vehicle was then digitised using the software. This was uploaded into a Garmin GPS 12XL unit that was used to identify the positions at which panoramic photographs were recorded on the ground. The white tracks show the route taken on this aggressively undulating moor. The white squares show the locations of panoramas. The process took a little over six hours.
It can be useful to take imagery where the subject matter does not fall completely within the frame. The bottom image in figure A2.4 shows another picture taken on Haytor Down but illustrating a very different physical environment, even though the tree canopies are not visible. Another photograph could be taken of the canopies and the software used to link from the panorama to the canopy image.

Figure A2.4 A selection of 360-degree panoramic imagery, Haytor Down, Dartmoor, UK. (C1)

These images provide an enormous amount of contextual information about the locations at which they were taken. The top two images were photographed on significant slopes and so distant topography can be identified (the lower Bovey Valley in the top image and the upper Bovey with Becka Vale in the middle image) along with features on both far and mid-distant horizons (Saddle Tor and Haytor Rocks in the top image, Hound Tor and Hayne Down in the middle image) and within the direct locality (the car park, eroded footpath and vegetation in the top image, the boulders, vegetation and field systems in the middle image). The bottom image shows how even in relatively enclosed environments, such as the wooded Becka Vale, an impression of both the local environment and its situation can be achieved. Here the level of Becka Brook can be seen, as can evidence of ancient working of the stream. The types of vegetation can be identified as can the health and state of the plants, and Great Tor is visible in the distance.

A2.2.4 What to record

The following data must be recorded when taking digital photographs in order to map the imagery.

- The coordinate at which image was taken:
  - Northing
  - Easting.
- The angles of image:
  - Bearing of the left hand side of the panoramic image in degrees from north
  - Field of view of the panoramic image in degrees

It is useful to use a GPS to mark a waypoint at the location of each image taken in the field as these can be downloaded into panoraMap, viewed and clicked to associate them with the
panorama files. An accuracy of around 50m, which is provided by standard GPS receivers, is adequate for this kind of application although greater accuracy can be gained by using differential GPS equipment. The bearings and field of view can be recorded with an accuracy of +/- 5 degrees. Bearing and field of view data can be recorded using traditional means, but a PalmPilot makes a useful weatherproof addition, particularly with the Stick-e Suite (Pascoe & Ryan, 1998) that links note-taking software with a geo-reference direct from a GPS receiver. Stick-e-Notes can be used to produce a metadata template for any data type with spaces for all metadata fields required for the panoramic images.
Appendix A3:

Selected Scripts from the 'cdv' Software

Jason Dykes

Department of Geography
University of Leicester
UK
# -- Procedure Definition 'read_spatial_data'.
# -- Reads geometry data in Arc/Info ungenerate format (or similar) from file.
# -- Stores data in structure of arrays and lists for visualization.
# -- Checks for relevant images to use as backdrop maps and calculates centroids.
# -- Takes a Single Argument - 'loc': Location name used as basis of filenames.
proc read_spatial_data {loc} {
    # -- Determine global variables and data structures that are to be used within procedure:
    global cdv data datalnstall message poly_coords colour popn name maxmin files colour
    global loc_name location east west north south e_range n_range
    global CANV_X CANV_Y XSIZE YSIZE
    global colour size Item_Type Item_Vars line_width popindex zones centroid
    global zonepolys polys polylD polyNUM datalDs IDs missinglDs link

    # -- Run procedure to find spatial limits of geometric data:
    limits

    # -- Variable identifying colour to fill polygons:
    set colour(fill) $colour(poly)

    # -- Update message in 'cdv' view control panel:
    set message "Reading Spatial Data"
    update

    # -- If there is a link file, open it, read the links and store the values in
    # -- a lookup array so that polygons can be given attribute identifiers:
    if {[file exists $files(link)]} {
        set link_from [open $files(link) r]
        set use_links 1
        while {[gets $link_from line] >=0} {
            set from [lindex $line 0]
            set to [lindex $line 1]
            set link($from) $to
            set link($to) $from
        }
        close $link_from
    } else {
        set use_links 0
    }

    # -- Open file containing boundary geometry:
    set poly_file [open $files(boundary) r]
    # -- Initialise polygon counter:
    set poly 0

    # -- Loop through file line by line, reading lines of characters into
    # -- list 'line' until a line containing zero characters is met.
    while {[gets $poly_file line] >=0} {
        # -- Issue statements depending upon number of words in list 'line':
        switch [llength $line] {
            1 { # -- Data line starts with ID, but no centroids:
                # -- Ignore lines containing the word 'END'.
                if {[string first $line END] < 0} {
                    # -- New polygon detected - increment polygon counter:
                    incr poly

                    # -- Record the case identifier of the new polygon.
                    # -- If a links file exists, set the ID using the lookup array,
                    # -- otherwise use the first word in the data line:
                    if {$use_links == 0} {
                        set id [lindex $line 0]
                    } else {
                        set from [lindex $line 0]
                        set id $link($from)
                    }
                    # -- Uses data IDs to store polygons.
                    set from [lindex $line 0]
                    set id $link($from)
                }
            } else {
                # -- New line contains a list of coordinates:
                # -- Append coordinates to existing polygon:
                foreach coord [split $line ,] {
                    set coord xcoord ycoord
                    set coord [list xcoord ycoord]
                    set poly_coords(poly) [lappend poly_coords(poly) $coord]
                }
            }
        }
    }
}
# -- Append identifier to list of all IDs.
# -- Append polygons number to list of all polygons belonging to case with ID.
lappend zonepolys($id) $poly
lappend ids(all) $id
# -- Record relationship between case ID and polygon number:
set polyid($poly) $id
set polynum($id) $poly

# -- Update message in 'cdv' view control panel:
set message "Reading Zone $id"
update
}

3 {  # -- File starts with ID, AND centroids:
# -- New polygon detected - increment polygon counter:
incr poly

# -- Record the case identifier of the new polygon.
# -- If a links file exists, set the ID using the lookup array,
# -- otherwise use the first word in the data line:
if {$use_links == 0}  {
    set id [lindex $line 0]
}  else {
# -- Uses data IDs to store polygons.
    set from [lindex $line 0]
    set id $link($from)
}
# -- Append identifier to list of all IDs.
# -- Append polygons number to list of all polygons belonging to case with ID.
lappend zonepolys($id) $poly
lappend ids(all) $id
# -- Record relationship between case ID and polygon number:
set polyid($poly) $id
set polynum($id) $poly

# -- Strip centroid coordinates from data line:
set easting [expr double([lindex $line 1])]
set northing [expr double([lindex $line 2])]

# -- Calculate and record centroid in screen coordinates:
set centroid(x,$id) [expr (($easting-$west)/$e_range)*$canv_x.0]
set centroid(y,$id) [expr $canv_y.0-((($northing-$south)/$n_range)*$canv_y.0)]

# -- Store original centroid as some procedures involve varying centroid:
set centroid(x,$id,new) $centroid(x,$id)
set centroid(y,$id,new) $centroid(y,$id)

# -- Store original centroid in World coordinates:
set centroid(x,$id,world) $easting
set centroid(y,$id,world) $northing

# -- Update message in 'cdv' view control panel:
set message "Reading Zone $id"
update
}

2 {  # -- Coordinates to add to a polygon.

# -- Strip coordinates defining point on polygon from data line:
set easting [expr double([lindex $line 0])]
set northing [expr double([lindex $line 1])]

# -- Calculate point in screen coordinates and store in local variable:
set canvas_x [expr (($easting-$west)/$e_range)*$canv_x.0]
set canvas_y [expr $canv_y.0-((($northing-$south)/$n_range)*$canv_y.0)]

# -- Append screen coordinates of point to list containing polygon coordinates:
lappend poly_coords($poly) $canvas_x
lappend poly_coords($poly) $canvas_y
}

# -- Close channel used to read geometry data from file:
close $poly_file
# -- Use utility procedure to shorten list of all case identifiers:
set IDs(all) [list_shorten $IDs(all)]
# -- Make sure that all IDs are selected as the default:
set dataIDs(select) $IDs(all)
# -- If a centroid file exists, with the expected name, read the centroids:
if {{file exists $dataInstall/$loc.cent}} {

    # -- Open file containing centroid details:
    set centroid_from [open $files(centroid) r]
    # -- Loop through file line by line, reading lines of characters into
    # -- list 'line' until a line containing zero characters is met.
    while {[gets $centroid_from line] >=0} {
        # -- Determine the identifier of each line of data.
        # -- If a links file exists, set the ID using the lookup array,
        # -- otherwise use the first word in the data line:
        if ($use_links == 0) {
            set id [lindex $line 0]
        } else {
            set id $link($from)
        }
        # -- Strip centroid coordinates from data line:
        set easting [expr double([lindex $line 1])]
        set northing [expr double([lindex $line 2])]
        # -- Calculate and record centroid in screen coordinates:
        set centroid(x,$id) [expr ((easting-$west)/$e_range)*$CANV_X.0]
        set centroid(y,$id) [expr $CANV_Y.0-( (northing-$south)/$n_range)*$CANV_Y.0]
        # -- Store original centroid separately as some procedures involve varying centroid:
        set centroid(x,$id,new) $centroid(x,$id)
        set centroid(y,$id,new) $centroid(y,$id)
        # -- Store original centroid in World coordinates:
        set centroid(x,$id,World) $easting
        set centroid(y,$id,World) $northing
    }
    # -- Close channel used to read centroid data from file:
    close $centroid_from
}
# -- Create centroids if array element does not exist for any case identifier:
foreach id SlDs(all) {
    switch {[catch [set centroid(x,$id)]]} {
        0 { }
        default {
            # -- Numeric value returned from 'catch' statement indicates error.
            # -- Use 'calc_centroid' procedure to determine centroid:
            calc_centroid $id
        }
    }
    # -- Initialise list of missing variables for each id in preparation for
    # -- reading of data file:
    set missingIDs(id$id) ""
}
# -- Store number of polygons read in global variable:
set polys Spoly
# -- Update message in 'cdv' view control panel:
set message "Read $polys polygons"
update
}

# -- Procedure Definition 'calc_centroid'.
# -- Calculates a default for any case identifier that has a geometry but no centroid.
# -- Takes a Single Argument - 'id': Case identifier of polygon without centroid
```java
proc calc_centroid {id} {
    # -- Determine global variables and data structures that are to be used within procedure:
    global centroid poly_coords zonepolys message
    # -- Initialise local variables to contain sum of x and y coordinates and number
    # -- of points in polygon so that mean location can be calculated:
    set xtotal 0
    set ytotal 0
    set N 0
    # -- Update message in 'cdv' view control panel:
    set message "Calculating centroid $id"
    update
    # -- Loop through each polygon used to represent identified case:
    foreach poly $zonepolys($id) {
        # -- Loop through points in coordinate list:
        foreach "x y" $poly_coords($poly) {
            # -- Add coordinates of point to variables containing totals of 'x' and 'y'.
            set xtotal [expr $xtotal + $x]
            set ytotal [expr $ytotal + $y]
            # -- Increment variable containing number of points by number of coordinate pairs:
            incr N [llength $poly_coords($poly)/2]
        }
    }
    # -- Calculate mean x and y coordinates.
    # -- Store in appropriate array referenced by case identifier:
    set centroid(x,$id) [expr double($xtotal/$N.0)]
    set centroid(y,$id) [expr double($ytotal/$N.0)]
}

# -- Procedure Definition 'limits'.
# -- Calculates spatial limits of selected geometry file and appropriate canvas size.
# -- Stores information in 'cdv' data structure.
# -- 'limits' Defined with No Arguments.
proc limits {} {
    # -- Determine global variables and data structures that are to be used within procedure:
    global north south east west e_range n_range files message
    global CANV_X CANV_Y XSIZE YSIZE cdvHome
    # -- Relies upon existence of 'files(boundary)' variable and file that it specifies.
    # -- Otherwise an error message is produced:
    if {[file exists $files(boundary)]} {
        # -- Open channel for reading boundary file:
        set poly_file [open $files(boundary) r]
    } else {
        # -- Message to screen:
        tk_messageBox -icon error -title "Fatal Error" -message "Can't find file $files(boundary)"
    }
    # -- Update message in 'cdv' view control panel:
    set message "Calculating Spatial Limits"
    update
    # -- Initialise variables to contain geographic extents with extreme values:
    set north -9999999
    set east -9999999
    set south 9999999
    set west 9999999
    # -- Loop through file line by line, reading lines of characters into
    # -- list 'line' until a line containing zero characters is met.
    while {[gets $poly_file line] 
    switch [llength $line] {
```
read_spatial

2 { # -- Point defining polygon boundary detected: Strip out coordinates.
   set x [lindex $line 0]
   set y [lindex $line 1]
   # -- Check against current boundaries and record if more extreme:
   if {$x < $west} { set west $x }
   if {$x > $east} { set east $x }
   if {$y < $south} { set south $y }
   if {$y > $north} { set north $y }
 }

3 { # -- Point defining polygon centroid detected: Strip out coordinates.
   set x [lindex $line 1]
   set y [lindex $line 2]
   # -- Check against current boundaries and record if more extreme:
   if {$x < $west} { set west $x }
   if {$x > $east} { set east $x }
   if {$y < $south} { set south $y }
   if {$y > $north} { set north $y }
 }

# -- Calculate percentage of range for use as border:
set nedge [expr ($north-$south) * 0.025]
set eedge [expr ($east-$west) * 0.025]

# -- Expand geographic extremities by calculated border:
set north [expr $north + $nedge]
set south [expr $south - $nedge]
set east [expr $east + $eedge]
set west [expr $west - $eedge]

# -- Determine ranges in north/south and east/west directions:
set e_range [expr $east - $west]
set n_range [expr $north - $south]

# -- Close channel used to read geometry data from file:
close $poly_file

# -- Calculate size of canvas based upon global variable specifying desired scale
# -- and calculated height:width ratio of geographic region:
if {$e_range > $n_range} {
   set CANV_Y [expr int(($n_range/$e_range) * $YSIZE)]
   set CANV_X $XSIZE
} else {
   set CANV_X [expr int(($e_range/$n_range) * $XSIZE)]
   set CANV_Y $YSIZE
}
# -- 'cdv' - the Cartographic Data Visualizer.
# -- Jason Dykes, University of Leicester, 1995-1999 - jad7@le.ac.uk
# -- Tcl Script 'draw_polys.tcl' - Contains a single definition: 'draw_polys'

# -- Procedure Definition 'draw_polys'.
# -- Draws polygons representing specified cases on canvas and symbolises variable.
# -- Takes 3 Arguments...
# -- 'c': Name of Tcl/Tk canvas.
# -- 'v': Number of variable to be mapped in 'cdv' data structure.
# -- 'IDs': List of case identifiers to be symbolised.
proc draw_polys {c v {IDs}} {

# -- Determine global variables and data structures that are to be used within procedure:
global colour item_col plot_info zonepolys poly_coords missingIDs datalDs

global image location CANV_X CANV_Y XSIZE

# -- Run procedure to check that statistics have been calculated for current selection
# -- of case identifiers (stored in 'datalDs(select)'):
summary_stats $c $v "$IDs"

# -- Draw polygons that do not have data values.
# -- Set variable containing colour with which to fill polygons to empty string,
# -- which will result in transparency:
set colour(fill) {""

# -- Loop through case identifiers of polygons with missing values:
foreach id $plot_info($c,IDmissing) {
# -- Loop through list of polygons representing case:
foreach poly $zonepolys($id) {
    # -- Create polygon.
    # -- Coordinates from geometry data array, tags identify case ID and polygon number:
    eval " $c create polygon $poly_coords($poly) -fill $colour(fill) -outline $colour(outline,0) -width 1 -tags {id$id poly$poly map polygon unselected}"

    # -- Record colour used to fill item in array structure by canvas and identifier:
eval set item_col($c,id$id) $colour(fill)
}
}

# -- Draw the selected polygons that do have data values.
# -- Set variable containing colour with which to fill polygons:
set colour(fill) SeaGreen2

# -- Loop through case identifiers of polygons:
foreach id $IDs {
    # -- Loop through list of polygons representing case:
    foreach poly $zonepolys($id) {
        # -- Create polygon.
        # -- Coordinates from geometry data array, tags identify case ID and polygon number:
        eval " $c create polygon $poly_coords($poly) -fill $colour(fill) -outline $colour(outline,1) -width 1 -tags {id$id poly$poly map polygon}"

        # -- Record colour used to fill item in array structure by canvas and identifier:
        set item_col($c,id$id) $colour(fill)

        # -- Add tags to identify items that have missing values for this variable:
        foreach n $missingIDs(id$id) {
            $c addtag mv$n withtag poly$poly
        }
    }

    # -- Scale polygons by the canvas size:
    $c scale map 0 0 $plot_info($c, SF) $plot_info($c, SF)
}
# -- 'cdv' - the Cartographic Data Visualizer.
# -- Jason Dykes, University of Leicester, 1995-1999 - jad7@le.ac.uk
# -- Tcl Script 'data.tcl' - Contains a single procedure definition: 'read_attributes'
# -- Procedure Definition 'read_attributes'.
# -- Takes a single argument - 'loc':
# -- Selects and loads attribute data from correctly named and located data file.
# -- Define global variables and data structures that are to be used within procedure:
# -- Determine global variables and data structures that are to be used withing procedure:
global first_var last_var D data files cdv key keys dataInstall
# -- Update message in 'cdv' view control panel:
set message "Reading Attributes..." update
# -- Initialise list of attribute keys (names corresponding to attribute numbers):
set keys ""
# -- Check for existence of attribute keys file and read if it exists:
if [file exists $files(key)] { 
    # -- Open channel for reading data file:
    set key_from [open $files(key) r]
    # -- Loop through file line by line, reading lines of characters into while { [gets $key_from line] >=0 } {
        # -- Strip attribute number and name from data line:
        set kl [lindex $line 0]
        set k2 [lindex $line 1]
        # -- Record attribute name in lookup array:
        set key($kl) $k2
        # -- Record attribute number in list:
lappend keys $kl
    }
    # -- Close channel used to read data from file:
    close $key_from
}
# -- Check for existence of attribute data file and read if it exists:
if ![file exists $files(attribute)] { 
    # -- Open channel for reading data file:
    set attribute [open $files(attribute) r]
    # -- Read first line from file as sample to be used to determine size of data array:
    gets $attribute sampleLine
    # -- Close channel for reading data file:
    close $attribute
    # -- Re-Open channel for reading data file:
    set attribute [open $files(attribute) r]
} else {
    # -- Relies upon existence of 'files(attribute)' variable and file that it specifies.
    # -- Otherwise an error message is produced and the program terminated:
    tk_messageBox -icon error -title "Fatal Error" -message "Fatal Problem! Can't find Attribute table file $files(attribute). cdv will terminate."
    # -- Terminate program:
    exit
}
# -- If no attribute keys file exists ('.ATT' file) generate attribute keys:
if ![file exists $files(key)] { 
    set last_var [expr [llength $sampleLine] - 3 + $first_var]
    for {set n $first_var} {$n <= $last_var} {incr n} {
        if ![lsearch $keys $n] <0 { 
            set key($n) "Variable $n"
lappend keys $n
        }
    }
}
# -- Initialise variable that counts through number of zones:
set zones 0

# -- Loop through attribute file line by line, reading lines of characters into
# -- list 'line' until a line containing zero characters is met.
while { [gets $attribute line] >= 0 } {
    # -- Increment zone counter as each line represents data for a case:
    incr zones
    # -- Read case identifier as first word in data line:
    set id [lindex $line 0]
    # -- Read case name as second word in data line:
    set idname [lindex $line 1]
    # -- Store case name and identifier in lookup array:
    set name($id) $idname
    set nametoID($idname) $id
    # -- Add case identifier to array containing list of all case.
    lappend dataIDs(all) $id

    # -- Calculate column numbers of first and last variables in data array:
    set last_var [expr [llength $line] - 3 + $first_var]
    # -- Loop through words representing columns in data line:
    for [set n $first_var} [($n <= $last_var} [incr n} { 
        # -- Identify location of word that represents current data column in data line:
        set index [expr $n - $first_var +2]
        # -- Determine number of variable from list of keys:
        set k [lindex $keys [expr $n -1]]
        # -- Strip word that contains data value from data line:
        set word [lindex $line $index]
        # -- Use 'catch' to identify strings as opposed to numeric values.
        # -- Store information about missing values with structure identified below:
        # -- dataIDs($k) - List of ids that have data for variable $k.
        # -- missingIDs($k) - List of ids that don't have data for $k.
        # -- missingIDs(id$id) - List of variables ids that don't have data for $id.
        switch [catch [set number [expr double($word)]]] {
            0 {  # -- No error - word is numeric:
                # -- Store double precision data value in array referenced by case identifier:
                set data($k,$id) [format %g [expr double($number)]]
                # -- Add case identifier to list of identifiers with data for variable 'k':
                lappend dataIDs($k) $id
            }
            default {  # -- Error - word is non-numeric:
                # -- Store word 'NONE' to represent data value in array referenced by case ID:
                set data($k,$id) NONE
                # -- Add case identifier to list of identifiers without data for variable 'k':
                lappend missingIDs($k) $id
                # -- Add variable to list of attributes without data for particular case:
                lappend missingIDs(id$id) mv$id
            }
        }
    }
    # -- Close channel used to read attribute data from file:
    close $attribute

    # -- If any cases are identified without missing values record this with an empty string:
    foreach id $dataIDs(all) {  
        if ![info exists missingIDs(id$id)] { set missingIDs(id$id) "" }
    }

    # -- Create list of all case identifiers without data values (missing zones):
    set missingIDs(all) [missing_list $IDs(all) $dataIDs(all)]

    # -- Shorten list of case identifiers with utility list procedure:
    set missingIDs(all) [list_shorten $missingIDs(all)]

    # -- Make list of missing values for each zone for which all data are missing:
    foreach id $missingIDs(all) { 
        set missingIDs(id$id) $keys
    }
}

Interactive Maps for Exploratory Spatial Data Analysis
# -- Create missings list for all other variables as above:
foreach n $keys {
    # -- Update message in 'cdv' view control panel:
    set message "Checking... $key($n)"
    update
    set missingIDs($n) [missing_list $IDs(all) $dataIDs($n)]
    set missingIDs($n) [list_shorten $missingIDs($n)]
}

# -- Update message in 'cdv' view control panel:
set message "Initialising Maps...\nPlease be patient."
update

# -- Define global variable that specifies name of base map:
set map_name .polygon_map1.canvas

# -- Procedure Definition 'missing_list'.
# -- Procedure compares two lists.
# -- Returns elements of second list that do not occur in first:
# -- Takes 2 Arguments...
# -- '11': First list
# -- '12': Second list
proc missing_list {11 12} {
    # -- Initialise local variable to contain solution list:
    set 13 ""
    # -- Loop through elements in first list:
    foreach i $11 {
        # -- Determine location of list element in second list:
        set index [lsearch $12 $i]
        # -- If the element does not exist, append it to the solution list:
        if {$index <0} {
            lappend 13 $i
        }
    }
    # -- Return the solution list:
    return $13
}
# -- Procedure Definition 'summary_stats'.
# -- Calls procedures that calculate summary statistics from data array if those
# -- statistics have not already been calculated.
# -- Takes 3 Arguments...
# -- 'c': Name of Tcl/Tk canvas by which to reference statistics.
# -- 'var_list': List of variables for which to calculate statistics.
# -- 'IDs': List of case identifiers to include in calculations.
proc summary_stats {c {var_list} {IDs}} {
    # -- Determine global variables and data structures that are to be used within procedure:
    global data dataIDs
    # -- Loop through list of variables:
    foreach v $var_list {
        # -- Check for existence of variable that identifies that stats previously calculated:
        switch {[catch {set data($c,$v,variable_stats) } 0] } {
            0 {  # -- No action: Statistics exist for variable '$v'. }
            default {
                # -- Run procedure that calculates summary statistics for current variable '$v':
                variable_stats $c $v $IDs
            }
        }
    }
    # -- Calculate statistics for bivariate views:
    if {([llength $var_list] >1)} {
        # -- Strip first and second element of list for use in bivariate views:
        set vx [lindex $var_list 0]
        set vy [lindex $var_list 1]
        # -- Check for existence of variable that identifies that ratio statistics calculated:
        switch {[catch {set data($c,$vx,$vy,ratio_stats)}]} {
            0 {  # -- No action: Ratio statistics exist for '$vx' and '$vy':
                default { ratio_stats $c $vx $vy $IDs }
            }
        }
    }
}

# -- Procedure Definition 'ratio_stats'.
# -- Scales pairs of data columns to absolute values so two variables can be compared.
# -- Used in creating ratio scaled scatter plots.
# -- Takes 4 Arguments...
# -- 'c': Name of Tcl/Tk canvas by which to reference statistics.
# -- 'vx': Number of first data column to be considered.
# -- 'vy': Number of second data column to be considered.
# -- 'IDs': List of case identifiers to include in calculations.
proc ratio_stats {c vx vy IDs} {
    # -- Determine global variables and data structures that are to be used within procedure:
    global data dataIDs
    # -- Determine range of data values to be used as common scaling factor:
    set ratio [expr $data($c,$vx,range)/$data($c,$vy,range)]
    # -- Record largest data range as common scaling factor:
    if {$ratio >1} {
        set data($c,$vx,$vy,range) $data($c,$vx,range)
    } else {
        set data($c,$vx,$vy,range) $data($c,$vy,range)
    }
    # -- Perform calculations for both variable columns: '$v' is primary variable.
    foreach v "$vx $vy" {
        # -- Set '$sv2' to be other variable in list.
        if {$v == $vx} { set v2 $vy }
        if {$v == $vy} { set v2 $vx }
        # -- Loop through all cases:
    }
}
foreach id $IDs {
    # -- Calculate ratio value for each zone using absolute data range:
    set data($c,$v,$v2,$id,ratio) \ 
        [expr ($data($v,$id) -$data($c,$v,min) ) / $data($c,$v,$vy,range)]
}

# -- Set flag to show that ratio statistics have been calculated for "$vx" and "$vy"
# -- for the specified list of cases and canvas.
set data($c,$vx,$vy,variable_stats) 1

# -- Procedure Definition 'variable_stats'.
# -- PROC-DESCRIPTION...
# -- Takes 3 Arguments...
# -- 'c': Name of Tcl/Tk canvas by which to reference statistics.
# -- 'v': Number of variable column in data array for which to calculate stats.
# -- 'ids': List of case identifiers to include in calculations.
proc variable_stats {c v ids} {
    # -- Determine global variables and data structures that are to be used within procedure:
    global data dataIDs missingIDs parallel dataInstall dataLocal location files
    # -- Initialise array elements to contain maximum and minimum values:
    set data($c,$v,max) -99999999
    set data($c,$v,min) 99999999
    # -- Record that statistics have been calculated for this variable,
    set data($c,$v,variable_stats) 1
    # -- Calculate and store data mean.
    # -- Initialise variable containing number of cases, and to contain column total,
    set n [expr double([llength $ids])]
    set data($c,$v,total) 0
    # -- Issue data discrepancy warning if no case identifiers occur in the list:
    if {$n == 0} {
        set n 1
        tk_messageBox -icon warning -title "cdv Data Discrepancy:" -message "
        Attribute IDs and Spatial Data IDs do Not Match.
        To map the data that you have loaded, the polygons must be related to the
        attributes.
        To match polygon IDs in $files(boundary) with attribute IDs in $files(attribute),
        use a link file called "$files(link)".
        E.g. to match
        \Attribute IDs: [lrange $dataIDs(all) 0 4]...
        \Polygon IDs: [lrange $IDs(all) 0 4]...
        Use a file with two columns that relates a polygon ID with the appropriate attribute ID. \n        Format:
        \[lindex $IDs(all) 0] [lindex $dataIDs(all) 0] [lindex $IDs(all) 1] [lindex $dataIDs(all) 1]
        \[lindex $IDs(all) 2] [lindex $dataIDs(all) 2] [lindex $IDs(all) 3] [lindex $dataIDs(all) 3]
        \[lindex $IDs(all) 4] [lindex $dataIDs(all) 4]
        ...
        "
    }
    # -- Loop through each case in data column:
    foreach id $ids {
        # -- Find and record maximum and minimum in data column:
        if {[$data($v,$id)] > $data($c,$v,max)} { set data($c,$v,max) $data($v,$id) }
        if {[$data($v,$id)] < $data($c,$v,min)} { set data($c,$v,min) $data($v,$id) }
        # -- Increment column total by case value:
        set data($c,$v,total) [expr $data($c,$v,total) + $data($v,$id)]
    }
    # -- Calculate data column mean:
    set data($c,$v,mean) [expr $data($c,$v,total)/$n]
    # -- Calculate data column range for scaling:
    set data($c,$v,range) [expr double($data($c,$v,max) - $data($c,$v,min))]
    # -- Calculate variance and standard deviation.
    # -- Initialise variable to contain total variation in column:
    set total_varn 0
}
# -- Initialise list of all scaled values for data column.
# -- It is used to create a sorted list from which quartiles can be found.
set parallel($c,$v,z) ""

# -- Loop thought all case identifiers:
foreach id $ids {
    set data($c,$v,$id,scale) \n    [expr ( $data($v,$id)-$data($c,$v,min) ) / $data($c,$v,range)]

    # -- Add the value to the list of scaled values:
    lappend parallel($c,$v,z) $data($c,$v,$id,scale)

    # -- Calculate the variation from the mean and add this to the sum of variations:
    set diff [expr $data($v,$id)-$data($c,$v,mean) ]
    set total_varn [expr $total_varn + ($diff*$diff)]
}

# -- Calculate and store the variance and standard deviation from the sum of variations:
set data($c,$v,variance) [expr $total_varn/$n]
set data($c,$v,stdev) [expr sqrt($data($c,$v,variance))]

# -- Add missing value tags to all symbols without values in the current view:
foreach id $missingIDs($v) {
    $c addtag mv$v withtag id$id
}

Interactive Maps for Exploratory Spatial Data Analysis
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

Appendix A4:
Scripts that Extend 'cdv'
Functionality

Jason Dykes

Department of Geography
University of Leicester
UK
Script 'mapAbsoluteVals.tcl'

--- Procedure Definition 'mapAbsoluteVals'.
--- Creates colour composite with absolute, rather than relative, data values for three
--- selected variables so that variations between, rather than within, variables can
--- be assessed using visual means (see figure 3.25).
--- Takes 3 Arguments...
--- 'c': Name of Tcl/Tk canvas on which data are plotted.
--- 'vList': List of variable numbers.
--- 'ids': List of identifiers of cases to be considered.

```tcl
proc mapAbsoluteVals {c vList ids} {
    # -- Determine global variables and data structures that are to be used within procedure:
    # global data max min range
    # -- Run procedure that calculates absolute data values in relation to variable list:
    calcAbsoluteVals c vList ids
    # -- Loop through case identifiers in list determined by procedure argument:
    foreach id $ids {
        # -- Calculate RGB composite from three selected variables.
        # -- Split variable list to identify three variables:
        set vl [lindex $vList 0]
        set v2 [lindex $vList 1]
        set v3 [lindex $vList 2]
        # -- Set colour component to hexadecimal value, 00-ff scaled by data range
        # -- of variables in list:
        set red [format %.2x [expr round($data($c,$vl,$id,scale)*255)]]
        set grn [format %.2x [expr round($data($c,$v2,$id,scale)*255)]]
        set blu [format %.2x [expr round($data($c,$v3,$id,scale)*255)]]
        # -- Collate components to produce RGB colour:
        set rgb "#${red}${grn}${blu}"
        # -- Re-configure item on canvas with case identifier in specified colour:
        $c itemconfigure id$id -fill $rgb
    }
}
```

--- Procedure Definition 'calcAbsoluteVals'.
--- Transforms data values on to absolute scale for comparison.
--- Takes 3 Arguments...
--- 'c': Name of Tcl/Tk canvas on which data are plotted.
--- 'vList': List of variable numbers.
--- 'ids': List of identifiers of cases to be considered.

```tcl
proc calcAbsoluteVals {c vList ids} {
    # -- Determine global variables and data structures that are to be used within procedure:
    # global data max min range
    # -- Call procedure to find absolute range of values of all variables in selected list:
    getAbsoluteRange c vList ids
    # -- Loop through case identifiers in list determined by procedure argument:
    foreach id $ids {
        # -- Loop through variables in list determined by procedure argument:
        foreach v $vList {
            # -- Calculate absolute values with determine minimum and range.
            # -- Store in data array structure located by canvas, variable and identifier:
            set data($c,$v,$id,scale) [expr ($data($v,$id)-$min)/$range]
        }
    }
}
```

--- Procedure Definition 'getAbsoluteRange'.

--- Example script to produce absolute colour shadings for selected variables that
--- record percentages of population with certain characteristics.
--- Enables ratios to be compared in absolute, rather than relative terms.
--- Script relies on data, procedures and structures defined by 'cdv' software.
# — Finds absolute data range from series of variables:
# — Takes 2 Arguments...
# — 'vList': List of variable numbers.
# — 'ids': List of identifiers of cases to be considered.
proc getAbsoluteRange {vList ids} {

# -- Determine global variables and data structures that are to be used within procedure:
global data max min range

# -- Initialise variables to contain maximum and minimum values.
# Variables supported are percentage ratios, hence range from 0-100.
set max -100
set min 100

# -- Loop through variables in list determined by procedure argument:
foreach v $vList {

# -- Loop through case identifiers in list determined by procedure argument:
foreach id $ids {

# -- Record maximum and minimum in array if magnitude is exceeded by current value:
if {$data($v,$id) > $max} {  set max $data($v,$id) }
if {$data($v,$id) < $min} {  set min $data($v,$id) }
}
}

# -- Calculate range and store in global variable:
set range [expr double($max-$min)]
}

# -- Run procedure with three selected variables and all case identifiers:
mapAbsoluteVals $map_name "56 58 59" $dataIDs(all)
'selectDistance.tcl'

# — 'cdv' - the Cartographic Data Visualizer.
# — Jason Dykes, University of Leicester, 1995-1999 - jad7@le.ac.uk
# — Tcl Script 'cdvEx2.tcl' - Contains a single procedure definition: 'selectDistance'
# — Example script lists and highlights all polygons within certain distance
# — of selected zone.
# — Script is used to brush views spatially based upon metric distance inclusion.
# — Procedure could be bound to mouse/cursor action for interactive query.
# — Script relies on data, procedures and structures defined by 'cdv' software.

# — Procedure Definition 'selectDistance'.
# — Returns list of case identifiers whose centroids fall within certain distance
# — of specified case.
# — Takes 2 Arguments...
# -- 'ID': Unique identifier of subject case.
# -- 'd': Distance from centroid of case that is used to check for inclusion.
proc selectDistance (ID d) {
    # -- Determine global variables and data structures that are to be used within procedure:
    global datalDs centroid
    # -- Set local variables to contain x and y coordinates of centroid of specified zone.
    set xC Scentroid(x,$ID,World)
    set yC Scentroid(y,$ID,World)
    # -- Initialise an array element to hold the list of zone ids within specified distance:
    set select($ID,$d) ""
    # -- Loop through the list of cases to check the distance:
    foreach id SdatalDs(all) {
        # -- Set local variables to contain zone centroid, based upon 'cdv' data structures:
        set x $centroid(x,$id,World)
        set y $centroid(y,$id,World)
        # -- Calculate distance between selected case and specified case:
        set dX [expr $xC -$x]
        set dY [expr $yC -$y]
        set h [expr hypot($dX,$dY)]
        # -- Check this distance against specified distance:
        if {$h <= $d} {
            # -- If inter-zone distance less than that specified,
            # -- append local case identifier to list of included zones.
            lappend select($ID,$d) $id
        }
    }
    # -- Return list of cases with centroids within specified distance:
    return Sselect($ID,$d)
}

# — Script that uses 'selectDistance' procedure as specified above.
# — 'selectDistance' returns list of all identifiers within 2500 metres of case 99.
# — This list is used to highlight the symbols representing cases on existing views,
# — the base map, a cartogram and a parallel plot (see figure 3.26).
foreach id [selectDistance 99 2500] {
    # -- Raise symbols with the unique case identifier returned from the list on the
    # -- base map and an existing population cartogram
    $map_name raise id$id
    .cartogram1.canvas raise id$id
    # -- Reconfigure symbols with the unique case identifier returned from the list
    # -- on the base map, existing population cartogram, and a parallel plot so that
    # -- they are highlighted.
    $map_name itemconfigure id$id -fill Orange
    .cartogram1.canvas itemconfigure id$id -fill Orange
    .parallel4.canvas itemconfigure id$id -width 2 -fill Orange
}
Appendix A5:
Virtual Reality and Visualization in the Fieldwork Context

Jason Dykes
Department of Geography
University of Leicester
UK
A5. Virtual Reality and Visualization in the Fieldwork Context

This appendix provides a more detailed account of the educational rationale behind the development of the 'panoraMap' software and its utility for and application to teaching fieldwork. It is included so that the reader is familiar with the broader context within which the software was developed. The program was developed in part to satisfy objectives of the Virtual Field Course (VFC) Project, an inter-departmental effort being funded by the Higher Education Funding Council for England (HEFCE) Joint Information Systems Committee (JISC). The project is addressing the use of virtual environments and information technology in teaching fieldwork. Teachers and researchers at the University of Leicester, Birkbeck College and Oxford Brookes University in the UK are approaching the subject through a number of channels. This appendix expands upon the information provided in section 6.1.1 and relates closely to the arguments that have been presented to the academy and remain in the public domain (Dykes et al., 1999).

A5.1 Fieldwork in the UK

Fieldwork is a multidisciplinary exercise that takes an important place in the curriculum of subjects such as geology, biology, geography, planning and architecture in the UK. It offers substantive experience of the world outside of the classroom and is often residential, providing a unique intensive educational experience where many aspects of a region can be analysed first hand. The 1995 UK Geography Teaching Quality Assessment (HEFCE, 1995) recognised that fieldwork adds "an important dimension to the curriculum". In the field, students can gain proficiency in a number of domains including the development of observational skills, learning about the environment from direct experience of it, and designing, conducting and reporting upon practical projects. These can be achieved through popular teaching modes such as excursions with instructors who identify items of interest and group work where students participate in the collection, integration and analysis of data addressing a specific issue. Fieldwork occurs on a grand scale, for example, Departments of Geography alone in the UK offer 144 degree courses that produce 2500 graduates per annum. Students on such courses typically undertake two residential excursions.

The VFC project undertook a survey and assessment of the aims and objectives of fieldwork (Williams et al., 1997) and an evaluation of fieldwork destinations (VFC, 1998). Subsequently generic software functionality and meaningful locations and data sets were identified to support the achievement of the reported aims and objectives and prototype software developed.
A substantial literature exists on the use of fieldwork for pedagogic purposes. Four stages of activity in the use of fieldwork I teaching are identified Kent et al. (1997):

1. 'Preparation and briefing' when students are made aware of the academic context in which the fieldwork will take place along with the objectives and the logistics. Students might also be sensitised to the fieldwork environment.

2. 'Engagement in the activity' when students are usually required to determine a suitable procedure for collecting information in the field and undertake some form of data gathering.

3. 'Processing the results of the activity' when the collected data are combined with other information and analysed in context.

4. 'Debriefing and feedback' when a review of the theoretical background and student experience in the field are combined.

Various accounts exist of using information technology in each of these stages. Digital media incorporated into preparation and briefing can inform students about logistical and safety issues, and also reduce the 'novelty space' that can hinder the student fieldwork experience (Warburton & Higgitt, 1997). A multitude of hypertext/WWW based 'field trips' with relatively low levels of interaction and utility of dynamic maps exist to fulfil this role (for examples see Ritter, 1998). The engagement stage might involve identifying measurement sites for student sampling at the start of a day's fieldwork: Gardiner and Unwin (1986) compute a stratified random technique while Warburton and Higgitt (1997) employ GIS to identify suitable locations. The processing of results can employ statistical (Gardiner & Unwin, 1986) or GIS (Warburton & Higgitt, 1997) software and communications technology (Williams et al., 1997), all of which can be equally useful media for debriefing, feedback and presentation.

A number of fieldwork teachers were surveyed to ascertain the desired educational aims and objectives of a field course (Williams et al., 1997). Enabling students to perform 'comparison' was distinguishable as a key objective, both between geographic locations, models (including maps) and reality and student-collected data and secondary sources. Developing 'observation' skills by enabling students to interpret the natural environment was another. A third requirement was the instruction of students in techniques that would provide them with 'transferable skills'. In light of these results an assessment of current and likely future technology led to the identification of desirable software functionality with a particular focus on virtual environments (Moore, 1997) and visualization. These themes compliment both the teaching and spatial aspects of fieldwork, support analysis and may provide transferable skills. The potential use of digital collection and logging devices during the activity stage was also considered and so applications that take advantage of
developments in this area by incorporating these new real-time data sources were regarded as being desirable.

A5.3 Virtual Reality for Learning

The process of academic learning is described by Laurillard (1997) through a 'conversational framework' containing discursive, adaptive, interactive and reflective types of learning activity. Ideas are only fully learned by students experiencing them through interactive operation on the real world, reflecting upon this experience to inform their communication with the teacher, and using this dialogue to adapt the way in which they operate on the world. The approach is particularly appropriate in fieldwork where the student-centred project is popular with the teacher acting as facilitator to guide the student's ideas and help them formulate their methodology, analysis and interpretation. This framework rationalises the requirement for student observation and comparison in teaching outlined above and goes a long way to explaining why the field course is such an intensive and valuable educational experience. Computer technology can be used to provide students with such a means for interacting with models of reality and providing adaptive feedback (Laurillard, 1997). The degree of interaction can also be controlled dependent upon function or the users skill level.

This approach is also analogous to the iterative 'visual thinking' model of the process of knowledge acquisition identified by MacEachren (1994) and can take advantage of very realistic and highly interactive maps. These are particularly relevant in fieldwork where immersion in the environment is one of the features that enables students to fulfil some of the stated objectives and high levels of interaction and self-lead learning are appropriate for the teaching mode.

An interface that is sufficiently immersive that the map image perceived by a user (see figure 1.1) operates in such a way that they perform and respond as though they were within the representation can be termed Virtual Reality or VR (see figure 6.3). Evidently degrees of VR exist. In its most convincing form VR provides the characteristics of autonomy, presence and place (Zeltzer, 1992). Educators can take advantage of these to extend learning techniques and to improve student's understanding and performance (Winn, 1997). Consequently the rate of uptake of this emergent technology is rapid (Emerson & Revere, 1997). VR is particularly applicable for education in the spatial sciences as the very spatiality of VR can convey a greater cognition of place (Boyd-Davis et al. 1996). It can provide an educational environment with high levels of interaction and real world interface affordances that supports the methods of learning identified above, whilst supporting the listed objectives of fieldwork.
A5.4 Visualization for Learning

Providing highly interactive maps for use by individuals to acquire information that is unknown to them in the method of map use termed ‘visualization’ by MacEachren (1994) provides huge potential for adopting Laurillard’s model for learning.

Indeed ‘visualization’ in its broadest sense is a key theme in the VFC. Methods of visual enquiry add an engaging element of realism (Moore, 1997), provide a formal model in familiar form (Laurillard, 1997) and supply an immediate integrated spatial view of a collected data set for analysis. The rapidity associated with user queries and map response that is one of the defining factors of ‘visualization’ can also be used as a metaphor. Just as computer technology has been shown to have increased the speed at which human-map interaction can occur, resulting in highly interactive ‘visual thinking’ or visualization as a viable aid to the process of ideation, so reducing the time between the student’s collection and analysis of data is likely to have considerable benefits in the analysis of data collected in the field and so in their learning. Gardiner and Unwin (1986) note that it is difficult to rekindle student enthusiasm on returning from a field trip and that the full investigative cycle from experimental design to the presentation of conclusions is best done whilst fresh in the student’s minds. Time is often short after a day in the field and data can be synthesised and ideas generated quickly with interactive visual techniques. Visual representations can be especially successful if sample data are collected by different groups of students over a large area as seeing an integrated whole provides a feeling of ‘ownership’ of the problem which can enthuse them (Gardiner & Unwin, 1986). Immediacy, integration, realism and ownership can be successful motivators even after a hard day in the field. In combination with the novelty and levels of engagement offered by visualization as a data processing and interpretation technique they form a potent combination when applied to the analytical stage of fieldwork. Haigh and Gold (1993) identify this stage as the most difficult for students and one that is often neglected, meaning that the time, resources and effort put into undertaking the fieldwork and collecting data is often under-utilised or wasted.

A5.5 VFC Software Objectives

An understanding of the educational rationale behind the use of spatial and visual information technology in fieldwork support gives rise to a number of more specific objectives adopted by the VFC project in general. These in turn were used to guide the development of software, data collection and applications.

In particular, the software should be able to:
• support visualization by providing a software environment for the interactive visual exploration of spatially referenced information;
• provide and support the development of a shared library of spatially referenced data to support field-based activity;
• enhance and extend all stages of field-based activity with appropriate teaching and learning materials;
• use affordable and accessible software, equipment and data resources;
• permit dynamic re-expression by allowing 'multiple world views' of the same spatial environment;
• maximise flexibility by allowing elements to be added and removed as necessary;
• allow data and software components to exist on distributed platforms.

The 'panoraMap' was designed to fulfil each of these objectives in order to form part of the VFC software component suite as well as additional specific goals as outlined in chapter 6. This appendix has provided a more substantial impression of the aims and objectives of the rationale behind the software and beds it in a well specified set of more general requirements proving that the approach taken here and implementation developed have been of merit and real application.
Appendix A6:
Scripts that Specify Focus and Orientation

Jason Dykes
Department of Geography
University of Leicester
UK

Interactive Maps for Exploratory Spatial Data Analysis:
Cartographic Visualization Approach, Implementation and Application
# -- Jason Dykes, University of Leicester, 1995-1999 - jad7@le.ac.uk
# -- Tcl Script 'focusProcs.tcl' - Contains 8 procedure definitions:
# --- makeSymbol 'changeFocus' 'doFocus' 'meanVal' 'makeCircle' 'makeRect' 'makeImg' 'getOpt'

# -- Produces items using 'focus' visual variable on Tcl/Tk canvas.
# -- Run with command with one of the following forms:
# -- makeSymbol .c $x $y -type circle -focus 0 -radius 10 -fill Blue -tags "circle"
# -- makeSymbol .c $x $y -type rect -focus 1 -width 10 -height 12 -fill Red
# -- makeSymbol .c $x $y -type image -focus 2 -image aPhoto -tags "image aPhoto"

# -- Procedure Definition 'makeSymbol'.
# -- Procedure to create symbol with 'focus' visual variable specified.
# -- Creates symbol on canvas as specified by list of arguments.
# -- Takes a Single Argument - 'args': List with following format.
# -- Required arguments:
# -- 'c': 1st argument - Name of Tcl/Tk canvas widget.
# -- 'x': 2nd argument - x location of symbol on canvas.
# -- 'y': 3rd argument - y location of symbol on canvas.
# -- General option and value pair arguments:
# -- '-type': Item type - valid values 'circle' 'rect' 'image'.
# -- '-fill': Colour of symbol.
# -- '-focus': Focus factor (kernel size is 2f+1).
# -- '-tags': List of any tags to be given to item.
# -- Specific option and value pair arguments:
# -- '-radius': Determines radius of circle symbols.
# -- '-height': Determines height of rectangle symbols.
# -- '-width': Determines width of rectangle symbols.
# -- '-image': Specifies name of image to be used for image symbols.

proc makeSymbol {args} {
# -- Strip values for required arguments from argument list:
set c [lindex $args 0]
set x [lindex $args 1]
set y [lindex $args 2]
# -- Strip values for general option arguments from argument list with 'getOpt':
set k [getOpt $args -fill]
set f [getOpt $args -focus]
set tags [getOpt $args -tags]
# -- Check symbol type - 'getOpt' strips value of '-type' option from arg list.
# -- Get appropriate options for symbol from argument list.
# -- Run appropriate 'make' procedure to create requested symbol:
switch [getOpt $args -type] {
circle {
    set r [getOpt $args -radius]
    makeCircle $c $x $y $r $k $tags
}
rect {
    set zH [getOpt $args -height]
    set zW [getOpt $args -width]
    makeRect $c $x $y $zW $zH $k $tags
}
image {
    set i [getOpt $args -image]
    makeImg $c $x $y $i $tags
}
# -- Call 'doFocus' procedure to change focus:
doFocus $iN $f

# -- reconfigure canvas item to contain newly focused image:
$c itemconfigure $iN -image $iN-$f

# -- Procedure Definition 'doFocus'.
# -- Performs mean filter to blur image object with specified kernel size.
# -- Takes 2 Arguments...
# -- 'iN': Tcl/Tk image name
# -- 'f': Focus factor (kernel size is 2f+1)
proc doFocus {iN f} {
    global SF

    # -- Determine image height and width:
    set iH [image height $iN]
    set iW [image width $iN]

    # -- Determine height and width of new image containing filtered information:
    set cW [expr $iW + ($f*2)]
    set cH [expr $iH + ($f*2)]

    # -- Create new image containing filtered information:
    image create photo $iN-$f -width $cW -height $cH

    # -- Loop through pixels of new image:
    for {set r 0} {$r <$cH} {incr r} {
        for {set c 0} {$c <$cW} {incr c} {
            # -- Identify starting row/column for filter kernel:
            set rp [expr $r-$f]
            set cp [expr $c-$f]

            # -- Perform procedure to calculate mean value of kernel:
            set mV [meanVal $rp $cp $f $iN]

            # -- Calculate hexadecimal RGB value from returned list of integers:
            set rH [format %.2x [lindex $mV 0]]
            set gH [format %.2x [lindex $mV 1]]
            set bH [format %.2x [lindex $mV 2]]
            set rgb # {rH}$ {gH}$ {bH}

            # -- Put RGB value into new image at current row/column location:
            $iN-$f put $rgb -to $c $r $c+l $r+l
        }
    }
}

# -- Procedure Definition 'meanVal'.
# -- Calculates and returns list of integers containing mean values of
# -- red, green and blue (from 0-255) at a location in an image.
# -- Takes 4 Arguments...
# -- 'r': Row number of point where mean to be calculated.
# -- 'c': Column number of point where mean to be calculated.
# -- 'f': Focus factor (kernel size is 2f+1)
# -- 'iN': Tcl/Tk image name
proc meanVal {r c f iN} {
    global SF

    # -- Initialise counts for number of cells, and colour components:
    set n 0
    set R 0
    set G 0
    set B 0

    # -- Create structure to loop through rows and columns of image in kernel.
    for {set r1 -$f} {$r1 <=$f} {incr r1} {
        for {set c1 -$f} {$c1 <=$f} {incr c1} {
            # -- Identify location of cell in image to be included in kernel:
            set rN [expr $r+$r1]
            set cN [expr $c+$c1]

            # -- Get values from image if image contains data at location.
            # -- If not, set returned value to "255 255 255";
        }
    }

    # -- Calculate mean value of kernel:
    set mV [expr $R/$n $G/$n $B/$n]

    # -- Return list of integers containing mean values at location:
    return [list $rN $cN [lindex $mV 0] [lindex $mV 1] [lindex $mV 2]]
}
switch [catch {set z [get $c $r $n]}] {0 { } default { set z "255 255 255" }
}

# -- Strip out RGB values from string.
# -- Increment appropriate variable by value returned at location:
incr R [lindex $z 0]
incr G [lindex $z 1]
incr B [lindex $z 2]

# -- Increment cell counter:
incr n
}

# -- Calculate mean value of each colour component:
set rA [expr round($R/double($n))]
set gA [expr round($G/double($n))]
set bA [expr round($B/double($n))]

# -- Return list containing mean values for red, green and blue:
return "$rA $gA $bA"
}

# — Procedure Definition 'makeCircle'.
# — Creates and plots image object depicting circle symbol.
# -- Takes 6 Arguments...
# — 'canvas': Name of Tcl/Tk canvas widget.
# — 'x': x location at which to center circle on canvas.
# — 'y': y location at which to center circle on canvas.
# — 'z': Circle width in pixels.
# — 'k': Colour of circle in hexadecimal.
# — 'tags': Any tags to add to circle item.
proc makeCircle {canvas x y z k tags} {

# -- Specify name for image, using location as unique identifier:
set iN c-($x,$y)

# — Create new image to contain symbol:
image create photo $iN -width $z -height $z

# -- Calculate radius of circle:
set z2 [expr $z*0.5]

# -- Loop through all locations in image pixel by pixel:
for {set r 0} {$r <$z} {incr r) {
    for {set c 0} {$c <$z} {incr c} {

        # — Calculate distance between current cell and centre of image:
        set dx [expr $z2-$r]
        set dy [expr $z2-$c]
        set hyp [expr hypot($dx,$dy)]

        # — Check if distance is less than the radius.
        # — If so, fill the location with the specified colour:
        if {$hyp < $z2}  {
            $iN put $k -to $r $c [expr $r+l] [expr $c+l]
        }  else {
            $iN put tffffff -to $r $c [expr $r+l] [expr $c+l]
        }
    }
}

# -- Create an item on the canvas containing the image, with the tags provided:
$canvas create image $x $y -image $iN -anchor c -tags "image $iN $tags"
}

# — Procedure Definition 'makeRect'.
# — Creates and plots image object depicting rectangle symbol.
# — Takes 7 Arguments...
# — 'canvas': Name of Tcl/Tk canvas widget.
# — 'x': x location at which to rectangle circle on canvas.
# — 'y': y location at which to rectangle circle on canvas.
# -- 'zW': Rectangle width in pixels.
# -- 'zH': Rectangle height in pixels.
# -- 'k': Colour of rectangle in hexadecimal.
# -- 'tags': Any tags to add to rectangle item.
proc makeRect {canvas x y zW zH k tags} {
    # -- Specify name for image, using location as unique identifier:
    set iN c-($x,$y)
    # -- Create new image to contain symbol:
    image create photo $iN -width $zW -height $zH
    # -- Fill specified area of image with selected colour:
    $iN put $k -to 0 0 $zW $zH
    # -- Create an item on the canvas containing the image, with the tags provided:
    $canvas create image $x $y -image $iN -anchor c -tags "image $iN $tags"
}

# — Procedure Definition 'makelmg'.
# -- Creates and plots image object containing specified image.
# — Takes 5 Arguments...
# -- 'canvas': Name of Tcl/Tk canvas widget.
# -- 'x': x location at which to rectangle circle on canvas.
# -- 'y': y location at which to rectangle circle on canvas.
# -- 'i': Name of Tcl/Tk image object.
# -- 'tags': Any tags to add to rectangle item.
proc makelmg {canvas x y i tags} {
    # -- Specify name for image, using location as unique identifier:
    set iN c-($x,$y)
    # -- Create new image to contain symbol:
    image create photo $iN
    # -- Copy data from specified image into new image:
    $iN copy $i
    # -- Create an item on the canvas containing the image, with the tags provided:
    $canvas create image $x $y -image $iN -anchor c -tags "image $iN $tags"
}

# — Procedure Definition 'getOpt'.
# — Checks a list for a specified string and returns following element.
# — Takes 2 Arguments...
# -- 'list': List of strings.
# -- 'string': String that identifies value to be returned as next element.
proc getOpt {list string} {
    # -- Find location of string in list:
    set i [lsearch "$list" "$string"]
    # -- If list does not contain string return an empty string:
    if {($i <0)} return
    # -- Return following element of list:
    return [lindex $list [expr $i +1]]
}
produces polygon items using 'orientation' visual variable on Tcl/Tk canvas.
-- Procedure to create symbol with 'orientation' and 'size' visual variable specified.
-- Takes a Single Argument - 'args': List with following format.
-- Required arguments:
  -- 'c': 1st argument - Name of Tcl/Tk canvas widget.
  -- 'x': 2nd argument - x location of symbol on canvas.
  -- 'y': 3rd argument - y location of symbol on canvas.
-- Command must define polygon with one of following pair of arguments:
  -- '-coords': Followed by coordinate list defining polygon shape in canvas units.
  -- '-poly': Followed by unique identifier specifying pre-defined polygon to use.
-- General option and value pair arguments:
  -- '-scale': Scaling factor relative to original to determine polygon size.
  -- '-fill': Colour with which to fill symbol.
  -- '-tags': List of any tags to be given to item.
proc makePoly {args} {
  # -- Strip values for required arguments from argument list:
  set c [lindex $args 0]
  set x [lindex $args 1]
  set y [lindex $args 2]
  # -- Strip values for general option arguments from argument list with 'getOpt':
  set z [getOpt $args -coords]
  set p [getOpt $args -poly]
  set o [getOpt $args -outline]
  set s [getOpt $args -scale]
  set k [getOpt $args -fill]
  set r [getOpt $args -orient]
  set tags [getOpt $args -tags]
  # -- If option values do not exist for fill, outline and tags, use defaults:
  if { [llength $k] < 1} { set k {{}} }
  if { [llength $o] < 1} { set o Black }
  if { [llength $tags] < 1} { set tags poly }
  # -- Return if no coordinate list exists:
  if { [llength $z] < 1} {
    return 1
  }
  # -- Use standard Tcl/Tk canvas command to create polygon on canvas.
  # -- Polygon has fill, outline and tags as specified by argument list:
  set id [eval $c create poly $z -fill $k -outline $o -tags $tags]
  # -- Find bounding box of polygon:
  set bbox [lindex $c $bbox $id]
  # -- Add specified tags and 'poly' tag to item:
  set $k withtag $id
  set $k addtag poly withtag $id
  # -- Calculate centroid by finding centre of bounding box:
  set x2 [expr ([lindex $bbox 0] + [lindex $bbox 2]) * -0.5]
  set y2 [expr ([lindex $bbox 1] + [lindex $bbox 3]) * -0.5]
  # -- Position polygon item at origin (0,0):
  set $id move $x $y
  # -- Scale polygon item by specified scaling factor:
  set $id scale 0 0 $s $s
  # -- Re-position polygon item at centroid (x,y):
  set $id move $x $y
}
# -- Use 'rotateItem' procedure to perform final rotation transformation:
orientItem $c $id $r
)

# -- Procedure Definition 'orientItem'.
# -- Rotates any polygon item on a canvas with a unique identifying tag with 'orientAbout'.
# -- Initial function of procedure is to calculate centroid about which item is rotated.
# -- Takes 3 Arguments...
# -- 'c': Name of Tcl/Tk canvas.
# -- 'id': Unique identifier of item to be rotated.
# -- 'd': Angle, in degrees, through which rotation is to be applied.
proc orientItem {c id d} {
# -- Check for existence of element representing centroid of item in 'dig':
if ![info exists dig($id,cX)] {
    # -- If the element does not exist, calculate centroid.
    # -- Find bounding box of item:
    set bbox [$c bbox $id]
    # -- Calculate and store centre of bounding box:
    set cX [expr ([lindex $bbox 0] + [lindex $bbox 2]) *0.5]
    set cY [expr ([lindex $bbox 1] + [lindex $bbox 3]) *0.5]
}
# -- Call 'orientAbout' procedure with centroid 'x' and 'y' as arguments:
orientAbout $c $id $cX $cY $d
}

# -- Procedure Definition 'orientAbout'.
# -- Rotates any polygon item on canvas with unique identifying tag about specified point.
# -- Takes 5 Arguments...
# -- 'c': Name of Tcl/Tk canvas.
# -- 'id': Unique identifier of item to be rotated.
# -- 'cX': x coordinate of point about which item is to be rotated.
# -- 'cY': y coordinate of point about which item is to be rotated.
# -- 'd': Angle, in degrees, through which rotation is to be applied.
proc orientAbout {c id cX cY d} {
# -- Store initial coordinates of item:
set coords [$c coords $id]
# -- Initialise variable to contain list of rotated coordinates:
set rCoords ""
# -- Loop through list of coordinate pairs in coordinate list:
foreach "x y" $coords {
    # -- Call 'rotateCoords' procedure to return location of coords following rotation:
    set nCoords [rotateCoords $cX $cY $x $y $d]
    # -- Strip 'x' and 'y' from returned list:
    set xN [lindex $nCoords 0]
    set yN [lindex $nCoords 1]
    # -- Add 'x' and 'y' to list of rotated coordinates
    set rCoords "$rCoords $xN $yN"
}
# -- Update coordinates of item to new list of rotated coordinates:
eval $c coords $id $rCoords
}

# -- Procedure Definition 'rotateCoords'.
# -- Rotates a single pair of coordinates about a location through a specified angle.
# -- Takes 5 Arguments...
# -- 'cx': x coordinate of point about which item is to be rotated.
# -- 'cy': y coordinate of point about which item is to be rotated.
# -- 'x': x coordinate of point to be rotated.
# -- 'y': y coordinate of point to be rotated.
# -- 'r': Angle through which rotation is to take place, in degrees clockwise.
proc rotateCoords {cx cy x y r} {
# -- Calculate distance between coordinates:
set dx [expr ($x-$cx)]
set dy [expr ($y-$cy)]

set d [expr hypot($dX,$dY)]

# -- If this distance is zero, return the original coordinates:
if {$d == 0} {  return "$x $y" }

# -- Use the magnitude of the variations in x and y to determine the quadrant of the
# -- circle in which the point to be rotated exists, in relation to the point about
# -- which the rotation is to take place. Store this in variable 'q':
if {$dX < 0} {
  if {$dY > 0} {  set q 1 }  else {  set q 4 }
}  else {
  if {$dY > 0} {  set q 2 }  else {  set q 3 }
}

# -- Use 'radtodeg' procedure to find angle from rotation point to rotated point:
set rO [radtodeg [expr asin($dY)]]

# -- This will depend upon the quadrant, update the angle 'rO' appropriately:
switch $q {
  1  {  set rO $rO }  
  2 {  set rO [expr 180-$rO] }  
  3 {  set rO [expr 180-$rO] }  
  4 {  set rO [expr 360+$rO] }  
}

# -- Calculate the angle between the rotation point and the rotated point following
# -- the rotation by adding the original and rotation angles:
set rN [expr $rO + $r]

# -- Find the sine and cosine of this new angle using 'degtorad' to convert:
set rC [degtorad $rN cos]
set rS [degtorad $rN sin]

# -- Calculate the new coordinates by multiplying the distance between the points by
# -- the appropriate function and adding this to the rotation point:
set xN [expr $cX + ($d * $rC)]
set yN [expr $cY + ($d * $rS)]

# -- Reformet the returned values to a precision of two decimal places:
set xR [format %.2f $xN]
set yR [format %.2f $yN]

# -- Return a list containing the rotated coordinate pair:
return "$xR $yR"

# -- Procedures to convert between degrees (as specified by the user) and radians
# -- as required and returned by the Tcl/Tk math functions.

# -- Set value of pi as global variable:
set pi 3.14159265

# -- Procedure Definition 'degtorad'.
# -- Converts angles specified in degrees to radians.
# -- Returns angle, sine of angle, or cosine of angle depending upon arguments.
# -- Takes 2 Arguments...
# -- 'n':  Angle in degrees.
# -- 'k':  Determines option that is used in returning value.
proc degtorad {n k} {

  # -- Specify global variables to be used in procedure:
  global  pi

  # -- Convert into radians:
  set v [expr ($pi*($n/180.0))]

  # -- Depending upon argument return converted angle, its sine or cosine:
  switch $k {
    rad { return $v }  
    sin { set s [format %.6f [expr sin($v)]]  
           return $s }  
    cos { set c [format %.6f [expr cos($v)]]  
           return $c }  
  }
}
# -- Defaults to returning converted angle:
return $v
}

# -- Procedure Definition 'radtodeg'.
# -- Converts angle in radians into degrees.
# -- Takes a Single Argument - 'n': Angle to be converted.
proc radtodeg {n} {
  # -- Specify global variables to be used in procedure:
  global pi
  # -- Convert into degrees:
  set v [expr (180*($n/$pi))]
  # -- Return converted value:
  return [format %.4f $v]
}

# -- Procedure Definition 'getOpt'.
# -- Checks a list for a specified string and returns following element.
# -- Useful for stripping values from options in argument list.
# -- Takes 2 Arguments...
# -- 'list': List of strings.
# -- 'string': String that identifies value to be returned as next element.
proc getOpt {list string} {
  # -- Find location of string in list:
  set i [lsearch "$list" "$string"]
  # -- If list does not contain string return an empty string:
  if {$i <0} return
  # -- Return following element of list:
  return [lindex $list [expr $i +1]]
}
Appendix B: Software Appendices

B1 - Tk Examples
B2 - Interactive Figures: Map Examples 1-6
B3 - Visual Variables in Tcl/Tk
B4 - 'cdv' Software
B5 - Interactive Figures: Map Examples 7-9
B6 - 'simplePanZoom' Example Software
B7 - 'panoraMap' Application Software
B8 - Software to Visualize Tcl/Tk Speed Test Results
B9 - 'focus' Example Software
B10 - 'orientation' Example Software

Jason Dykes
Department of Geography
University of Leicester
UK
Ten software appendices are provided on the accompanying compact disk.

The software should be used in conjunction with the text of the thesis.

The software is suitable for use under Microsoft Windows™ 32-bit operating systems including Windows 95, Windows 98, Windows 2000 and Windows NT.

All software appendices are located in a directory named 'Appendix-B'. The individual programs are located in a sub-directory structure that identifies the appendix by number. So for example, the software that forms appendix B1 is located in:

'/Appendix-B/1/

Each directory thus represents a single appendix.

Instructions on running the software in Appendix B are provided in a hypertext file in each directory entitled 'index.html'. This will usually involve double clicking an icon representing the Windows program executable. Some browsers allow this to occur within the browser. If this is not the case, the file should be located using 'Windows Explorer' and the appropriate icon double clicked. Instructions on using the software are provided through a 'Help' or 'Info' menu item within the program unless otherwise specified.

The performance of the software is dependent upon the specification of the machine upon which it runs, and obviously, the higher performance machine, the better. A recommended minimum set-up would involve a 166MHz Pentium processor with 32MB of RAM. The performance of the software may be improved by copying the software to a hard disk. It is essential that the files and directory structure provided on the CD are replicated on the hard drive exactly.

Attention is drawn to the copyright and disclaimer statements made in the software and the copyright notices provided in the acknowledgments section of this thesis.
Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization

Approach, Implementation and Application

Appendix C: Support Materials

C1 - Dynamic and High Resolution Figures
C2 - Guide to Using 'cdv'
C3 - 'cdv' Scripts: Documented Source Code

Jason Dykes

Department of Geography
University of Leicester
UK
C. Support Materials

Three appendices containing digital documents are provided in Appendix C on the accompanying compact disk.

The information should be used in conjunction with the text of the thesis.

It is provided in two ASCII formats: HyperText Mark-up Language and Tcl/Tk Script. Files of the former format are identified by the 'HTML' extension and should be viewed with an appropriate WWW browser. Files of the latter format are identified by the '.TCL' or '.TXT' extension and should be viewed with a text editor or a suitable software control system.

All resources in Appendix C are located in a directory named 'Appendix-C'. The individual appendices are located in a sub-directory structure that identifies the appendix by number. So for example, the documents that form appendix C1 are located in:

'/Appendix-C/1'

Each directory thus represents a single appendix.

The file 'index.html', located in each directory, will either outline the use and contents of the appendix or provide a series of hyperlinks to the materials that are available.

Attention is drawn to the copyright notices provided in the acknowledgments section of this thesis.
Appendix D: Academic Papers

D2 – Dykes (1997) Computers & Geosciences

Jason Dykes
Department of Geography
University of Leicester
UK
D. Academic Papers

An series of paper reprints is provided in appendix D in the rear cover pocket.

The papers are contributed to support the stated objective of a thesis submitted for admission to the degree of Doctor of Philosophy that a thesis "contains material which has proved to be worthy of publication" (University of Leicester, 1998).

Each represents a formal published contribution based upon the work presented here. Significant additional details, background and context are provided.

The papers span the period of the research and include:

- an initial assessment of the developing field of visualization and its likely effects on cartography and specifically map design
- an interactive introduction to the methodology utilised here and the application of these techniques to address research issues in visualization
- a formal description of the 'cdv' software
- a report on the Virtual Field Course software and architecture, including the aims and objectives of the project and a description of the 'panoraMap' software.

They are as follows:


Interactive Maps for Exploratory Spatial Data Analysis:

Cartographic Visualization
Approach, Implementation and Application

10. Bibliography: Cited References

Jason Dykes
Department of Geography
University of Leicester
UK
10. Bibliography: Cited References


Apple Computer, 1999, Quicktime VR:
www.apple.com/quicktime/qtvr/


Ball, S., 1997, Td meets the World Wide Web:
tdk.anu.edu.au

Ball, S., 1998, TdXML:
www.zveno.com/zm.cgi/in-tdxml


Carter, J. and Dykes, J.A., 1998, Developing a Visualization Gateway for the UK Census of
Population,, Proceedings International Symposium on Spatial Data Handling, Vancouver, Canada:, 543-555.


Casio, 1998, Casio Online:

www.casio.com/digitalimaging/


Cleveland, W.S., 1998, An Approach to Model Building Based on a Bayesian Theory of Data Exploration:


Couch, J., 1998, Proposal for a VRML 97 spec addition: External authoring interface:

www.vrml.org/WorkingGroups/vrml-eait/proposals/justin/proposal.html

Critchlow, R.E., 1995, expr Manual Page:

www.scriptics.com/man/tcl8.1/TclCmd/expr.htm


Dublin Core, 1997, Dublin Core Metadata: purl.org/metadata/dublin_core

Duckham, M., 1996, pers. comm., Program submitted as part of GY708 Assessment, University of Leicester, UK.


Dykes, J.A. and Wood, J.D., 1993, SITE: Developing an Integrated Environment for Spatial Interactive Maps for Exploratory Spatial Data Analysis


Dykes, J.A., 1996, Td/Tk for Mapping: Main Menu:

www.mimas.ac.uk/argus/Tutorials/CartoViz/examples/


Dykes, J.A., 1998, cdv Guide:

www.geog.le.ac.uk/cdv/Guide

Dykes, J.A., 1998, cdv Home Page:


Dykes, J.A., 1999, GUIs for the Geographer: Some Experiences with Interactive Geo-Graphics:

Dykes, J.A., 1999, Internet Home Page:


Emerson, M. and Revere, D., 1997, Virtual Reality: Education and training applications:


Esperanca, C., 1998, A Tk Open GL Widget:
www.esri.com/software/arcview/


Friends of the Earth, 1996, Chemical release inventory map:
www.foe.co.uk/cri/html/postcode.html

Friends of the Earth, 1999, Factory Watch:
www.foe.co.uk/factorywatch/


George Mason University, 1999, Project ScienceSpace:
www.virtual.gmu.edu/

GeoVRML Working Group, 1998, GeoVRML Working Group:
www.ai.sri.com/geovrml/


Harris, J., 1999, CasWeb - web based interface to census area statistics:  
  [census.ac.uk/casweb/](http://census.ac.uk/casweb/)


Herzog, A., 1998, Dorling Cartogram:  
  [www.zh.ch/statistik/map/dorling/dorling.html](http://www.zh.ch/statistik/map/dorling/dorling.html)

Higher Education Funding Council of England (HEFCE), 1995, Quality Assessment of Geography:  
  [www.niss.ac.uk/education/hefce/pub95/qq_11_95.html](http://www.niss.ac.uk/education/hefce/pub95/qq_11_95.html)


Hopkirk, M.J., 1997, Example Code: Socket server and client:  


JavaSoft, 1998, Java Beans Home Page:  
  [www.javasoft.com/beans](http://www.javasoft.com/beans)
Jenks, G.F., Optimal data classification for choropleth maps, *Occasional Paper 2*, Department of Geography, University of Kansas.


MacEachren, A.M., 1994, Some truth with maps: A primer on symbolization and design, AAG: Washington D.C.


Manchester Metropolitan University, 1996, Project Argus Homepage: www.mimas.ac.uk/argus/


Microsoft Inc., 1998, What is Microsoft Repository?:

www.eu.microsoft.com/repository/techmat/whitepapers/whatismr.htm

Microsoft Inc., 1998, Microsoft TerraServer:

terraserver.microsoft.com


Muehrcke, P.C., 1978, Map use: Reading, analysis and interpretation, J.P. Publications: Madison,


Muller, J.-C., 1976, Number of classes and choropleth pattern characteristics, *The American Cartographer*, 3(2), 169-175.


Oden, N.L., 1984, Assessing the significance of a spatial correlogram, *Geographical Analysis*, 16(1), 1-16.


Pascoe, J. and Ryan, N., 1998, Mobile Computing in a Fieldwork Environment:
www.cs.uk.ac.uk/research/infosys/mobicomp/Fieldwork/index.html

Paul, B. and Bederson, B., 1996, Togl - a Tk OpenGL Widget:
www.cs.umd.edu/~bederson/Togl.html

Pennsylvania State University, 1998, The Apoala Project:
www.geovista.psu.edu/apoala/index.htm

Pennsylvania State University, 1998, GeoVISTA Center:
www.geovista.psu.edu


PictureWorks Technology Inc., 1999, Spin Panorama:
www.videobrush.com/spinpano/


Reddy, M., 1998, The QuadLOD Node for VRML:
www.ai.sri.com/~reddy/geovrml/new_lod/


Ritter, M., 1998, Virtual Fieldtrips:
wwwhost.cc.utexas.edu/~cor/pub/grg/virtdept/contents.html

Robertson, P.K., 1988, Choosing data representations for the effective visualisation of spatial data, Proceedings Third International Symposium on Spatial Data Handling, 243-252.


Salvemini, M., 1991, Considerazioni territoriali sulla legge 142/90 sull'ordinamento delle autonomie locali ed elaborazioni tramite sistema informativo territoriale di dati statistici ad essa relativi, Dipartimento di Caratteri degli Edifici e dell'Ambiente, Universita di Roma, "La Sapienza".


Scriptics Corporation, 1998, Scriptics: The Td platform company:

www.scriptics.com

Scriptics Corporation, 1998, Td Web Server:

www.scriptics.com/tdhttpd/

Scriptics Corporation, 1998, TdPro:

www.scriptics.com/tdpro/

Scriptics Corporation, 1999, Td Java Integration:

www.scriptics.com/products/java/

Scriptics Corporation, 1999, Td Plugin:

www.scriptics.com/tdplugin

Scriptics Corporation, 1999, TdPro:

www.scriptics.com/products/tdpro/

Scriptics Corporation, 1999, Safe-Td:

www.scriptics.com/plugin/safetd.html


www.wstonline.com/mfwtparse.html?page=showarticle&aid=5801


Southall, H. and White, B., 1998, Electronic atlas of industrialising Britain: www.geog.qmw.ac.uk/aib/


Stynes, K., 1996, Data for ESDA with LISA: www.mimas.ac.uk/argus/esdalisa/midwest/Data.html


The Manchester Metropolitan University, 1999, The Janus Project: www.kinds.ac.uk/kinds/janus.htm

The Virtual Field Course, 1998, UK Higher Education Field Course Locations: www.geog.le.ac.uk/vfc/education/fieldwork.html
The Virtual Field Course, 1998, About the Virtual Field Course:
  www.geog.le.ac.uk/vfc/about/index.html
The Virtual Field Course, 1999, Welcome to the Virtual Field Course:
  www.geog.le.ac.uk/vfc/
The VRML Consortium, 1997, The Virtual Reality Modelling Language:
  www.vrml.org/technicalinfo/specifications/vrml97/
The VRML Consortium, 1998, The Virtual Reality Modelling Language:
  www.vrml.org
Theus, M., 1996, Theorie und anwendung interaktiver statistischer graphik, Ph.D. Thesis,
  Augsburger Mathematisch-Naturwissenschaftliche Schriften, 140pp.
  Geography, 46, 234-240.
Tobler, W.R., 1979, A transformational view of cartography, The American Cartographer, 6, 101-
   106.
Tomlin, C.D., 1990, Geographic information systems and cartographic modelling, Prentice
  Hall:Englewood Cliffs, NJ.
Troyer, M., 1999, The World of H.C. Berann:
  www.imDetus.ne.ip/berann/
  Series 2, Princeton University.
University of Leicester, 1998, Postgraduate Regulations 1998-99, University of Leicester:Leicester,
   204pp.
Unwin, A.R., 1996, Exploratory spatial analysis and local statistics, Computational Statistics, 11,
   387-400.
Unwin, A.R., 1998, Information visualisation for large data sets:
www.research.att.com/~andreas/abstract-unwin.html


US Census Bureau, 1997, 1990 Census Lookup: 

venus.census.gov/cdrom/lookup/


Videobrush Corporation, 1998, Panorama V2.0: 

www.videobrush.com/videobrush/products/panorama/

VistaPro, 1999, VistaPro Web Site: 

/www.egames.com/egames/other_sites/vistapro/default.htm

Ware, C. and Beaty, J.C., 1988, Using color dimensions to display data dimensions, *Human factors*, 30, 127-142.

Wasabi Software, 1999, SkyPaint: 3D Panorama Paint Tool:
www.wasabisoft.com/SkyPaint/

Web Publishing Inc., 1999, XML for the absolute beginner - A guided tour from HTML to processing XML with Java:


Wilhelm, A., 1998, Interactive Statistical Graphics: A theoretical approach based on the paradigm of linked views:
www.research.att.com/~andreas/abstract-wilhelm.html


Wills, G.J., 1999, *pers. comm.*, Statement made during visit to Lucent Technologies, Naperville, IL.

Wills, G.J., 1998, EDV: The Exploratory Data Visualizer:
www.bell-labs.com/user/gwills/EDVguide/edv.html


Documents referenced with universal resource locators were available in the public domain, at the specified address at the time of publication. The author cannot be responsible for any change in the location or content of these documents.

Jason Dykes

30th August 1999
Scientific Visualization is a growing area of computing with the underlying philosophy that displaying visual representations of data assists humans in generating ideas and hypotheses about the data. The emphasis is on the fostering of ideas as opposed to presentation of information, as in conventional cartography. Having said that, it would appear that there is much opportunity for cartographers and geographers to benefit from the computational possibilities being opened by visualization researchers, and similar for the visualization software developers and users to learn best-practice in display methods from cartographers.

Map design and visualization

P. Fisher, J. Dykes and J. Wood

Department of Geography, University of Leicester, Leicester LE1 7RH

INTRODUCTION

In 1987 a report submitted to the National Science Foundation in the US, and re-published in the journal of the Association of Computing Machinery Graphics Special Interest Group (McCormick et al., 1987), took the computer graphics community by storm, and introduced a wide readership to the concept of SCIENTIFIC VISUALIZATION. The ideas expressed there are slowly permeating the geographical and cartographic communities, particularly among those practitioners engaged in computer-based work.

According to McCormick et al. (1987), "Visualization is a method of computing. It transforms the symbolic into the geometric, enabling researchers to observe their simulations and computations" (p. 3). They go on to state that it is "a tool for both interpreting image data fed into a computer, and for generating images for complex multidimensional data sets" (ibid.). It is clear that an uncritical reading of this could embrace all or none of cartography. As Buttenfield and Mackaness (1991), MacEachren with others (1992), and MacEachren and Monmonier (1992) point out, however, neither of these are the case (see also MacEachren and Fraser Taylor, in press; Hearnshaw and Unwin, in press).

The objective of scientific visualization, as it interacts with cartography, is the development of ideas and hypotheses about spatial information, and is embodied in the computational and non-computational tool kit which assists a scientist in the exploration of their data so that they can develop ideas (ideation) which lead to the better understanding of the information. Those ideas may later form the basis for formal hypotheses which can be tested by either standard or novel statistical methods, or may remain as ideas to be further investigated. Thus in the context of statistics, the methods of scientific visualization clearly build on the ideas of Exploratory Data Analysis as laid down by Tukey (1977), but with the advanced graphics capability of the modern computer workstations. Scientific visualization is therefore at the heart of a response to appeals for exploratory spatial data analysis in GIS (Fotheringham, 1992); an appeal which is being addressed by current research reviewed below (e.g. Haslett et al., 1990; Egbert and Slocum, 1992; MacDougall, 1992).

Within a computer environment the central theme of visualization is that a scientist can quickly create a series of images of the same and/or related information, which can be displayed and re-displayed in any order and with control of time variables. Individual images may vary static or dynamic variables. Interactive capabilities mean that further variation may be provided through re-expression where images respond in real-time to transformations of the data. These techniques build up diverse impressions of the data and facilitate the user's understanding. In most definitions visualization is not dependent upon interaction, however, and presentation graphics and static views of data can both do much to foster ideas in the viewer. The emphasis in scientific visualization is on the development of ideas, not, as in traditional cartography, the presentation of an idea or view.

To assist the distinction between this and other papers in this special issue, discussion here will focus on methods of interaction with data, and of dynamic display, at the expense of static visualization although even here illustrations are necessarily static images. The importance of static views is fully acknowledged by the authors, and reference is made to these elsewhere in this issue.

The paper is divided into three sections. The next section presents some simple examples of the power of visualization, and then the technological facilitators of scientific visualization are outlined. Finally, some of the developing cartographic techniques made available to cartographers through dynamic visualization are reviewed.

SIMPLE EXAMPLES

One of the simplest examples of the power of visualization techniques is to compare Equation 1 and Figure 1. Which
is more recognisable as the normal distribution? To the
majority undoubtedly the latter serves the purpose better,
although the former is more precise and so may be more
acceptable to some. How many readers can identify the
first in the second, and how many can 'picture' the dis­
tribution and its implications without the figure? Many of
us can handle the idea of the normal distribution, and
associate it with a curve of a particular form, but few of us
are as comfortable with the equation.

\[ f(X) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(X-\mu)^2}{2\sigma^2}} \]  

Equation 1

In spatial analysis too, visualization is very important in
understanding, and particularly in teaching (the fostering of
ideas). One of the most complex spatio-statistical concepts for
students to understand is spatial autocorrelation. An interac­
tive package (with visualization) can transmit this concept
quickly and effectively (Figure 3). Similarly, the point patterns
algorithms at any time during processing by plotting the
order of items at any stage in the processing against the
order they should be in. Indeed, a similar visualisation is
included as a demonstration program with Microsoft's
Visual Basic for the PC.

A further example of the power of visual analysis applied
in research can be found in the study and validation of com­
puter algorithms for generating pseudo-random numbers.
Sharp and Bays (1992), among others, have shown that
simple scatter plots of pseudo-random number sequences
can show very pronounced and disturbing properties
(Figures 4), which are a reflection of non-randomness in the
data. The beauty of this is that the time to execute the tests is
trivial, no understanding of statistical methods is needed
(other than the intended properties of the random number
sequences), and the result is apparent to almost anyone. By
contrast, and published within months, Jennings and
Mohan (1991) present a program which contains twenty
statistical tests designed to explore the acceptability of a
generator. The user needs a very high level of statistical
sophistication to use any of these methods, and any result
requires interpretation of test statistics. The quality of the
generator is therefore not so readily apparent in the statisti­
ical analysis as in the visualization method which employs
only 2-D and 3-D scatterplots, one of the simplest of
graphics.

which may yield a particular value of the nearest neighbour
statistic are very varied, and a student can explore this in a
suitably versatile package (Figure 2). In similar vein, Baecker
and Small (1990; Baecker, 1979) present visualizations which
assist understanding of how sorting algorithms work. They
allow the viewer to visualize the success of various

Figure 1. The normal distribution is more easily understood from the
curve than from the formula (Equation 1).

Figure 2. Spatial statistics for point patterns are relatively complex
and the nearest neighbour statistic in particular can be very hard to
grasp. Here a user can add or remote points, and outlines and calcula­
tate and re-calculate various point pattern statistics. The figure is a
screen dump from a program written by PF.

Figure 3. Spatial autocorrelation is notoriously difficult to under­
stand. This program displays a raster image with a specified value for
Moran's I. The user can set any 'Target' value of 1 between 1 and -1
by moving the lower arrow, and watch the display evolve towards that
autocorrelation shown by the 'Current' arrow. In another option they
use the same basic interface to guess the value of 1 in a game. The
figure is a screen dump from a program written by PF.

Figure 4. Scatterplots of number pairs derived from a pseudo random
number generator can reveal very disturbing patterns quickly and
easily. The apparently random pattern itself shows regularity when
plotted as triples in the 3-D scatterplot (redrawn from Sharp and Bays,
TECHNOLOGICAL PREREQUISITES

A computing environment is necessary for interaction with the data which is a fundamental part of the visualization process. This has only recently become widely available with the massive advances in personal computing power. Indeed, the potential for visual display now possible in the relatively modest PC compatibles and Macintoshes is enormous, and increasingly used. Such machines, relatively low cost as they are, can be used to excellent effect in developing novel visualization strategies. Furthermore, UNIX based workstations are now becoming common in scientific, including geographical, research establishments, and in map production facilities, which deliver still more computing power to the user.

The hardware advances have been matched by advances in the available software. It is possible to see a progression from the implementation of exploratory data analysis within such packages as Minitab, the expansion in later versions of that program to contouring and mapping, and the more powerful graphics capability in more recent statistical systems such as Systat. Furthermore, we can observe the development of generic programs for graphics production and exploration such as the widely used UNIRAS. The latest generation of dedicated graphics software epitomize the visualization revolution in computer graphics, however, and include such systems as PV Wave, AVS, and Explorer, which are all commercially available, and deliver generic toolkits for the production of visual displays of many different types of data. The functionality of those is matched by software in the public domain such as apE and Khoros (Earnshaw and Wiseman, 1992).

SCIENTIFIC VISUALIZATION AND CARTOGRAPHY

Many visualization methods are grounded in earlier computer graphics and the possibilities of scientific visualization for cartography are often logical developments of computer cartography. Three ways in which map design and visualization interact are through expanding the variables available to the cartographer, the development of computational techniques which transform data prior to mapping, and interaction between the user and the map resulting in a dynamic user-defined representation of the data. Each is discussed below. Finally, discussion focuses on the possible benefits to visualization developments of historical cartographic research.

New Cartographic Variables

Dynamic Variables

DiBiase et al. (1992) have presented a full analysis of dynamic visual variables (after Bertin, 1983), and conclude
that there are three primary visual variables for dynamic cartography: order, duration and rate of change. Commonly in dynamic displays the ordering is by time, but this is not necessarily the case. Monmonier (1992b), for example, orders a series of images by attribute value to convey the correlation between variables in geographic space, and ordered by region to highlight geographic patterns. Similarly, Salvemini (1991; Figure 5) in attempting to map population of all the townships in Italy resorts to dynamic presentation ordered by size, to facilitate understanding in an otherwise cluttered display. Dorling (1992a, 1992b) presents several images of change with time, intended as both static maps and as animations. Egbert and Slocum (1992; Figure 6) provide an innovative generic system for statistical mapping, which allows animation with ordering by both region and by attribute.

In contrast, Fisher (in press a) uses no specific ordering, but rather random selection to visualize error in soil maps and in classified remotely sensed images. Shoup (1979) discusses animation of the colour table, colour cycling, whereby the colours painted on the computer screen are systematically changed. Present experiments by one author (JW) are examining this as a tool to aid visualizing circularly distributed data such as aspect, as a means of overcoming implied abrupt changes in the property which occur in many colour schemes, and continuous data which are usually represented with an arbitrary threshold, such as drainage networks defined by flow magnitudes.

Sound
Several researchers have suggested sound as a possible cartographic variable, but it is one which can only meaningfully be used in a computer environment. For example, Cassetteri and Parsons (1993) have suggested that in a GIS study of potential noise pollution (airport siting) results should be presented in audible form with the appropriate levels of the noise of taking off aeroplanes. Krygier (in press) gives a more complete overview of the potential applications of sound, including an analysis of the different sonic variables which may be used in cartography, and Fisher (in press b) shows how sound may be used to convey reliability in classified satellite images.

Computational Possibilities
Derivative Information
Geographical Information Systems (Burrough, 1986), and especially what is known as cartographic modelling (Tomlin, 1990), is concerned with formalizing the transformation of one or more spatial data sets into another. Thus from a matrix of elevation values the slope at a point can be found, and similarly, given a point in an area a field of bearings onto that point may be calculated. Such transformations are fundamental to visualization, as was recognised by Tobler (1979). Not only can data be transformed, but the geographical coordinate system itself may be rapidly changed. Particularly there is a recent and increasing use of cartograms which are very difficult to develop by hand, but relatively easy for computers, especially with the development of increasingly efficient algorithms for doing the calculations (e.g. Upton, 1991; Dorling, 1992a, 1992b, in press; Guscín-Zade and Tikunov, 1993).

Pseudo 3-D views
Terrain may be visualized in pseudo 3-D as an alternative to the topographic map, and presenting landscapes in such
views is a traditional cartographic technique, but one which is time consuming when performed by hand. Computer systems have facilitated this technique, and modern systems have extended it. The basic pseudo 3-D view of terrain has been discussed at length by cartographers (Robinson et al., 1984).

Terrain may be integrated with unrelated information of the same area (Figure 7), or with information derived from the terrain data by appropriate GIS functions (Figure 8). Such displays can be extremely impressive in their own right, and the benefits of this in error detection, for example, are discussed by Wood and Fisher (1993). Advanced algorithms allow the data to be viewed from any position and with a number of techniques to convey perspective, depth and detail. Several commercial systems can simulate haze, causing distant objects to be less clearly distinguished and colours to be attenuated. It is also possible to create landscapes as they might look as a result of environmental impacts such as building (Ervin, 1993) or forestry developments, or to test models, such as those governing fire behaviour (Bishop, in press). The landscape details may also be totally artificial (Figure 9) giving false impressions of the area depicted (Monmonier, 1991).

Animation
The potential to create multiple spatial data sets along with the new cartographic variables gives huge scope for animated mapping. Animation of time series cartographic data has a long tradition (Campbell and Egbert, 1990). It employs the ordering of a sequence of static views with slightly different characteristics to highlight change. Among the earliest examples of cartographic animation are Tobler's (1970) film of the development of Detroit, and Moellering's (1976) study of traffic accidents. More recently the advantages of temporal animation has been demonstrated by Van Voris et al. (1993) in their simulation modelling of forest evolution, and Weber and Buttenfield (1993) in animating US surface temperatures for the last century. When the changing variable is the viewer's location, movement is simulated. A series of images may enable a viewer to move across a static map, or zoom in and out. When viewpoints are sequenced a series of varying three dimensional views can be used for a number of tasks. Such a series is termed a fly-by, and can provide insight into unfamiliar or real, abstract data. Dramatic examples of fly-bys are the films of planetary exploration produced by NASA, showing tours of many of the satellites of the solar system, based on digital data gathered by satellite. Dorling and Openshaw (1992), on the other hand, explore the utility of fly-by and animation in socio-economic data.

Animation may also involve interaction whereby users can control duration, rate of change and order. For example users can interact with a terrain fly-by, controlling the viewing position and speed of movement. In a more abstract data space they may need to control these locational variables, but also the re-ordering of the data object. Interactive 3-D animations are essential in computer games and in more serious applications. Some of the most interesting are to be found in flight simulators for both the home computer and military training. In the former the graphic design to enable rapid updating of 3-D scenes on relatively slow computers are most interesting, while in the latter the power of the computers used makes such design compromises unnecessary. Very effective and rapid manipulation of terrain for flight simulation is available on Unix work stations, including games giving dog-fights between networked computers. In more serious vein, such fly- and walk-throughs are used in building and landscape design.

Solid Models
Full 3-D models can only be developed and explored in a computer environment; it is not possible to explore such diagrams on paper. Such models attempt to present voluminous data sets such as those of solid geology and hydrogeology, and atmospheric and oceanographic processes. Eddy and Looney (1993) show a water contamination plume at the Savannah River Nuclear plant in South Carolina. Rhine et al. (1993) illustrate a number of limnological and atmospheric applications of visualization, including the Antarctic ozone hole. A number of tools are available for examining and exploring the 3-D model including probes and planes to develop secondary views, and vector streamers to visualize directions on a surface.

The subsurface oceanographic and geological applications have in common a scarcity of spatial data, and an intensity of vertical values, either in ocean soundings or in boreholes. This creates very interesting interpolation and data management problems (Raper, 1989).

Interrogation
Exploration
Rapid access to the information which goes to make up the view is one of the major advantages of the computer environment, and just one aspect of interaction. The probes and planes mentioned above are just one example of this. Pop-up windows containing numerical information is another, and more advanced scatterplots of attributes, integrated with display of maps is another. Exploratory systems of this type include Polygon Explorer (MacDougall, 1992), Spider (more recently named Regard, Haslett et al., 1990), and EXPLORE MAP (Egbert and Slocum, 1992). In the last, for example, it is possible to recover not only attribute information for specific polygons on the map, but also ratios of any two polygons, and to show histograms of the attribute with the means, standard deviations, etc. (Figure 6b).

Multimedia and Hypermedia
The principles of hypertext, extended into other graphics, video and sound (multimedia) provide one mechanism for browsing through multiple different types of information about the same area or location (Wiggins and Schiffer, 1990). With the use of hotkeys (where implemented) to jump from one piece of information to another related but not necessarily juxtaposed piece, the user can rapidly get a very diverse view, much like using an encyclopedia, or reference book (Schiffer, 1992; Lindsay and Raper, in press). Batty (1992) presents a system integrating urban modelling and visualization which gives access to text, graphics, maps, statistical charts, and to complex analysis.

Established variables and designs
Computer-based interactive visualization does introduce some challenging questions, and reconsiderations of classic cartographic design guidelines. Some specific issues of design and layout might include whether a user needs the scale bar, north arrow, legend and many of the other
classic elements of the map. In an exploratory visualization session, the user should be able to acquire those and other meta information by interrogation if they are required, but does not need them on every map. The user's familiarity with the area makes them largely redundant. Indeed, visualization modules of Geographical Information Systems only rarely include these, although many allow them to be incorporated if desired, and for final, hardcopy maps. Clearly other aspects of layout are anathema when the user may zoom in or out at will, and re-adjust the view, removing the area illustrated (the study area) from any design control.

Visualization systems incorporate a rich toolkit for drawing information on the computer screen, but, as DiBiase et al. (1992) point out, they provide no guidance on the use of the tools. Thus the graphic variables of Bertin (1983), for example, are completely ignored by most systems and unfamiliar to many users, but the rigour implicit in the recognition of the variables would help many users of systems floundering in the richness, and possibly mis-using them.

Cartographers have a wealth of knowledge and a history of research in how to present complex data visually. Specifically, issues such as feature generalization, graphic symbolization, data classification, and user's reactions are all subjects well rehearsed in the cartographic journals. They are not, however, widely recognized in other areas of graphic design, and many visualization systems ignore some of the simplest rules. Most particularly the use of colour in some systems is based on very simplistic translation of colour tables, and on RGB colour space, and has nothing to do with human perception of colour, or colour theory developed over many years by cartographers (e.g., Robinson et al., 1984; Brewer, 1989; Monmonier, 1991). Indeed, a valid question for research is whether visualization systems need to incorporate this colour theory. Arguably, through familiarity with the data, users do not need perceptual colour schemes, but can recognize patterns irrespective of garish and clashing colours. The novice users, however, could be seriously disadvantaged in deriving appropriate interpretations if the theory developed for and by cartographers is ignored in visualization systems.

CONCLUSION

It has yet to be seen whether many of the visualization methods discussed here will be placed in the hands of na""ive users. They all assist scientific investigators in exploring their data, and in developing ideas. Many are very time consuming to use, are complex to understand, may be deceptive to some users, and are reliant on high-power computer platforms.

The static map is an adequate, even desirable, medium for the presentation of cartographic and scientific information, but we can already see users being presented with computer-based visualization tools to accompany (sometimes replace) paper publications (floppy disks and videos distributed in support of articles and electronic atlases are examples of these). Thus interactive visualization is an increasingly important way of not only exploring but also disseminating scientific information. As is so often the case with computer buzz word is rapidly being replaced by another and virtual reality is itself an outgrowth of scientific visualization, although one where the emphasis is on realism in interaction (Bishop, in press), not abstraction.

There are many questions to be asked about visualization methods; most particularly the effectiveness of many novel display methods needs to be demonstrated if they are to become part of widely available analytical packages. But currently we are on the threshold of major developments in the use of scientific visualization as a part of the cartographic toolkit.

ACKNOWLEDGEMENT

We would like to thank various colleagues, especially Mitchel Langford, Kate Moore, and David Unwin, for discussions, the outcome of which may inadvertently appear here unattributed. Kate Moore also prepared Figure 4.

REFERENCES


EXPLORING SPATIAL DATA REPRESENTATION WITH DYNAMIC GRAPHICS*

JASON A. DYKES
Department of Geography, University of Leicester, Leicester LE1 7RH, U.K.
(e-mail: jad7@le.ac.uk)

(Received 11 October 1996; accepted 14 November 1996)

Abstract—Dynamic mapping capabilities are providing enormous potential for visualizing spatial data. Dynamic maps which exhibit observer-related behaviour are particularly appropriate for exploratory analysis, where multiple, short-term, slightly different, views of a data set, each produced with a specific task or question in mind, are an essential part of the analytical process.

This paper and the associated coloured and dynamic illustrations take advantage of World Wide Web (WWW) delivery and the digital medium by using interactive graphics to introduce an approach to dynamic cartography based upon the Tcl/Tk graphical user interface (GUI) builder. Generic ways of programming observer-related behaviour, such as brushing, dynamic re-expression, and dynamic comparison, are outlined and demonstrated to show that specialist dynamic views can be developed rapidly in an open, flexible, and high-level graphic environment.

Such an approach provides opportunities to reinforce traditional cartographic and statistical representations of spatial data with dynamic graphics and transient symbolism which give supplementary information about a symbol or statistic on demand. A series of examples from recent work which uses the approach demonstrates ways in which dynamic graphics can be effective in complementing methods of measurement and mapping which are well established in geographic enquiry. © 1997 Published by Elsevier Science Ltd

Key Words: Dynamic maps, Visualization, Cartographic representation, Statistical representation, Tcl/Tk.

INTRODUCTION

Dynamic maps

The accelerating advances in computer technology are resulting in a host of electronic media and methods through which representations of spatial information can be constructed. Maps that provide more than a single static view of a spatial data set, or those which change in some way over time, are increasingly popular and becoming easier to produce. This paper and its accompanying dynamic illustrations present ideas on, and interactive examples of, ways in which dynamic maps can be used in the exploratory stage of the research process, with a particular focus on extending data representation with dynamic graphics.

Shepherd (1995) provides a broad classification of the various ways in which maps can incorporate dynamism. His typology of "the dynamic in visualizations" encompasses a wide range of changes that can occur in map displays, which he terms "map behaviour". These can come from a variety of sources and each form of behaviour can assist the observer in making sense of the displayed information. One such source is the map user, and under "observer-related behaviour" Shepherd (1995) describes map behaviour that responds to actions taken by the user. These include observer motion, object rotation, dynamic comparison, dynamic re-expression, and brushing.

Observer-related map behaviour has a wide variety of applications, providing the user with control over what aspects of a spatial data set they see, and how and when they see them. Two-dimensional representations are the concern in this paper, where observer motion and object rotation equate to changing the observer's viewpoint by rotating, zooming, and panning. There are analogies with folding, rotating, and using a magnifier on a paper map. The remaining types of behaviour, dynamic comparison, dynamic re-expression, and brushing, involve changing map symbolism over time.

Shepherd's use of the term "brushing" includes a whole host of data selection procedures which date from the late 1980s when statisticians recognised the analytical utility of direct manipulation and instantaneous change in their graphs (Becker, Cleveland, and Wilks, 1987). Techniques were introduced whereby cases could be identified with "transient labels", deleted from a view to provide a "focused" subset, and linked by the "transient highlighting" of selected entities in statistical views of

*This paper was written specifically as an electronic document (see the CD version) and the printed version may be awkward to follow in places.
data and the corresponding entities in separate views. Each technique involved temporarily symbolising with a unique colour or shape. By plotting coordinate pairs on orthogonal axes a spatial view of the data could locate selected entities on statistical views and select regions on spatial views for statistical analysis. The addition of polygonal boundaries for enumerated data resulted in a more familiar map representation (Monmonier, 1989), and techniques for geographic brushing were enhanced and developed (Haslett, Wills, and Unwin, 1990; MacDougall, 1992).

Dynamic re-expression involves alternating more than one graphical version of a data set. Shepherd (1995) generalises this technique to include the interactive modification of cartographic display parameters.

Dynamic comparison entails displaying more than one data set, in the same coordinate space at different (usually successive or cyclical) times. The techniques are an extension of Tukey's ideas on alternating graphics (Tukey, 1973).

Egbert and Slocum (1992) used these methods in EXPLORERMAP (or EMAP). The software provided users with control over the design, symbolism, and class delimitation of choropleth maps as well as incorporating transient symbolism techniques to display significant values within map classes and brushing for identifying regions. Dynamic comparison was provided through the cycling of attribute classes in order, in a technique developed by Taylor (1989).

Each of these observer-related techniques involves interaction with dynamic views of data by identifying entities in single or multiple views and applying transient symbolism to the graphics that represent them to show some characteristic of the entity. The ability to interact and symbolise in such a temporary way has numerous uses, and the provision of software tools which encourage it, such as those described previously, has resulted in a change in emphasis of map use from being presentation tools to playing a significant role in the process of knowledge construction (MacEachren, 1995).

Map uses for observer-related dynamic behaviour

A graphic framework for identifying and differentiating map uses is provided by MacEachren's (Cartography)3 (MacEachren, 1994), where a threedimensional space of potential uses is defined by three continua. These range between: (1) private and public map use, (2) revealing unknowns and presenting knowns, and (3) high and low levels of interaction with the map. The types of representation permitted by the observer-related map behaviour are particularly suited to the process of producing individual views of data that answer private expert queries in an attempt to reveal unknown information. MacEachren (1994) labels this corner of the map-use cube as "visualization", and the term is used in that context here.

At this stage in the research process analysts are generally unaware of what they want to see until they see it, but they will have ideas about how they might search for it and what representation might be useful (MacEachren, 1995). Brushing and re-expression are examples of the kinds of techniques that are being used increasingly across the sciences at an exploratory stage, to investigate large, complex, and multivariate data sets, by sifting through varying views of data until interesting representations appear (McCormick, DeFanti, and Brown, 1987). The more interactive this process, the faster questions can be answered and thought patterns developed. Additionally, flexible graphic software will provide required views more quickly than before, and cause fewer obstacles to restrict the research.

How to map for visualization?

Shepherd (1995) reports that there is little experimental evidence upon which to base rules for making the most appropriate use of dynamic symbolism methods. Indeed there is insufficient indication that dynamic maps are advantageous even for communicative purposes. Slocum and Egbert (1993) conclude from their experiments with class sequencing that tailor-made dynamic products for individual consumption might provide more advantages than their investigations uncovered, and Koussoulakou and Kraak (1992) detect no overall communicative advantage from using dynamic maps, but suggest that those familiar with dynamic methods might find them more useful than static methods. Each of these conclusions points to the potential utility of individual transient maps for experienced users who know the data, for visualization purposes, but MacEachren (1995) notes how little we know either about how maps work in the visualization context, or about appropriate tools to provide, and DiBiase and others (1994) recognise the problems of evaluating exploratory methods. McGuinness (1994) contributed evidence for using dynamic methods when she found that experienced users would create and re-create individual transient composite maps of relatively few variables when analysing a series of distributions in her test of techniques used for a visualization task.

Lindholm and Sarjakoski (1994) concede the difficulty in designing maps for experts and make a recommendation that the power and the flexibility of a map to be used for visualization should not be sacrificed. We have not yet begun to explore the potential for using multiple cartographic representations of data sets to allow researchers to see their data from different perspectives (MacEachren, 1995). An environment which encourages experimentation with cartographic representation is essential if we are to start doing so. Map design for
Exploring spatial data representation

visualization may currently be less concerned with appropriate representation than with defining and providing an appropriate flexible environment for exploratory dynamic cartography.

A dynamic mapping environment for visualization

DiBiase and others (1994) define cartographic visualization software as being dynamic displays comprised of an interface tailored to a specific set of users, which is interactive, and encourages experimentation with different combinations of data and graphic symbols, and which is intended for use in fostering discovery rather than presenting conclusions.

In describing the types of interaction that cartographic visualization software might permit DiBiase and others (1994) list selection and transformation operations which broadly concur with Shepherd's concepts of brushing and dynamic re-expression (Shepherd, 1995). They argue that if an analyst is able to change their perspective of the data through selection and transformation, then meaningful relationships amongst data variables are more likely to be revealed.

To achieve this, a dynamic environment for visualization should permit the location of multiple graphic symbols in space with the variety of symbolism outlined by Bertin (1983) and others. It should manage and record the relationship between the symbols which constitute the map and the user, and change the way in which symbols convey meaning. A more general situation is the capability of a symbol to issue software commands when it is touched, or interacted with in a specified manner, in the way that graphic user interface (GUI) widgets do. (These may or may not involve changing the symbolism of the touched, or some other, symbol.) Lindholm and Sarjakoski (1994) refer to such symbols as "smart objects". A dynamic map for visualization could be made up of smart symbols which permit user-related dynamism through transient labeling, transient symbolism, and transient graphics and might link to other views and media.

A smart map, then, would provide the ability to create views by locating symbols in space, show multiple perspectives through re-expression and brushing with transient symbolism, and also be capable of incorporating the kind of multimedia linking that is being used successfully for communicative purposes (Cartwright, 1997) to provide additional information about the phenomena represented by selected symbols.

To retain flexibility and openness, an environment of this sort would be essentially programmable, and allow experts to prototype their representations in the search both for individual maps which illustrate unknowns, and for suitable techniques which provide assistance in doing so. R. Bivand (written comm., 1996) notes that for exploratory analysis most commercial products permit and require extensive customisation and implement "little languages" which support scripting and interface construction at a high level. By programming at this level, researchers can rapidly experiment with representation and prototype new dynamic views of data in a "hands on" fashion to suit particular, transient, tasks.

This paper presents an environment which meets many of these criteria. It provides an interactive introduction to Tcl/Tk (Ousterhout, 1994), a graphical scripting language with an open and flexible development environment and many desirable features for modeling and visualizing discrete spatial entities in two spatial dimensions (Dykes, 1996a). The following section consists of an interactive introduction to Tcl/Tk for dynamic cartography. It uses the digital medium to review and build upon some of the ideas previously presented in static form (Dykes, 1996a) and gives a taste of the features. Then, by way of example, the focus falls on employing dynamic graphics to enable users to investigate cartographic and statistical representations of spatial data in later sections. At the time of writing the work is at "proof of concept" stage, with the hope that further development and implementations will result. The latest examples and developments can be accessed via the World Wide Web (Dykes, see URLs Appendix).

AN INTERACTIVE INTRODUCTION TO TCL/TK FOR CARTOGRAPHIC VISUALIZATION

Tcl is a scripting language that provides variables, substitution, arrays, lists, control structures, and file management facilities. Tk is a GUI toolkit based on Tcl, which enables Tcl commands to be run by graphic widgets. Collectively they are known as Tcl/Tk (Ousterhout, 1994) and are available on the Internet. Tcl/Tk has a broad community of users who primarily use the GUI widgets such as buttons and entry boxes to build interfaces for and between programs. However, the programming and graphics structures can be utilised to provide the kind of visualization environment described in the previous section.

Following the launch of a Tcl/Tk "plug-in" for Netscape 3.0 from Sun Microsystems, on July 10, 1996, it is possible to demonstrate the approach by distributing Tcl/Tk code over the Internet to be interpreted by local browsers. The developers are promoting Tcl/Tk as a "Universal scripting language for the Internet". This could be of significant value to visualizers, as the code and graphics are readily distributed for both communication and verification (R. Bivand, written comm., 1996). Currently researchers are likely to find that access speeds, memory constraints, and problems associated with distributing data sets (such as copyright, liability, the value of data, and personal investment...
in data collection and enrichment) restrict the amount and type of data that they can publish in the public domain using interactive graphics on the WWW. However, the significance of these issues is reduced if WWW access is restricted to a small group of known colleagues.

The following text, code, and graphics require the Tcl/Tk plug-in. They illustrate ways in which Tcl/Tk can provide a suitable environment for visualization. Two types of graphic are used, each of which may be static or dynamic.

The graphics labeled “Example” rather than “Figure” are embedded Tcl/Tk source code which browsers with the Tcl/Tk plug-in will interpret. They are simple illustrations which are provided for communicative, rather than truly exploratory, purposes in this instance. They use a small, generalised, example data set to reduce access times and minimise the computer specification required to load and run them as this paper is intended for a wide audience. Most of these examples demonstrate dynamic features and allow interaction. To inspect the source code used to create the examples, click the tick, “✓”, by the title.

The graphics labeled “Figure” are small, canned animations with no interactive capabilities. They are designed to give an impression of the functionality and dynamic nature of the Tcl/Tk graphics used in practical applications with real data sets, but once again their role is a communicative one. Full, stable, debugged, working examples for exploratory data analysis, containing large digital data sets, are not within the scope of this document. Some more ambitious examples can be downloaded from the URLs listed in the Appendix.

The fundamentals of Tcl/Tk

Four basic Tcl commands are shown below to demonstrate some essential Tcl/Tk syntax.

Note that the “set” command is used to set variable values, and a preceding dollar sign leads to a variable value being substituted for the variable name.

A Tcl/Tk evaluator is provided, through which these, and other Tcl/Tk commands used in this document, can be tried and tested.

```tcl
#- -E x a m p l e s  o f  s o m e  b a s i c  T c l  c o m m a n d s .
set n 1
set array(1) "A Number"
set array(2) 22.5682
puts "Here is $array($n) ($array(2))"
> Here is A Number (22.5682)
```

Sets a variable to an integer value
Sets an array element to contain a text string
Sets an array element to contain a real number
Prints the variables to the screen.
Tk example 1. Tk consists of several classes of widgets. Objects from each class can be created by stating a widget class, such as "button", and a name for the object. Names are hierarchical, rather like the DOS or UNIX file-naming convention, but full stops (".") are used to separate names rather than slashes ("/" or ":"). Each object derives default characteristics depending upon its class, but these can be varied by adding a relevant value to a legitimate option when defining a widget. These are encoded as "option-value" pairs. For example a button widget can have a value designated for its "text" option by adding "-text value"; or for it's "height" option by adding "-height value".

Tk Example 1. shows five widget classes, each with relevant options and suitable values. The "-bg" option sets the background colour of the widget. Colour options accept either an X-Windows colour or a hexadecimal red green blue specification preceded by a hash, where "#F00" describes red, and "#FFF" white. Command options accept any relevant Tcl Tk command and execute it when a widget is clicked. A "pack" statement is used to display the defined widgets. Note how the checkbutton has been defined to store a value in variable "v" dependent upon its state with the combination of "-variable" "-onvalue" and "-offvalue" option-value pairs, and that this value is shown in the yellow label as it has a "-textvariable" option, which specifies that the text to show in the widget is the value of "v". This means that whenever the checkbutton, or red button are pressed, the value of "v" and the resulting label text change. Appreciation of the dynamism provided by GUI objects such as this which pass messages between one another and can be programmed to interact in a specified way is provided by interacting with the widgets.

```tcl
# -- Tk Example 1.
button.b -text Hello -bg Red -command { set v Click}
label.11 -text "A Label" -bg #0f9
scale.s -from 10 to 20 -orient horizontal -variables
checkbutton.k -onvalue On -offvalue Off -variable v
label.12 -textvariable v -bg Yellow
canvas.c -width 50 -height 25 -bg SeaGreen4 -relief sunken
pack.b.11.s.k.12.c -fill x
```

Tk example 2. Once Tk widgets have been defined, widget names become commands which can be used to affect the values of widget options and so change the appearance and behaviour of the widget. In this example, the button "b" has its background colour changed with a command of the form "widgetname configure -option value". The canvas "c" has its height increased and colour changed in a similar way. The value of the "command" option can also be altered. Here the red button is provided with a command which resets the scale widget ".s" to a value of ten.

Bindings provide a means of linking mouse cursor events to widgets. There are a whole range of cursor events recognised and managed by Tk. They are defined with commands of the form:

```
bind widgetname <Event> {TclScript}
```

In this example the label ".l" is bound to commands which change the value of the variable "v" depending on which of the mouse buttons is clicked over the label. The visual effect is immediate as the "-textvariable v" option-value pair ensures that the value of "v" is always shown in the yellow label.
Tk example 3. These features are useful in themselves, providing colour symbolism, text labels, and communication between "smart" objects, and the kind of dynamism required in a visualization mapping environment, but little control over the location or shape of widgets which are required for mapping. The canvas widget class provides control over the location of objects. Canvases comprise a planar coordinate system with an origin at the top left of the widget, and classes of graphic symbols, known as items, which can be located on the plane. Line, polygon, oval, rectangle, arc, bitmap, and image items can all be located and assigned a variety of visual variables through their option-value pairs. This forms the basis of an environment for smart mapping as items are dynamic objects which can be assigned spatial locations and symbolism which reflects data values.

In Tk Example 3, the canvas is expanded, and four items are defined. Oval items require coordinates to define their extremities, and lines and polygons expect a list of x-y coordinate pairs. The "-fill" option specifies the colour with which an item is filled. Other options available include the outline width and/or colour and the name of a bitmap which is mapped into the symbol.

The "-tags" option is used to add one or more identifying labels to each item. These are the equivalent of widget names as they provide an identifier through which items can be reconfigured to change their symbolism or behaviour. Here colour, shape, and simple geographic names are used as tags, but any string can be used. Items can be tagged with any number of strings and tags thus provide a way of adding attribute information to symbols.
Tk example 4. Just as widgets can be reconfigured with widget commands, tagged items can be used to change symbolism or issue commands. Canvas commands are the key to dynamic mapping in Tcl/Tk. A canvas command to reconfigure the symbolism of an item has the form:

```
canvasname itemconfigure tag -option value [-option value ...]
```

In Tk Example 4 all items with the “road” tag are given a width of eight pixels and a yellow fill colour. A second command shows that tags can be changed temporarily, adding the tag “lake” to an item with the “blue” tag. Temporary tags are appropriate for transient labeling or symbolism. Finally four item bindings are demonstrated. They have a similar form to the widget bind commands:

```
canvasname bind tag <Action> {Tcl script}
```

In Tk Example 4 clicking items with the “red” tag with mouse buttons 1 and 2 (usually the left and central mouse buttons, but the exact outcome will depend on local mouse and driver configuration) will change the colour of items tagged “line”. The “water” tag is bound to commands that colour all “water” tagged items red when a “water” item is touched with the cursor, and blue when the cursor leaves the item.
Tags are also used to link all items to a procedure which scales the symbols' coordinates whenever a scale bar is moved. The colour and width tags are unchanged by the procedure which can be viewed by clicking the tick “✓” by the example title.

**Tcl/Tk for dynamic cartography**

These basic properties provide an appropriate environment for cartographic visualization. The rest of this section is used to demonstrate some examples of how spatial data can be mapped for exploratory analysis. The data used for these examples depict the ten British regions and illustrate some of the obstacles presented to those wanting to distribute dynamic maps on the WWW. The spatial data show only a few, highly generalised examples for reasons of both simplicity whilst trying to communicate ideas and access speed. Copyright is an additional important consideration when distributing information in the public domain. All the attributes used here for the British regions are hypothetical as regional data for the U.K. are subject to Crown copyright and thus not distributable in digital form. The emphasis of these examples is on the dynamism, not the numbers.

In these examples the data are stored in four arrays. Tcl contains powerful file-handling and data-manipulation capabilities for incorporating data sets for real applications. The arrays represent information for the regions numbered 1-10. The “poly” array contains ten elements, each is a list of x and y coordinate pairs that comprise an enclosed polygon representing a region. Brunsdon and Charlton (1996) have outlined some of the advantages of modeling spatial data in list form. Centroids for each polygon are stored in two arrays, “x” and “y”, which have ten single numerical elements. The spatial data are scaled to fit the 200 x 200 pixel canvas used throughout the examples, but Tk does provide scaling, zooming, and scrolling capabilities which are not demonstrated here. The attribute data consists of twelve variables, scaled between 0 and 100. They are stored in an array of dimensions 13 x 11 which has the form:

```
data(zone-number,variable-number)
```

The names of the appropriate region or variable are stored in data(zone-number,0) and data(0,variable-number). The full data set is available for inspection. A list of zone numbers, “zones”, contains the identity of each of the included zones and is used for looping purposes:

```
set zones “12345678910”
```

The code provided in the text is essentially accurate, but stylised slightly in places. The Tcl/Tk code is interpreted locally and embedded within your browser. It can be viewed by clicking the ticks, “✓”, by example titles and using a browser’s “View Source” facility.

**Creating a static map view.** Static maps and statistical views are created by locating symbols on canvases in a way that reflects data values. In Example 1 a canvas is named “view 1”. A “foreach” loop is used to progress through each element of the list “zones”, setting the variable “zone” to each of the values in turn.

Two commands are issued for each value of “zone” to create a shaded polygon map. First a variable “intensity” is calculated using the “expr” command which returns the answer of a mathematical expression. The variable is comprised of two parts, the string “Grey”, and the result of the expression which subtracts the attribute value held in the “data” array at the location specified by the zone number and variable 1. For example, a data value of 42, which is the value of variable 1 for zone 10, will result in “intensity” being set to Grey58, which is one of 100 legitimate X-Windows grey shades. This scale of colours can be used to shade canvas items and so produce a virtually classless choropleth (Muller and Honsaker, 1976).

Secondly a polygon is created, as shown previously using the list of x and y coordinates con-

```
data(zone-number,variable-number)
```

The names of the appropriate region or variable are stored in data(zone-number,0) and data(0,variable-number). The full data set is available for inspection. A list of zone numbers, “zones”, contains the identity of each of the included zones and is used for looping purposes:

```
set zones “12345678910”
```

The code provided in the text is essentially accurate, but stylised slightly in places. The Tcl/Tk code is interpreted locally and embedded within your browser. It can be viewed by clicking the ticks, “✓”, by example titles and using a browser’s “View Source” facility.

**Creating a static map view.** Static maps and statistical views are created by locating symbols on canvases in a way that reflects data values. In Example 1 a canvas is named “view 1”. A “foreach” loop is used to progress through each element of the list “zones”, setting the variable “zone” to each of the values in turn.

Two commands are issued for each value of “zone” to create a shaded polygon map. First a variable “intensity” is calculated using the “expr” command which returns the answer of a mathematical expression. The variable is comprised of two parts, the string “Grey”, and the result of the expression which subtracts the attribute value held in the “data” array at the location specified by the zone number and variable 1. For example, a data value of 42, which is the value of variable 1 for zone 10, will result in “intensity” being set to Grey58, which is one of 100 legitimate X-Windows grey shades. This scale of colours can be used to shade canvas items and so produce a virtually classless choropleth (Muller and Honsaker, 1976).

Secondly a polygon is created, as shown previously using the list of x and y coordinates con-

```
data(zone-number,variable-number)
```

The names of the appropriate region or variable are stored in data(zone-number,0) and data(0,variable-number). The full data set is available for inspection. A list of zone numbers, “zones”, contains the identity of each of the included zones and is used for looping purposes:

```
set zones “12345678910”
```

The code provided in the text is essentially accurate, but stylised slightly in places. The Tcl/Tk code is interpreted locally and embedded within your browser. It can be viewed by clicking the ticks, “✓”, by example titles and using a browser’s “View Source” facility.

**Creating a static map view.** Static maps and statistical views are created by locating symbols on canvases in a way that reflects data values. In Example 1 a canvas is named “view 1”. A “foreach” loop is used to progress through each element of the list “zones”, setting the variable “zone” to each of the values in turn.

Two commands are issued for each value of “zone” to create a shaded polygon map. First a variable “intensity” is calculated using the “expr” command which returns the answer of a mathematical expression. The variable is comprised of two parts, the string “Grey”, and the result of the expression which subtracts the attribute value held in the “data” array at the location specified by the zone number and variable 1. For example, a data value of 42, which is the value of variable 1 for zone 10, will result in “intensity” being set to Grey58, which is one of 100 legitimate X-Windows grey shades. This scale of colours can be used to shade canvas items and so produce a virtually classless choropleth (Muller and Honsaker, 1976).

Secondly a polygon is created, as shown previously using the list of x and y coordinates con-
tained in the array "poly" in the element specified by the value of "zone". The polygon is filled with the calculated "intensity" and given a unique tag for future reference comprising the letters "id" followed by the zone number (for example id1). The map shown in Example 1 is the result of looping through the ten cases. Maps for larger data sets can be developed from spatial data files with an identical loop. The RGB colour definition capabilities mean that appropriate and alternative colour schemes can be applied to the symbols, such as those advocated by Brewer (1994).

Alternative map views can be created by using different or additional symbols. In Example 2 a graduated circle map is produced using polygons and ovals. Statistical views can be created by plotting symbols on a canvas in attribute space where canvas coordinates are taken from the "data" array, rather than cartographic space. Examples are provided later in this section.

**Adding dynamism: changing map symbolism.** Transient symbolism can be applied to items with specific tags from item bindings, from widget commands, or via the command line. This permits brushing, dynamic re-expression, and dynamic comparison. In Example 3, a procedure is defined and called from the "-command" option of a button to perform the latter two of these. Procedures are defined with a standard Tcl "proc" command of the form:

```
proc procedurename { arguments} {Tcl commands}
```

They are called by the procedure name followed by values for the necessary arguments. For example a procedure to add two values would be defined and called as follows:

```
proc addvalues { v1 v2} {
    set answer [ expr $v1+ $v2]
    puts "$v1+ $v2= $answer"
}

addvalues 10 4
```

Procedures provide an efficient way of repeating frequent tasks or recreating required view types. In Example 3 a "foreach" loop creates twelve buttons named ".bl" to ".b12" with options that show the
appropriate variable name, and call the procedure "rcconfig" with a single argument, the variables number from 1 to 12. The procedure uses the global variables "data" and "zones". A "foreach" loop structure calculates the temporary variable "intensity" for each zone number in turn, based on the value stored in the data array at the element specified by the variable number passed into the procedure. A canvas command is used to configure the item tagged with "id$zone", so that it is filled with the value of "intensity". Variable number 1 is thus mapped with the command "reconfig 1".

Example 3. Transient symbolism — changing the map.

In Example 4, both the polygon map and circle map are changed by a redefined version of the "reconfig" procedure which calculates new radii and bounding coordinates for the graduated circles and changes circle configuration in "view2" as well as that of the polygons in "view1".

Example 4. Transient symbolism — changing multiple views.

Linking symbolism changes to the user. Buttons are an appropriate way to invoke dynamic re-expression and dynamic comparison. Other observer related map behaviour, such as brushing and labeling, require direct interaction with symbols. This is achieved with item binding. Example 5 takes adva-
tage of item binding to interrogate symbols for information. It uses transient symbolism and produces transient labels.

Tk bindings can have two forms, as shown next, each of which are employed in Example 5. The first is used to bind a series of Tcl commands to items with a particular tag when some cursor/mouse action takes place, the second binds the cursor/mouse action to any location on the canvas, irrespective of items and tags:

```tcl
canvasname bind tag <Action> { Tcl commands}
bind canvasname <Action> { Tcl commands}
```

Bindings provide users with access to the way in which Tk controls and records the interaction between the mouse, cursor, and canvas. For example, Tk gives the tag "all" to all items on a canvas and adds a tag "current" to any item that the cursor touches. The following binding, which is included in Example 5, will thus cause any item that is clicked with mouse button 1 (usually the left button) to be coloured in red.

```tcl
.viewl bind all <Button-1>
{ .viewl itemconfig current -fill Red)
```

Tk also records and provides access to the location of the cursor in canvas coordinates. These can be passed to a procedure as part of a binding with "%x" and "%y" and used to report locations, or create graphics, as they are in the following binding which calls the procedure "show_location" with two arguments. This binding can also be tested in Example 5, by clicking mouse button 1 twice.

```tcl
.viewl bind all <Double-Button-1>
{ show_location %x %y)
```

To interrogate symbols for attribute or locational information the id of a selected symbol has to be detected, and the relevant information retrieved from the data array and displayed in some graphical or textual way. In Example 5 a label is defined with the "-textvariable" option set to "info" meaning that the value of the variable "info" will always be shown in the label. Two bindings are then used to extract attribute information from symbols that the cursor enters.

The first of these binds all items to a series of commands whenever an item is entered using the "< Any-Enter >" action which Tk manages. The commands set the variable "atts" to contain the result of a Tk canvas command which returns all of the tags of the "current" item. The id ("zone") is then determined by applying some Tcl commands that operate on lists to the list of tags returned to "atts". The result is an integer representing the zone, which is used to retrieve information from the data array. The variable "info" is then set to contain the name of the current region, the name of the currently chosen variable, and the value for that combination of region and variable. The current zone is given an orange outline and is raised to the front of the canvas with another Tk command which has the form "canvasname raise tag". This ensures that it is distinguishable from other items and visible above them. The second binding states that whenever the cursor leaves an item, it's outline is returned to black.

Example 5. User interaction with symbols — interrogating for values.

More elaborate bindings of this sort can be used to add transient labels to the canvas, and to ensure that like tagged symbols are highlighted in multiple views. Transient labels can be created with line and text items, and linking is achieved by highlighting like tagged items in more than one canvas. Some possible bindings are illustrated by Example 6 in which the polygon and circle maps are linked, and where more durable brushing is invoked by clicking symbols with the mouse button.
These kinds of dynamic graphics capabilities provide plenty of scope for the kind of open environment for experimentation that DiBiase and others (1994) recommend. They mean that researchers can control how, and why, and when maps are dynamic and extend the representation possibilities for cartographic visualization.

Another persuasive reason for using Tcl/Tk is its openness and flexibility. The interpreted nature of the language, and the open shell and command line allow algorithms and graphics to be changed or "tweaked" to suit individual needs, questions, and data, in an interactive way recommended by Brunsdon and Charlton (1996) and MacEachren (1995) (who observes that visualization tools should permit the interactive selection of different representations, each of which might enhance a particular aspect of the data). The scripted nature of the code means that users can alter representations rapidly.

A piece of visualization software, called "cdv" has been developed to demonstrate the use of these and other features of Tcl/Tk with a real enumerated data set, and to promote visualization in research and teaching in the spatial sciences. The software has been produced to suit the needs of the U.K. Population Census of 1991 and to be appropriate for other enumerated data sets. It allows users to select from a series of logical representational choices as advocated by MacEachren (1995). "cdv" is free-ware and provides the most complete application of the dynamic mapping techniques outlined here. (See URL Appendix which follows References.)

The following two sections consist of annotated and animated examples of the ways in which Tcl/Tk widgets can be used to produce dynamic graphics which respond to interaction from the user. Each shows additional information to the mapped value when symbols are interrogated, encouraging exploration of the representation. Some apply transient symbolism to enduring symbols, whereas others use transient symbols. The examples use development scripts that extend both the previous examples and the "cdv" code.

EXPLORING CARTOGRAPHIC REPRESENTATIONS

This section contains examples which demonstrate the utility of the approach by using observer-related map behaviour to provide additional information about a spatial data set which is not apparent in static views.

Spatial structure

Tk items possess spatially "smart" characteristics which make them particularly useful structures for dynamic mapping. Canvas commands exist to find the bounding box of an item and to add tags to all items which overlap or are enclosed by a bounding box, and the closest item to a canvas coordinate. The tags can then be used to highlight items that overlap, fall within, or are closest. These commands have the following forms:

```
canvasname bbox tag

canvasname addtag tag overlapping
bottom right

canvasname addtag tag enclosed
bottom right

canvasname addtag tag closest

Example 7 takes advantage of these features. It contains two views, the familiar polygon map, and a scatterplot, which shows the relationship between two variables. Scatterplot axes provide a label showing the variable mapped on an axis when it is touched. The variable can be changed by clicking an axis to make it active (the active axis is normally coloured orange) and selecting a variable button. The button invokes a procedure which moves the
location of each scatter plot symbol depending upon the data value of the selected variable.

The polygon map uses the “closest” command to select the closest item to any location. Double click on the canvas to create a transient symbol showing a location, then click the symbol to highlight the closest polygon. The related item is highlighted in both views.

The “bbox” command can be used to create a rectangular brush and “overlapping” to find items within it. Rectangle brushing is implemented with this technique. Drag the mouse whilst depressing mouse button 1 and the keyboard shift key, over either canvas, to create a transient rectangle. Click the rectangle to select overlapping items in the view, cases relating to which are highlighted in all other views. Transient symbols such as the rectangular brush can be removed by clicking them whilst depressing the “Ctrl” key.

Neighbourhood information is calculated by applying the “bbox” and “overlapping” commands to map polygons. Hold down the “Shift” key and touch any item in any view. This results in the appropriate symbol being highlighted dark green in both views. All items within the bounding box of the highlighted item in the spatial view are identified, and then highlighted in both views in a lighter green. The rectangle is drawn around the selected item in the polygon map for reference (visualizing the algorithm). These techniques add a spatial component to the statistical views (Dykes, 1996b).

Contiguity relationships are implicit in enumerated digital data sets and neighbourhoods at a variety of spatial lags can provide a useful basis for spatial analysis. Whereas this method will not result in a list of the contiguous zones, it does provide a subset for testing contiguity, from which a list can be developed efficiently. Anselin and Smirnov (1996) have demonstrated that a list structure provides an efficient way of generating higher-order adjacency matrices. An algorithm that provides a graphical statement of progress is implemented in “cdv”.

The nature of a dynamic “contiguity probe” is demonstrated in Figure 1, through an animated sequence of five images. They show how a data set comprising the wards of the county of Leicestershire has been mapped in “cdv” and can be dynamically interrogated for information about the way in which data for a continuous region have

Example 7. Spatial “smartness” — rectangle brushing and neighbourhoods.

Figure 1. Contiguity probe.
been collected in discrete areas. The central orange zone shows where a cursor is moved across a representation of Leicestershire, and contiguous neighbours are coloured dark green. Their neighbours, those which are second-order links from the orange zone, are coloured lighter green, and so on, up to a spatial lag of order 4 from the interrogated zone.

**Attribute composition**

Transient symbols can be combined to produce transient graphics for the interrogation of data values, to show more information at a location than is provided by a single symbol. Such techniques have been used for exploratory analysis of a spatio-temporal data set which forms part of a project concerned with tourist behaviour, based at the University of Wageningen, The Netherlands. Tk is used to map tourists located on roads and pathways, or in areas, of a German park at different hours of the day. Symbols are linked to reconfiguring commands issued by a scale bar, to produce a time-series map which permits interactive comparison between distributions throughout the day. Additionally a GUI is provided so that tourists can be included in, or omitted from, the map according to their attributes. So, a time-series map can be created of a subset of the tourists, for example, Dutch people who arrived by bicycle. This provides a method for exploring an extremely rich data set (see Dykes, 1996a).

Figure 2 shows a series of line and circle symbols which represent numbers of tourists at different locations throughout the day. Transient graphics can be summoned by the observer to provide a temporal overview of each location. When any symbol is clicked, additional items are produced to create a line graph which reveals numbers of people at that location at each hour through the day. If subsets of individuals are selected, the graphs are shown for each subset. In Figure 2 the green line shows the total number of tourists, as does the map. The light blue line shows the number of men, and the purple line the number of women. The graphs are dynamically linked so that lines can be highlighted in each view. As the scale bar is used to select different hours of the day the symbolism changes and the selected hour is shown by a red indicator on the graphs of attribute composition. Figure 2 shows these changes with an animated sequence of five images, representing five hours chosen with the scale bar (not shown).

A second example comes from an application which maps employment structure for an enumerated data set through a dynamic choropleth map. Interrogation capabilities are provided by reconfigur-
guring items in an enduring additional view. This view is the human shaped dynamic graphic shown in Figure 3. It appears next to the map, which shows the spatial distribution of selected variables in a similar way to the examples outlined previously. When a zone on the dynamic choropleth map is touched, the graphic is reconfigured with stripes, the height of which show the percentage of employment in various sectors of the economy as recorded by the U.K. Census of Population.

The nominal classes are distinguished by contrasting hues and bitmaps. A pick-axe for "Agriculture and Mining", a computer for "Manufacturing", a chef's hat for "Hotel and Catering", a dollar sign for "Money and Finance", and so on. The icons’ dress appears to change as the cursor is moved around the region and as the map designer pointed out, an icon with "money bag pants" and a stripy jumper might represent a good location for industrial investment, with plenty of financial skills and a broad industrial base. The application also contains a toggle button which reconfigures the icon shape to a more sedate rectangle. Figure 3 shows how the graphic changes as the cursor is moved across a number of zones. The animated sequence gives an impression of the dynamic nature of the additional view.

Cartographic abstraction

A third aspect of maps which can be investigated with observer-related dynamic techniques are spatial transformations performed when mapping. Cartograms use the visual variable of space to depict an attribute (usually population) but retain a partial mapping of geographic space (usually restricted to the topological aspects of it) MacEachren (1995). Dynamic graphics permit the spatial dimensions to be used to map more than one variable almost instantaneously.

Strachan (1996) praises Dorling’s non-continuous cartograms (Dorling, 1995), but expresses a familiar drawback that the identification of places of interest is not always easy. He suggests the provision of a transparent overlay with an atlas of cartograms. This section illustrates how abstract spatial representations, such as those used by Dorling (1995), can be deciphered with smart symbols.

A dynamic cartogram of the kind shown in Example 8 provides two aids to interpretation. Brushing between a geographic and a cartogram view is of assistance in identifying locations, and a scale bar is linked to a procedure which interpolates between the spatial and non-spatial views using the “canvasname tag coords” command to change the coordinates of the circle items. The first-order contiguities are visualized for reference, and oval symbols can be dragged with the mouse to investigate the adjacency structure. This technique is analogous to the Grand Tour between alternative statistical views of data sets (for example, see Cook and others, 1995) and is a smoothed form of alternative graphs. Dragging is implemented by combining the cursor action with a “canvasname tag coords” or “canvasname tag move” command to redefine or increment coordinates.

The following figures use the Leicestershire data set. They take advantage of dynamic linking to explore and reveal the locational characteristics of the cartogram and the success in retaining the topological nature of geographic space. In Figure 4 the location of any probed cartogram symbol is identified in the accompanying polygon map and vice versa. Adding a label which names the zone is achievable in the same way as it was in Example 6 above.

Figures 5A and 5B use the contiguity probe to reveal the spatial structure of both representations. The polygon map is probed for contiguous neigh-

Example 8. Dynamic views to aid interpretation of cartographic abstraction.
Figure 4. Brushing linked cartograms.

Figure 5. A. Spatial structure through linked cartograms — transient symbolism. B. Spatial structure through linked cartograms — transient symbols.
bourhoods at a variety of spatial lags. The cartogram is updated immediately and the success with which neighbourhoods are retained can be judged (it will of course be spatially variable). Equally the cartogram can be probed for neighbourhoods which are shown on the polygon map. The cartogram neighbourhoods can be visualized with transient symbolism as in Figure 5A, or with transient symbols, as in Figure 5B where lines of variable hue and width are added temporarily. The advantage of the latter is that the transient symbols can symbolise, or be interrogated for, information about the two zones which they join. For example the difference in their values, or the magnitude of flows between them.

These animated figures illustrate two symbolism techniques for contiguity probing. They provide a complex snapshot showing ten views of a data set where many hundreds might be required when undertaking exploratory analysis. An interactive session would aid legibility as the map viewer could request views which answer particular queries by showing certain aspects of the data. Developing views which reveal unknowns and views which confirm or refute their existence is at the core of visualization (MacEachren, 1994). Tcl/Tk provides an interactive environment where commands can be issued from the keyboard, or through programmed widgets, to change or remove colour symbolism, resize, crop, pan, and interrogate the map for data values, to link it to other spatial or statistical views, and to show contiguity for single lags, or selected pairs of zones, and switch contiguity symbolism on and off as required (Dykes, 1996b).

EXPLORING STATISTICAL REPRESENTATIONS

Both transient symbolism and transient symbols can also be used to provide analysts with information about attribute distributions and the statistical techniques used to measure them. Here an enumerated data set comprising the counties of Wisconsin is used for illustrative purposes. The data show median household income for 1990 and were obtained from the U.S. Census Bureau.

Statistical distribution within a locality

Here the contiguity probe has been extended to show central tendency and dispersion through standardised local z-score and local difference statistics. Each time a polygon is entered a procedure determines the neighbours and calculates a local mean. Neighbouring polygons are mapped with red or blue hues depending on whether they are above or below the local mean. The lightness and saturation of hue are varied to produce a colour scale which reflects the strength of variation about the local mean in terms of the local standard deviation. Darker, more saturated hues represent the extremes. Yellow polygons denote values that are within 0.25 standard deviations of the mean. The probe shows changes within a locality that might not be detectable when using a globally defined colour scheme such as one based on intensity. The colour scheme is effectively varied depending on the data set, a technique advocated by MacEachren (1995) for enhancing ranges where contrast is required to explore pattern. Here the "ranges" are regional, defined by a locality.

The local difference statistic shows the difference between the value of the interrogated polygons and their neighbours, again in terms of the local variation. This is a good tool for detecting local form as a county surrounded by blue indicates a local high, whereas a county surrounded by red indicates a local low. Troughs, slopes, and ridges can all be identified.

A lagged contiguity matrix allows measurements to be visualized at a range of scales. In Figure 6B a central county is selected, and the immediate neighbourhood shows a north-east to south-west "slope" of medium household income. As the scale at which the statistics are measured is increased the detected pattern reverts direction, revealing a more general trend from south-west to north-east, showing the dominance of Milwaukee and its suburbs on the distribution of mean household incomes.

Visualizing spatial autocorrelation

A spatial indicator of variation can be calculated from standardised local values. Perhaps the best known index of spatial autocorrelation amongst geographers is Moran's I (Cliff and Ord, 1973; Getis and Ord, 1992). The statistic computes a measure of spatial autocorrelation based on the spatial distribution of variation around the mean. When measured globally the index may fail to detect non-stationarity in data sets where several regimes of spatial association might be present across a region (Hansen, 1996). Capabilities for interactive visualization, such as those demonstrated here, have generated a propensity for using techniques that explore these local patterns of association in spatial data (Anselin, 1995). Regional variation in spatial autocorrelation can be detected graphically by extending the contiguity probe to produce transient symbolism which reveals Moran's I for some local region defined by the probe position.

Figure 7 shows an autocorrelation probe in use. When calculating a single statistic the contribution of individual zones and sensitivity of the statistic's value to their inclusion in the locality and/or the chosen weight matrix are often overlooked. In 77 an additional view has been produced with transient symbolism to show the covariance matrix for each probed locality.
Figure 6. A. Contiguity probe original map, local z-score, local difference statistic. B. Lagged contiguity probe for single zone original map, local z-score at variety of spatial
Figure 7. A. Probing for covariance matrix. B. Interrogating covariance matrix. C. Interrogating covariance matrix at spatial lag of 2 zones.
Moran’s I is calculated in Figure 7A at the same locations as the local z-score was calculated in 66A. Moran’s I is a scaled product of local covariance, multiplied by a weight factor. If the distance weight factor is a binary indicator of contiguity, I is proportional to the covariance, the sum of the product of the variation about the mean for all neighbours. So neighbours that are the same side of the local mean have a positive effect on I and neighbours that traverse the mean have a negative effect. The size of the effect is dependent upon the magnitude of the variation.

The coloured map shows the local z-scores, and the view to the right of the map shows the covariance matrix. Each row and column represents a zone in the defined locality, and where two zones are contiguous the corresponding square in the matrix is coloured. Red represents a positive value, blue a negative one. Again the saturation and lightness of hue symbolise magnitude. Each column is summed to produce a figure of “local influence”, which is symbolised by the coloured bar above each column. For the central zone, which is compared with all of it’s neighbours, this is Anselin’s local Moran’s I (Anselin, 1995). The sum of the columns (and so the sum of the covariance matrix) represents Moran’s I for the selected region, the figure for which is displayed above the matrix.

Interrogation of the graphic provides information about the statistical representation. Figure 7B shows how a column can be clicked to show the zone which contributed the value, and the zone on the map and the constituent symbols on the covariance matrix are highlighted. Any symbol on the matrix can be interrogated too, and the two zones which contributed the value are revealed. Thus the dependence of a statistic upon a small number of values, it’s stability, and an idea of what it represents and why, can be determined.

A final figure in this section shows that the autocorrelation graphic can also be extended to measure at a variety of scales based upon spatial lags. 77C shows how second-order neighbours can produce a positive value for Moran’s I when the first-order figure is negative. The contiguity matrix is interrogated and a pattern emerges where a single zone has a dominant effect. There are authentic geographical reasons for this. Menominee County, the dominant zone which is probed in Figure 7C, has an extremely low median household income. The unique multivariate attribute behaviour of Menominee is a result of the American Indian reservation that lies within the county.

Distance measurement for weighting

The spatial statistics contemplated thus far have used binary inclusion rather than distance weightings which reflect the spatial significance of individual zones in spatial statistics, alternative dynamic techniques become useful.

A large number of parameters must be considered when calculating a local statistic. These include the distance function, centroid definition, and location at which the statistic is measured. Each can have a major impact upon the index calculated, but is essentially arbitrary in selection. An approach that recognises this variation in potential outcomes is one where the stability of the index is determined when parameters are changed. This idea is analogous to Muehrcke’s concept of map stability (Muehrcke, 1990) where confidence in representations is assessed by examining the extent to which small changes in the method of map creation affect the visual product.

Transient symbolism can be used to illustrate the distance weights of zones, the effects of changing the location of a point which represents an area, and the location from which a statistic is measured. The resulting tools permit experimentation with multiple statistic definitions, and provide information about the sensitivity, stability, and validity of measurements made from maps.

Visualizing distance weighting between points. The example boundaries shown in Figure 8A are counties in Illinois. Tel Tk code added to “cdv” provides the ability to plot routes with the cursor. Localities can then be defined by positioning a brush at each location on the route. This process is known as tracing. Brushes can be circular (basing localities upon distance inclusion) or irregularly shaped.

Statistics can then be derived for the computed locality with weightings based on a distance function for each point on the trace in turn. The function can be keyed in and interpreted by a Tel Tk string parser.

The process is graphic with distances, local zones, and centroids symbolised. Figure 8A shows the results of changing the distance weighting function. Centroids are shown in light blue, and local zones are highlighted with green outlines. A red hue is used to colour zones contained in the locality, with the saturation of the hue dependent upon the strength of the distance function for each zone.

In the first example a 1/d function is symbolised. The two zones with centroids near to the sample point (indicated by the black cross) are dominant. This effect is enhanced with the 1/d² function used in the second example. The third subtracts the distance to each centroid from the maximum distance to the furthest centroid (which therefore has zero weighting), and shows less skewed distribution weights. This alternative produces a local statistic with more regional input.

Attribute location sensitivity. A second use of dynamic graphics is in examining the sensitivity of attribute location. The first image in Figure 8B
Exploring spatial data representation

shows the weightings associated with a $1/d^2$ function. In this example centroids have been calculated automatically as the average of all of the x and y points in the zone. The two zones closest to the sample point have a meandering common edge which is digitised with far more points than the opposite edge. In the second image the two centroids have been dragged interactively to locations closer to the “centre of gravity” of the zone. The same weight function has been used, but the different strengths of red show the effect of an alternative centroid definition algorithm, and reveal the sensitivity of a distance weight based statistic.

This interactive movement can be used to researchers’ advantage, as centroids can be placed at estimated centres of population. In the third example shown in Figure 8B the polygons have been reconfigured to appear transparent and an image item has been used to display a geo-referenced map. The centres of population can be viewed in each zone. In the fourth example the dark-blue crosses show that the centroids have been dragged to new locations where centres of population have been identified. The final image shows how the same $1/d^2$ function weights the attribute values of each zone with the centroid distribution as refined through comparison with the scanned map.

Variation with measurement location. In this example the sensitivity of the location at which a statistic is measured is visualized. Figure 8C shows a trace, digitised interactively with the mouse, about the original point of measurement. The distance weighting function is again $1/d^2$ and the adjusted centroids are used throughout. As the trace moves (shown by the black “X”) the locality, defined by a distance about the point of measurement, varies and the weights attributed to the various neighbouring zones are shown by the saturation of the red hues.

Whereas a moving statistic should be expected to alter, the seemingly violent differences in attribute weighting produced by the $1/d^2$ function with little change in location demonstrate that a traced index, which may be located on-screen with a mouse as it was here, or interpolated between two specified points, should be employed with due care and attention, and certainly with consideration of its sensitivity to change. Dynamic graphics such as those shown here can provide analysts with an indication of the validity and stability of a statistical representation.

**CONCLUDING COMMENTS**

This paper and its accompanying dynamic illustrations have attempted to do several things. They have outlined the need for an open, flexible environment for exploratory spatial data analysis where
Figure 8B
Figure 8. A. Visualizing alternative distance weight functions. B. Alternative attribute locations. C. Variation in location at which statistic is measured.
maps behave in response to user requests. A dynamic environment for visualizing in such an interactive way has been introduced with simple examples and data sets, and applications have illustrated techniques which use such map behaviour to visually explore cartographic and statistical representations of spatial data.

A concluding graphic integrates many of the features of the examples and demonstrates that multivariate views can be combined with abstract spatial views in a dynamic, linked, environment for visualization in which the observer controls the temporally changing symbolism to extract information about the data. A dynamic parallel coordinates plot (Wegman, 1990) which shows the relationships between all twelve example variables has been added to the previously constructed views in Example 9. "cdv" includes a more powerful dynamic parallel coordinates plot in which axes can be re-ordered and pairs of variables visualized as transient linked scatter plots. The views contain the spatial smartness demonstrated in Example 8. Clicking mouse button 2 on any symbol will produce transient lines and shading which reveal contiguous zones in the polygon and scatter plot views. These symbolise attribute differences, where the arrow and hue show direction, and line width represents magnitude. In addition the example provides a method for visualizing the effects of altering distance weights parameters across the British regions, as outlined in the previous section. This is achieved by double-clicking the polygon map to set a location, holding the Shift key and clicking the "X" symbol which identifies the location. Symbols will be filled with a red hue, and saturated according to the relative distance weight of each zone in all views. The coloured lines on the parallel-coordinates plot therefore provide distance-weighted multivariate information on the "locality". The equation used to calculate the distance weights can be changed by typing in the text entry box and pressing return. The effects of a new equation can be visualized by holding the Shift key and clicking the "X" symbol once more. The equation parser will interpret the "+, *, -", and operators and the following functions and variables:

\[ d \]
\[ \max \]
\[ \text{pow}(d,n) \]

So legitimate distance weight functions include:

\[ \frac{1}{\text{pow}(d,2)} \]
\[ \max-d \]
Exploring spatial data representation

Whereas the examples show that the available techniques are powerful, the proposed environment does have restrictions. For example, speed is greatly reduced as symbol numbers increase. One solution is to use a Tcl/Tk compiler and run binary code rather than interpreted scripts, but users then lose the openness, flexibility, and interaction provided by a real-time command line. Additionally, the available symbols do not match exactly with those that a cartographer might require. For example, items cannot easily display crispness, resolution, or transparency and no canvas commands are available to change the orientation of symbols. A make-shift solution is to use bitmap or image items showing symbolism, but much of the interactive nature of the items and ease of reconfiguring is lost. More importantly no genuine georeferencing exists. All coordinate calculations and locating of symbols are accomplished by the visualizer.

Potential solutions exist to many of these problems in that Tcl/Tk source code is distributed, and extension of the toolkit’s capabilities is both encouraged and frequent amongst the user community. Programmers can thus develop additional cartographic items and create georeferenced canvases for spatial views.

Although it is becoming increasingly recognised that interactive environments, which encourage users to experiment with combinations of data and graphic symbols, are appropriate for visualization tasks (for example, DiBiase and others, 1994), the knowledge base of suitable representation techniques for visualization is modest. MacEachren (1995) suggested that the “optimal” multivariate analysis tools might restrict the selection of particular representations to those deemed appropriate by tool designers, but leaving specific choices with which the analyst can experiment. This concurs with Lindholm and Sarjakoski’s notion of a personal cartography where users see data on their own terms (Lindholm and Sarjakoski, 1994). We know currently so little about representations for exploratory analysis that no empirical assessment has been made on one of the earliest forms of cartographic re-expression, the effects of interaction with class delimitation, on the ability of expert analysts to uncover patterns (MacEachren, 1995). This is the situation, even though the technology has been available to produce such representations since at least 1992 (Egbert and Slocum, 1992). It seems appropriate at this stage to provide analysts with the tools or environments to create multiple personal cartographies.

Hansen (1996) demonstrates that proprietary GIS are providing increasing flexibility of the sort demonstrated here, and now use object-based symbolism. Whereas it could be argued that there are some advantages in using proprietary GIS for visualization, the flexibility and control available when using a graphical toolkit make the approach more appropriate for development work. This notion is supported by the range of examples presented here, and by considering the “potential enhancements” wish list for visualizing enumerated data sets provided by Egbert and Slocum (1992) when reporting on EXPLOREMAP, and noting that the majority of these are eminently feasible with the graphic tools afforded by Tcl/Tk.

This paper has aimed to demonstrate that the Tcl/Tk graphical scripting environment is at an appropriate programming level for spatial analysts to produce dynamic two-dimensional maps for visualization where “wish lists” will change between individuals and throughout the research process, and for prototyping and investigating the utility of dynamic representations of data which exhibit user-related behaviour.

Acknowledgements—The tourist data were kindly provided by Wim van der Knaap and colleagues at the Centre for Recreation and Tourism Studies, University of Wageningen, The Netherlands. The British regions data set is based on data provided with the support of the ESRC and JISC and uses boundary material which is copyright of the Crown and the Post Office. The attribute data are entirely hypothetical and were composed specifically for these illustrations. They have no authenticity or bearing on reality whatsoever. The U.S.A. socio-economic data were acquired from the U.S. Bureau of Census WWW Server. The employment composition icon was the idea of Matt Duckham, and is reproduced with his permission. Much of this work was undertaken through the U.K. HEFC Joint Information Systems Committee, New Technologies Initiative, funding of Project Argus.

REFERENCES


DiBiase, D., Reeves, C., MacEachren, A. M., von Wys, M., Krygier, J. B., Sloan, J. L., and Detweiler, M. C.,


APPENDIX

Related URLs: (7 May 1997)

Tcl/Tk Homepage
Tcl Plug-In
Tcl Plug-In Feature
Tcl Examples - Command Interpreter
U.S. Census Bureau 1990 Census Lookup
cdv Information
Jason Dykes, Tcl/Tk Examples
Cartwright, W., 1997.

http://www.sunlabs.com/research/tcl/
http://www.sunlabs.com/research.tcl.plugin/
http://www.sun.com/960710/cover.index.html
http://www.sunlabs.com/research/tcl.plugin/eval.html
http://www.aw.com/cp/Oust.html
http://venus.census.gov/cdrom/lookup/
http://www.geog.le.ac.uk/argus/
http://www.geog.le.ac.uk/jad7/Tcl/examples/
http://www.srl.rmit.edu.au/icawgl/viz_96_1/
http://www.geog.le.ac.uk/jad7/cdv/
Cartographic visualization: exploratory spatial data analysis with local indicators of spatial association using Tcl/Tk and cdv

Jason Dykest
University of Leicester, UK

[Received June 1996. Revised January 1998]

Summary. cdv, a cartographic data visualizer, is a piece of demonstration software designed for research and teaching. It uses established symbolism techniques for displaying spatially referenced information, and adds newer dynamic capabilities that are suitable for exploratory spatial data analysis (ESDA). The software takes advantage of the symbolism, dynamism and inherent spatial information afforded by the Tcl/Tk language and comprises interpreted scripts which provide a flexible environment for prototyping and extension. This introduction to the software and approach outlines some of the rationale behind the development and contains an overview of the cartographic and statistical ESDA functionality. The distributed scripts produce concurrent linked views of enumerated data sets. Views are available for statistical and geographic visualization of univariate, bivariate and multivariate distributions. These include dotplots, scatterplots, polygon maps, population cartograms and parallel co-ordinates plots. Each view is extremely dynamic so that transient maps can be produced to highlight specific cases and to focus on points of interest to suit the researcher. The ideas are applied to procedures for determining locality and assessing the characteristics of local variation, and some examples of dynamic graphical local indicators are provided. The examples and images provide a flavour of the approach and the dynamic and flexible nature of the software. One of the advantages of the approach is that, as the software is free and accessible on the Internet, the author is not restricted to static black-and-white graphics for communicating dynamic colourful ideas. Larger colour versions of the images can be downloaded to support the paper.

Keywords. Cartography; Exploratory spatial data analysis; Flexibility; Local indicators of spatial association; Scripting; Visualization

1. Introduction

One of the advantages of this approach is that, as the software is free and accessible on the Internet, the author is not restricted to black-and-white static graphics for communicating colourful dynamic subject-matter. Larger colour versions of the figures, the software and example data and links to all universal resource locators (URLs) referenced can be obtained and used in conjunction with this paper from

http://www.geog.le.ac.uk/jad7/Statistician

Both statisticians and cartographers are aware of the utility of locating symbols on a plane to show the statistical or spatial distributions of one or more variables. Geographic data sets are customarily mapped by cartographers using the two dimensions of the plane to portray spatial
location. Statistical graphics of data sets that are not georeferenced often make use of location to show data values. Cartographers have developed techniques for symbolizing one or more statistical values at locations in space, from which geographers and other spatial scientists can assess distributions. The graphic symbols that are available to the cartographer are most famously outlined by Bertin (1981).

Advances in computing have greatly augmented and enhanced mapping techniques, with numerous results including the following:

(a) New methods for representing (multiple) data values have emerged. These include dynamic symbolism techniques and those for displaying time or attribute series.

(b) The restrictions put on cartographers to use the plane to show locational information have been lifted by dynamic linking and interrogation techniques which relate symbols in non-spatial plots to geographic locations.

(c) The turnaround time for producing maps and graphics has reduced to the point where new maps can be produced almost instantly. Unique transient views of a data set can be produced to answer very specific queries set by a map user in an extremely interactive way.

These factors, and others, have contributed to a confidence in the use of graphics at the exploratory stage of the research process where spatial data sets are viewed and reviewed under changing cartographic conditions in an attempt to elicit patterns and trends which might be present, and to assess their validity (MacEachren, 1994). Software which is intended for fostering ideas and discovery rather than presenting conclusions, by providing dynamic displays which encourage experimentation with different combinations of data and graphic symbols, is termed cartographic visualization software by DiBiase et al. (1994). MacEachren (1994) termed the stage of the analytical process where cartographic visualization is useful as ‘visual thinking’. Such tools which utilize modern mapping techniques, combine geographic and statistical information and present it in a dynamic, interactive and visual manner are attractive for spatial scientists who are interested in assessing multivariate spatial distributions.

2. A flexible approach to cartographic visualization

DiBiase et al. (1994) stated that cartographic visualization software is tailored to a specific set of users with a specific set of uses. Yet the complex multifarious nature of spatiotemporal data and geographic inquiry mean that applications tend to be unique in terms of their combination of data types and the kinds of questions that are appropriate. The statistical and graphical representations that are suitable to permit best the enquiring scientist to perform effective exploratory analysis may also be unique. The continuing development of new techniques and the availability of new media further complicate the situation. In these circumstances the potential rigidity of a tailored interface might not be the most beneficial for the visual thinking stage of the research process when scientists are likely to need to view their data in a variety of ways which may not be apparent at the onset.

An environment for visualization is required which allows users to view and review their data, to select and reselect geographical and statistical subsets, to create and prototype suitable representations and to ask questions and to modify enquiries repeatedly. Such flexibility is likely to require a solution that is becoming increasingly popular, that of the high level scripting language.

The capabilities of the Tcl/Tk graphical user interface (GUI) builder (Ousterhout, 1994) for dynamic cartography have been introduced elsewhere (Dykes, 1996, 1997). The language provides an extremely flexible high level scripting environment and the range of symbolism required by cartographers. Symbols are created with simple, high level commands, and symbolism is changed
in a consistent way for all symbol characteristics. Dynamic properties of the language, such as binding cursor movement and events to symbols so that the commands that change maps can be linked to users' behaviour, equip map makers with a series of new dynamic symbol characteristics. These are particularly appropriate to visual thinking as views can be changed by the user to suit specific conditions, uses and questions.

The Tcl/Tk language is interpreted, rather than compiled (although compilers exist), providing researchers with ultimate control whereby they can modify their maps from a command line, as they need a new angle on the information that they are mapping to answer successive queries of their data. The command line scripting environment permits rapid prototyping and the kind of level of engagement with data that is required for exploratory analysis. New views, that are specific to certain data types or data sets or queries, can be created to take advantage of the symbolism options that are available in a suitable way. This mode of interaction is supported by electronically published papers (e.g. Dykes (1997)) and a series of on-line tutorials (URL 1 in the Web site given in Section 1) which demonstrate the approach. Further support can be obtained from the growing Tcl/Tk user community, many of whom (including the author) are accessible at the click of an electronic mail 'send' button to provide experience and exchange, and to debug and comment on code.

Although DiBiase et al. (1994) recognized that it is difficult to evaluate the success of exploratory methods, this kind of open flexible environment for cartography is highly appropriate and promising. An application that implements this flexible approach to mapping for exploratory spatial analysis is presented here, 'cdv for enumerated data' is a highly interactive piece of software which has been developed using the Tcl/Tk GUI builder as a teaching and research tool to demonstrate some of the dynamic cartographic techniques that might be useful when exploring multivariate, ratio-scaled, enumerated data sets.

3. Symbolism for multivariate exploratory spatial data analysis

cdv can be run in two modes. The 'tailored' mode for demonstration and teaching provides a GUI to techniques which are appropriate to enumerated ratio-scale data sets for certain tasks. Alternatively, by running the scripts from the command line, techniques can be extended and applications developed to suit the particular data set under assessment.

The software loads two text files, one containing enclosed polygons with identifiers, another containing lines comprising an identifier and a name and a line of numeric information. Each space-delimited number in the line relates to a recorded measurement associated with the area defined by the correspondingly identified polygon. The data used here consist of spatial and attribute information collected in the 1990 US census of population for counties of the states collectively known as the 'East North Central Midwest' (see URL 2). They are distributed with the software.

The statistical distribution of single variables can be represented using symbols located along an axis.

Each symbol represents a county and the location along the axis its relative value. A box plot is added to show the median and quartiles. Interactivity is provided by a series of GUI buttons through which the representation can be varied. The view can be resized, as can symbols. The symbols can be made transparent and raised or lowered above or below other symbols to inves-
tigate the density of cases at any location. This shaded dotplot shows the distribution of '% of Asian/Pacific Islanders' using location and intensity ranging from 100% for the minimum value to 0% for the maximum in a linear, non-perceptual scale.

The geographical distribution is shown by colouring corresponding polygons with the same intensity.

Views are transient and highly interactive. Each can be interrogated for the numerical values of selected variables, and all views are dynamically linked so that touching any symbol in a view will show the name of the location and highlight corresponding symbols in all other views. Thus a spatial component is added to views which use symbol locations to represent statistical information about spatial data.

An important aspect of this data set is the huge range of population totals in each of the counties, from a minimum of under 2000 to a maximum of over 5 million! American enumeration zones tend towards equal geographic area, rather than their British equivalents for which cdv was developed. These are used for electoral purposes and so tend towards equal population. Cartographers have developed a series of techniques for displaying population information in addition to data values with symbols that use size to represent population and colour characteristics for attributes. Here a precomputed population cartogram displays the same distribution.

Dorling’s (1993) algorithm has been used to create a non-continuous cartogram of the five states with circle symbols sized to represent population totals in each county. The program runs an iterative procedure which moves overlapping symbols apart, while drawing neighbouring cases together. In doing so it attempts to retain the contiguity structure of the geographic space. Linking corresponding cases in all views demonstrates that the conventional geographic representation under-represents densely populated urban zones.

A familiar set of techniques for selecting cases allows spatial or statistical groups to be highlighted in a range of colours, enabling classed choropleths, cartograms, dotplots or scatter-plots to be created with colour schemes that are appropriate to the data (Brewer, 1994). Selection
can be used for temporary highlighting or to produce views relating to subsets of the data. Highlighting reduces the oft-quoted difficulties in relating cartogram symbols with the places that they represent. These images illustrate that case selection can be accomplished in any view, and the selected zones highlighted in all other included views.

Firstly a geographic region is selected and the equivalent zones shown in the abstract cartogram space (top pair). Then a region of the cartogram is chosen and displayed on the polygon map (bottom pair).

By creating views of subsets of the data the whole range of graphic symbolism can be applied to a restricted data range and local variation can be assessed using the full extent of perceptible visual cues. Here ‘% poverty status determined’ is mapped and the distribution shown for the full data set (left and top). A locality is then selected for further assessment, with the grey intensities and dotplot locations of subsequent views restricted to the local data range (bottom right). Groupings on statistical views could equally be selected.

Variables can be combined for analysis in cdv in several ways. The calculation function provides an equation parser so that combinations of variables can be created by using Tcl’s mathematical functions. Newly created variables are saved for later use. Additionally univariate views, such as the dotplot or the population cartogram, can be shaded to show a second variable. For example, the symbols on a dotplot showing the number of people in each zone, or a population cartogram, might be shaded with intensities showing percentage unemployment. Overlapping symbols can reduce the utility of such a plot. An alternative was scripted for the Midwest data set whereby zone sizes are used to symbolize population totals on the dotplot.
This example showing '% below poverty level' allows the distribution of numbers of people living below the poverty level to be assessed. It illustrates the wide range of population totals in the dataset and emphasizes people as well as cases.

Pairs of attributes can be displayed using locations along orthogonal axes with the scatterplot view which takes advantage of all the dynamic features of the dotplot. In addition, a 'Ratio' button toggles between axes which show values by using ranges that are individual or common to both variables and a 'Swap' button switches axes. Box plots are added to each axis so that the distribution of values can be assessed, and parallel box plots on the y-axis allow the two distributions to be compared.

Three variables can be assessed concurrently by using a variety of techniques. A third variable dimension can be added to a bivariate view by shading symbols with an additional attribute. Alternatively each of the three primary hues can be used to distinguish between three variables and an intensity applied to each symbol dependent on the magnitude of the data value for the associated county. A red–green–blue colour composite results. Greys (in a range from black (low values) to white (high values)) indicate low relative variation between the three variables. Zones coloured in any of the three primary hues indicate extreme values in the variable represented by the hue. Other colour combinations can be assessed by using a colour key.

The composites provide a useful visual means of identifying alike zones. Plotting the colours in a cartogram representation gives a spatial indication of the numbers involved in each multivariate group and allows counties of similar population total and attributes to be identified.

Here percentages of those 25 years of age or over with a high school diploma, with any college
degree and with a higher or professional degree are combined using red, green and blue for the respective variables. The multivariate composition of localities can be assessed as demonstrated above, by selecting a spatial subset and using colour variation over the local range.

Parallel dotplots provide the key to viewing more than three variables. Producing a number of dotplots in cdv and using the dynamic links to view the distributions of cases allows multivariate distributions to be assessed.

These views show the same information as the red–green–blue composites (above) but symbolize with linked locations rather than three elements of colour. A single case representing Monroe County is highlighted in each view. If symbols representing the same case in multivariate parallel dotplots are joined, a parallel co-ordinates plot (Wegman, 1990) is produced. This provides a multidimensional signature for any case. Outliers, clusters, multiple modes and other qualities of multidimensional data sets can be determined. With experience users can interpret subtle multidimensional patterns from the geometry of such plots. cdv provides a number of means of aiding interpretation.

Parallel co-ordinates plots only show the relationships between parallel axes, and cdv orders these by calculating Pearson's product moment coefficients and plotting the two most correlated axes, followed successively by the next strongest relationship. The two most correlated variables are displayed with orthogonal axes as a conventional (but dynamically linked) scatterplot. This produces an ordered combination of n axes from n! possible combinations. Any pair of parallel axes can be displayed as a scatterplot by clicking the axis with the right-hand mouse button, and axes can be moved interactively by dragging them, allowing relationships between any two variables and any of the n! combinations of axis orders to be viewed.

The above example is a dynamic parallel co-ordinates plot for a selected subset of the Midwest data set. It shows (from left to right) the three variables for which the colour composite maps and parallel dotplots were produced above, and then the percentage for whom poverty status has been determined, and the percentage living below the poverty level as recorded by the US census of 1990. Lines are highlighted, and shown in all other selected plots, by touching them with the cursor. Here, the highlighted line represents Monroe County (notice that the straightness of the line across the first three axes mirrors the aligned distribution of the highlighted symbols on the parallel dotplots shown above). It displays relatively high values for the first three variables, a very low value for those with poverty status determined and a remarkably high figure for the percentage
living below the poverty level. This contradicts the expected positive relationship between the final two variables which can be seen to exist in most other cases. Geographic questions result, and additional information and views are required to answer them.

4. Visualizing local statistics

cdv was developed as a tool for dynamic graphics rather than for statistics, but for assessing local variability within distributions the concept of transient symbolism can be extended to include the production of temporary local statistics, calculated when users interrogate locations within a view. Localities and indicators can be calculated and visualized in a transient and dynamic fashion for exploration and synthesis (MacEachren, 1994).

Tcl/Tk symbols implicitly contain information about their locations and extent, and so commands can be issued which return the bounding box of any symbol and the identities of those symbols that overlap, or are enclosed by, any box. This means that polygons can return a list of likely neighbours, which is examined to find first-order contiguity. The excellent list handling capabilities of Tcl/Tk are appropriate for the Anselin and Smirnov (1996) technique for calculating lists of neighbours at higher order lags. Once neighbourhood lists are computed zones can be probed for neighbours at a variety of spatial lags.

www.geog.le.ac.uk/jad7/Statistician/ Figures/Contiguity.html

Here lags of a single zone (top pair), and from one to five zones (bottom pair) show the ‘contiguity probe’ being used in two linked views. The graphics change rapidly as the cursor is moved to show the currently defined neighbourhood.

In cdv the local variation of a single attribute can be visualized with the probe in two ways: either by colouring symbols within the defined neighbourhood depending on the distribution relative to a local mean (effectively showing local z-scores) or relative to the value of the probed symbol (thus showing local variation from the central zone). As with regional selection the symbolism is applied to the local data range and so local patterns are visually emphasized. cdv uses red and blue hues for values above and below the mean or central value and varies colour saturation dependent on the magnitude of the variation. So the local variation probe identifies peaks, pits and ridges with blue, red and ramped colours respectively. Each probe can be used at a variety of spatial lags.

Local z-scores can be combined with distance information to determine a local value of
Moran's autocorrelation index. cdv calculates the index and visualizes the covariance matrix whenever a symbol is touched with the cursor. The matrix reveals the significance of each pair of values that contribute to the index. Pairs of zones, represented by cells in the contiguity matrix, are coloured in the scheme outlined above. Bars above each matrix column show the total contribution of that zone to the index. For the central zone this value is equivalent to Anselin's (1995) local indicator of spatial association.

www.geog.le.ac.uk/jad7/statistician/ Figures/Autocorrelation.html

The graphics are linked and can be interrogated. In the top image here a bar symbol above the matrix is touched to highlight the component cells in the matrix and the zone that it represents in the base map. The central zone can be seen to have five neighbours, three of which (the blue cells) produce a negative value when their variation from the local mean is combined with that of the central cell. Any cell within the covariance matrix can also be interrogated to show the pair of zones that it depicts. In the second image a very negative (deep blue) cell highlights the two most extreme values among neighbours. The covariance matrix visualization can be extended to include an interzone weight function by using cell size to represent the proximity of each pair of zones. cdv provides an equation interface where the weight function can be varied, and the effects of different assumptions about local statistics weighting assessed. The third and fourth images in this series show weight functions of $1/d$ and $1/d^2$ respectively.

www.geog.le.ac.uk/jad7/statistician/ Figures/Autocorrelation.html

As the weight function becomes increasingly significant the pairs of zones that have closest centroids dominate the index, as shown by relative sizes of the squares in the covariance matrix,
and the changing lengths of the bar symbols, in the lower images. The effect of the \( 1/d \)-function is to increase the value of the local indicator, as the two zones with the highest values are closest. The \( 1/d^2 \)-function further exaggerates this effect until the previously measured negative autocorrelation is eliminated. The graphical representations of the statistic allow its components and parameters to be assessed interactively and the sensitivity of the index, and utility of the measurement, to be observed in a manner mirroring that outlined by MacEachren (1994) with regard to spatial, rather than statistical, graphics. Such analytical methods can be extended to visualize the effects of centroid repositioning from, say, geometric locations to those based on population centres (Dykes, 1997).

5. Local multivariate views

The flexibility offered by the Tcl/Tk programming environment means that new views and ways of interacting with spatial data can be prototyped and tested. Local indicators consist of a combination of locational information with multivariate attribute data. The need to investigate ‘traces’, where local statistics can be measured interactively, much like the contiguity probe, but along a route and with a neighbourhood defined by the user, was identified following meetings between Project Argus researchers and statisticians based at the Institute for Mathematics, University of Augsburg. It was suggested that such traces could be used to investigate local variations in multivariate information along linear geographic features. The following examples demonstrate scripts added to cdv that enabled some of the proposed ideas to be investigated and techniques to be assessed. They use the state of Illinois for clarity.

A GUI widget was added to cdv to control tracing. Points at which local information is required are digitized interactively on the base map and linked with a series of arrows. Delimiting the neighbourhood around each sample point can be achieved in several ways. Distance inclusion is one possibility, where distances are drawn interactively on the map and any zone with a centroid within the specified distance of a point on the trace is included in the locality. Shaped brushes are another possibility and are defined with a ‘rubber band’ box, or by drawing an enclosed polygon. Brush-based selection can be set to include all zones that overlap the brush or are totally enclosed by it. The brushes can remain at the orientation at which they are drawn or be rotated to retain their relative orientation to the direction of the trace (if for example a rectangular buffer around a linear geographic feature is being investigated).

When localities are defined the software highlights the zones included and adds crosses to identify their centroids. Once zones have been selected to create a locality, information about the locality can be calculated or visualized. An additional GUI is used to define a distance weighting for any local analysis. The magnitude of the distance weighting coefficient for each zone is then depicted with saturation of hue. The relative effects of varying the distance coefficient, the centroid locations, and the point at which a sample is taken, can then be assessed (Dykes, 1997).
Here opposite, a distance inclusion technique is used to define a locality, along a trace shown by the dark line. The crosses and zone shades show the centroids and distance-based weights of the zones respectively. The dark square shows that a Tcl/Tk bounding box around the circle symbol depicting the selected distance is used to reduce the search for included zones to those overlapping that area. The potential efficiency savings of such a spatially restricted search are huge. Distances between the trace location and centroids for each overlapping zone are calculated and those within the specified distance included. The weight function is then used to assign a weight and colour saturation to each zone ($1/d^2$ here).

Once zones have been selected interactively, a variety of statistics can be computed. For exploration and in accord with the graphic nature of the analytical environment, prototype multivariate graphics are produced here. Multivariate local graphics embrace the concepts of cartographic symbolism, that the information processing capabilities of the eye–brain combination can be used successfully to analyse and assess complex geographic information, and apply them to interactive probes. These examples take the form of local parallel co-ordinates plots, which at their simplest (top plot) show the relationship between a number of chosen variables within the locality (shown on the map).

The plots show the three educational attainment variables and percentages with a high school diploma, college degree and graduate or professional qualifications. The variation in the selected attributes and relationships within the locality can be assessed. A multivariate mode and local outliers are visible here. The graphics contain all the dynamic and linking properties of the global parallel co-ordinates plots and also incorporate circle symbols which show zone population totals. The local view updates to plot the information relating to each new locality as the sample location moves along the path of the trace. Various visualization options exist to reveal more about the local distribution. The colouring applied to the zones included in the map view, based on the distance function, can be used to show the proximity of the zones symbolized in the parallel plot.
(second image) to the centre of the location defined by the trace. Closer zones dominate and the zone with heaviest weighting can be seen to follow the local mode. The third plot uses $1/d^2$, weighting the graphical indicator towards the case closest to the sample point. As the sample location moves along the trace lines fade in and out of the locality and the graphic. The result is a multivariate, distance-weighted, local, dynamic graphic. In the fourth image the sample has moved along the trace. The graphic shows that the zone with the greatest weighting is a multivariate outlier containing most of the population within the locality, suggesting a locality with a distinct geographic core and a periphery with comparable multivariate characteristics.

6. Assessment and outlook

cdv demonstrates a tailored application of Tcl/Tk dynamic symbolism to provide an interactive visual interface to a particular type of data set. It has varying degrees of suitability for similar data sets, depending on the questions that need to be asked. Additional functionality can be rapidly scripted to suit other data sets, to respond to their intricacies and to prototype techniques, as shown in the previous section. As a demonstrator of methods and approach cdv has been successful. The Knowledge-based Interfaces for National Data Sets (KINDS) Project (URL 3) at the University of Manchester has attracted funding to appoint a National Census Visualization Officer who is running courses using the software to publicize the ideas to the academy and commercial concerns. The software is being supplied on line as part of the Manchester Information Datasets and Associated Services service and supported through the provision of a series of generalized boundaries and population cartograms for use by the academic community. The universities of Leeds, Keele and Leicester are incorporating the software into their undergraduate geography programs.

The Tcl/Tk language provides additional attractions for exploratory spatial data analysis as well. Although advances in the world of computing are turbulent and change is difficult to predict, the current status of Tcl/Tk is that it is free for academic use, and Sun Microsystems appear to be promoting, supporting and developing the product (URL 4). New releases have added useful features for visualizing spatial information and speed up the runtime of applications, meaning that larger, more complex data sets can be incorporated. Developments include a World Wide Web browser facility which allows applications to run remotely over the Internet (URL 5) and hypertext support which permits Tcl/Tk maps to include links to hypermedia resources across the globe as described by Cartwright (1997). Links between Java and Tcl/Tk are being developed, under the suggestion that, whereas the former is best suited to doing calculations, interfaces are most easily created by using the latter. The concept of cdv as an analytical spatial interface to geographic data is accurate. Proposals to incorporate the techniques within KINDS as a 'visualization gateway' to the UK census data stored at the University of Manchester will take the analogy further.

The combination of the wide range of symbolism possibilities offered by Tcl/Tk symbols, their dynamism and inherent spatial information mean that the environment is profitable for computing local indicators and extending graphical symbolism to multivariate visualization of such information. The cdv software and example data sets are freely available on the Internet (URLs 6 and 7), and feedback is encouraged.

Acknowledgements

Much of this work was undertaken by Project Argus (URL 6), funded by the UK Joint Information Systems Committee New Technologies Initiative, along with colleagues Kath Stynes, Jo Wood, Peter Fisher and David Unwin to whom I am indebted.
The local statistics section resulted from collaboration between Project Argus and colleagues at the Institute for Mathematics, University of Augsburg, Germany, funded by the British Council Anglo-German Research Fund. Ideas and comments contributed by Antony Unwin, Adi Wilhelm, Martin Theus, Robert Steck, Heike Hofmann and Meike Sander are gratefully acknowledged.

Jackie Carter, of Manchester Computing, assisted greatly with the release of cdv 1.1.

References


Research Article

Virtual environments for student fieldwork using networked components*

JASON DYKES, KATE MOORE and JO WOOD
Department of Geography, University of Leicester, LE1 7RH, England, UK
e-mail: {jad7, mek, jwo}@le.ac.uk

Abstract. The topics of virtual environments and information technology are addressed here in the context of fieldwork teaching. A pedagogic rationale is presented that considers the objectives of fieldwork and outlines the potential utility of networked graphical tools in this mode of teaching that is so essential to a range of subjects. The Virtual Field Course project (VFC) is developing software that realises this potential by fulfilling a stated series of objectives. A software architecture and implementation are presented that enable visualization software to communicate with a secure and remote shared library of spatially referenced data to support field-based activity. The software is extremely flexible and uses equipment, data and resources that are affordable and accessible to the higher education community. Two example applications are provided. Each uses the described architecture to visualize geographic information in support of fieldwork. Empirical feedback is provided gained from experience of using the software in the field.

1. Introduction

Fieldwork is a multidisciplinary exercise that takes an important place in the curriculum of subjects such as geology, biology, geography, planning and architecture. It offers substantive experience of the world outside of the classroom and is often residential, providing a unique intensive educational experience where many aspects of a region can be analysed first hand. The 1995 UK Geography Teaching Quality Assessment (HEFCE 1995) recognised that fieldwork adds 'an important dimension to the curriculum'. In the field, students can gain proficiency in a number of domains including the development of observational skills, learning about the environment from direct experience of it, and designing, conducting and reporting upon practical projects. These can be achieved through popular teaching modes such as excursions with instructors who identify items of interest and group work where students participate in the collection, integration and analysis of data addressing a specific issue. Fieldwork occurs on a grand scale, for example, Departments of Geography alone in the UK offer 144 degree courses that produce 2500 graduates per annum. Students on such courses typically undertake two residential excursions.

The Virtual Field Course (VFC) Project is an inter-departmental effort being undertaken to address the use of virtual environments and information technology in
teaching fieldwork. Teachers and researchers at the University of Leicester, Birkbeck College and Oxford Brookes University in the UK are approaching the subject through a number of channels (VFC 1998a). An initial survey and assessment of the aims and objectives of fieldwork (Williams et al. 1997) and evaluation of fieldwork destinations (VFC 1998b) were undertaken. Subsequently generic software functionality was identified to support the achievement of the reported aims and objectives and prototype software developed. Here some of the experiences gained from developing and using the prototypes in the field with students and teachers are described.

2. Educational rationale

Kent et al. (1997) identify four stages of fieldwork activity. These are:

1. 'Preparation and briefing' when students are made aware of the academic context in which the fieldwork will take place along with the objectives and the logistics. Students might also be sensitised to the fieldwork environment.
2. 'Engagement in the activity' when students are usually required to determine a suitable procedure for collecting information in the field and undertake some form of data gathering.
3. 'Processing the results of the activity' when the collected data are combined with other information and analysed in context.
4. 'Debriefing and feedback' when a review of the theoretical background and student experience in the field are combined.

Various accounts exist of using information technology in each of these stages. Digital media incorporated into preparation and briefing can inform students about logistical and safety issues, and also reduce the 'novelty space' that can hinder the student fieldwork experience (Warburton and Higgitt 1997). A multitude of hypertext/WWW based 'field trips' exist to fulfil this role (see Ritter 1998). The engagement stage might involve identifying measurement sites for student sampling at the start of a day's fieldwork: Gardiner and Unwin (1986) compute a stratified random technique while Warburton and Higgitt (1997) employ GIS to identify suitable locations. The processing of results can employ statistical (Gardiner and Unwin 1986) or GIS (Warburton and Higgitt 1997) software and communications technology (Williams et al. 1997), all of which can be equally useful media for debriefing, feedback and presentation.

A number of fieldwork teachers were surveyed to ascertain the desired educational aims and objectives of a field course (Williams et al. 1997). 'Comparison' was distinguishable as a key objective, both between geographical locations, models (including maps) and reality and student-collected data and secondary sources. 'Observation', or student centred interpretation, was another. In light of these results an assessment of current and likely future technology led to the identification of desirable software functionality with a particular focus on virtual environments (Moore 1997) which compliment both the teaching and spatial aspects of fieldwork. Virtual reality (VR) provides the unique characteristics of autonomy, presence and place (Zeltzer 1992). Educators can take advantage of these to extend learning techniques and to improve student's understanding and performance (Winn 1997). Consequently the rate of uptake of this emergent technology is rapid (Emerson and Revere 1997). VR is particularly applicable for education in the spatial sciences. The very spatiality of VR can convey a greater cognition of place (Boyd-Davis et al. 1996). The potential use of digital collection and logging devices during the activity stage was also considered and so the software takes advantage of developments in this area.
The process of academic learning is described by Laurillard (1997) through a 'conversational framework' containing discursive, adaptive, interactive and reflective types of learning activity. Ideas are only fully learned by students experiencing them through interactive operation on the real world, reflecting upon this experience to inform their communication with the teacher, and using this dialogue to adapt the way in which they operate on the world. The approach is particularly appropriate in fieldwork where the student-centred project is popular with the teacher acting as facilitator to guide the student's ideas and help them formulate their methodology, analysis and interpretation. This framework rationalises the requirement for student observation and comparison in teaching outlined above and goes a long way to explaining why the field course is such an intensive and valuable educational experience. Computer technology can be used to provide students with such a means for interacting with models of reality and providing adaptive feedback (Laurillard 1997). The degree of interaction can also be controlled dependent upon function or the users skill level.

This approach is also analogous to the iterative 'visual thinking' model of the process of knowledge acquisition identified by MacEachren (1994). Indeed 'visualization' is a key theme in the VFC. Visual methods add an engaging element of realism (Moore 1997), provide a formal model in familiar form (Laurillard 1997) and supply an immediate integrated spatial view of a collected data set for analysis. Just as computer technology has increased the speed at which human-map interaction can occur, resulting in highly interactive 'visual thinking' or visualization as a viable analytical process, we aim to reduce the time between student's collection and analysis of data. Gardiner and Unwin (1986) note that it is difficult to rekindle student enthusiasm on returning from a field trip and that the full investigative cycle from experimental design to the presentation of conclusions is best done whilst fresh in the student's minds. Time is often short after a day in the field and data can be synthesised and ideas generated quickly with interactive visual techniques. Visual representations can be especially successful if sample data are collected by different groups of students over a large area as seeing an integrated whole provides a feeling of 'ownership' of the problem which can enthuse them (Gardiner and Unwin 1986). Immediacy, integration, realism and ownership can be successful motivators even after a hard day in the field. In combination with the novelty and levels of engagement offered by visualization as a data processing and interpretation technique they form a potent combination when applied to the analytical stage of fieldwork that Haigh and Gold (1993) identify as the most difficult for students.

3. VFC objectives
An understanding of the educational rationale behind the use of spatial and visual IT in fieldwork support gives rise to a number of more specific objectives adopted by the VFC. These in turn were used to guide the development of software, data collection and applications.

In particular, the VFC should be able to:

- Provide a software environment for the interactive visual exploration of spatially referenced information;
- Provide and support the development of a shared library of spatially referenced data to support field-based activity;
Enhance and extend all stages of field-based activity with appropriate teaching and learning materials;
Use affordable and accessible software, equipment and data resources;
Allow 'multiple world views' of the same spatial environment;
Maximise flexibility by allowing elements to be added and removed as necessary;
Allow data and software components to exist on distributed platforms.

4. Software design

As a solution an object-oriented component design was adopted, with each component performing a specific set of tasks necessary for the virtual fieldwork environment. These components communicate with each other via a central Hub that handles all the translation necessary for inter-component communication. Thus the VFC software consists of the following components:

The Database. A store of all the spatial and non-spatial data required by the VFC. Examples include raster and vector models, images, video, hypertext documents, virtual reality scene definitions etc. Despite the identification of the database as a single conceptual component, it need not be single file store but may be distributed over a variety of platforms.

The Metadatabase. To allow efficient querying and retrieval of data in the database, it is necessary to log all database items in a metadatabase. This includes such information as spatial location (point, linear and areal 'footprint'), keyword index, data source and database location. Metadatabase information facilitates the process of making the data interoperable. For example, metadata can be output using the Dublin Core metadata standard (Dublin Core 1997). This is seen as important since one of the objectives of the VFC is to act as an accessible repository of information relevant for fieldwork.

The Hub. All communication between components is routed through a central Hub to ensure database integrity and security. So, for example, if a component required the use of a raster Digital Elevation Model, it would pass the request to the Hub, which in turn would query the metadatabase, retrieve the file and pass it to the component. A simple communication protocol between Application Components (see below) and the Hub allows components to communicate with other and share data. New components that use this protocol can be added to the VFC by application developers.

The Toolkit. A number of the Application Components may have functionality in common. For example, the conversion of file formats, spatial search facilities ('retrieve all the digital photos that are within 4 km of this point'), compression/decompression of data etc. Conceptually, such common functionality is grouped in a single Toolkit that is accessible to all components, while in practice the tools themselves may be distributed over a number of platforms.

Application Components. These are the components that the user of the VFC is exposed to. Examples include software for terrain 'fly-through'; a video editor; a panoramic photograph viewer etc. These components are all based around visual interaction of some kind and provide the 'multiple world views' required by the VFC. They share common data by issuing requests to the database via the Hub.
5. Software implementation

A number of architectures exist that allow software components to be integrated, for example CORBA (Object Management Group 1998), JavaBeans (JavaSoft 1998a), and ActiveX (Microsoft 1998). Each technology requires the software components themselves to be 'wrapped' in some kind of Application Programming Interface (API). Rather than imposing a particular component architecture on application developers (which can be quite complex), the strategy for the VFC has been to provide a simplified communication protocol between components and the Hub using a three-tiered component-client-server architecture. The tiers consist of the Application Components, locally installed Clients, and the (possibly remote) Hub Server. Additionally, the Hub Server communicates directly with the toolkit, database and metadatabase (back end). This is shown diagrammatically in figure 1.

Note that this design does not preclude the use of component technology linking Application Components to Hub Clients (indeed, the clients are currently provided as both JavaBeans and ActiveX controls). By providing the three tiered structure the choice of component technology is left to the application developer.

This design has a number of benefits:

— Multiple software environments can be used for development. For example, one application developer might write a video editor utility on a PC in Visual Basic using the HubClient ActiveX Control, while another could write a Tcl/Tk application on a Unix machine for data browsing using the command

![Figure 1. VFC component architecture.](image-url)
line version of the HubClient. Both applications could communicate with each other and share common data via the Hub.

- Application Components can be added and removed without necessarily interfering with the system as a whole. A 'minimal VFC' would consist of the Hub, the toolkit, the metadatabase and database. Application Components can be added to the system independently.

- Database integrity and security is ensured by passing all data and metadata changes through a single gateway (the Hub). This gateway can be used to impose restricted (and secure) 'views' of the data and metadata depending on user account privileges.

- Components and data can remotely exist on a distributed network. For example, an Application Component could request the Hub to search a number of databases at different sites, extract the relevant data and save them locally.

Three forms of communication are required to implement a working VFC. Application Components communicate with a locally installed HubClient using a simple communication protocol; HubClients communicate with the HubServer, possibly over a network; the HubServer communicates with the local toolkit, metadatabase and database. Details of these three forms of communication are given below.

5.1. **HubServer—data connection**

The VFC database consists of data items such as GIS rasters, images and video clips as well as their associated metadata. To keep the system as portable as possible it was decided to separate the metadata (which are text-based and storable in any Relational Database Management System) from the data (which may not be storable in a RDBMS). Figure 2 shows the relationship schema used for storing the metadata.

The design consists of 15 tables, of which 10 describe the fundamental data types used by the VFC, 4 provide supplementary information and 1 stores compulsory metadata. Each object in the database (e.g. raster, audio clip, VR world etc.) has a unique ID which is used to relate it to its metadata (e.g. database file location, spatial location, caption, data source etc.). Note that many of the data type tables (shown in the left column of figure 2) contain only the Object_ID. This allows future extensibility by permitting new metadata to be associated with each type without affecting table relations.

The Hub communicates with the metadatabase using the Java Database Connectivity Connection (JDBC)—a set of Java classes that allow platform independent communication between a Java program and a RDBMS. Any database management system can be used, or even a collection of text files defining relational tables. The metadatabase can then be queried by a Java program issuing standard SQL commands. For example, to identify all the images in the database along with their descriptions that overlap with the rectangle defined by national grid coordinates 120000, 174000 and 450000, 320000, the following SQL command would be issued:

```sql
```
As the example illustrates, even relatively simple queries are represented by complex SQL commands. Consequently, this form of communication is hidden from the application developer and user by the Application-Client communication protocol.

5.2. HubServer—HubClient connection
This connection is also invisible to Application developers and users, and involves transmitting, data, metadata and client requests between a client and server on possibly remote machines. Both the HubServer and HubClient are written in Java, using Java sockets to route information between them.

Where a distributed environment is used and the server-client connection is
relatively slow, a typical VFC session might involve an initial transfer of relevant from the Hub to the local machine, followed by a catching of the data to be used by the Application Component.

5.3. HubClient—Application Component connection

The design aim of the communication between Application and HubClient is to make it as simple as possible while allowing sufficiently precise query of the VFC database. A simple communication protocol has been developed using six keywords with various optional arguments. Details of each command are shown in table 1. Together these commands allow the query, update and transfer of data and metadata between the Application and database.

The way in which these commands are issued depends on the preferred programming environment of the application developer. Currently, four options are available—batch file, Java class, JavaBean and ActiveX Control. The MS-DOS batch file or Unix shell file can be run by applications that can make operating system calls. Java programs can instantiate a Java object from the class provided, or they can use the JavaBean equivalent. Finally, application environments that can make use of ActiveX components can issue commands through an ActiveX control. In all cases, the command set is identical in syntax.

It would be expected that even this abstracted level of communication is hidden from the VFC user by the Application Component. Typically, the Application will provide a graphical front end to any data manipulation as illustrated by the examples below.

6. Example application components

traVelleR and panoraMap are example Application Components that take advantage of this architecture to communicate with the database and each other despite having been produced in different software environments. The former is a VRML/Java-based geo-referenced virtual reality environment for exploring 3D worlds such as terrain landscapes and urban models whilst the latter uses Tcl/Tk to provide a sense of immersion with seamless 360° panoramas. Both applications supply a highly interactive geographical interface to a rich database containing multimedia information and spatial data-sets.

6.1. vfc traVelleR

traVelleR provides a combined virtual reality and 2D interface to a spatial database through which the user may interact with a field study area. Its primary functions are the abilities to track movement through a VR scene on a 2D map, interactively vary the visual characteristics of the scene and query its properties. Other facilities include linked multimedia representations either viewed externally or embedded within the virtual scene, specification and animation of routes or ‘tours’ through the VR world, and dynamic addition or removal of data objects to and from the model. A specifically designed component application for urban model building is also available. traVelleR has been developed using the Virtual Reality Modelling Language (VRML97—VRML Consortium 1998), Java (JavaSoft 1998b) and the VRML External Authoring Interface (EAI) (Couch 1998). The EAI has been designed to extend basic VRML functionality and promote the construction of fully integrated web-based multimedia and VR applications by linking to external programming languages such as Java (Brutzman 1998). Initial impediments to
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Arguments</th>
<th>Return</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PING</td>
<td></td>
<td>Name/IP address of Hub Server or error message if connection failed.</td>
<td>Tests the connection with the hub.</td>
</tr>
<tr>
<td>GET</td>
<td>&lt;ObjectID&gt; [directory &lt;directory&gt;]</td>
<td>Requested object written to output directory and a message identifying the files transferred.</td>
<td>Retrieves the named object from the database and writes it to the output subdirectory. The output subdirectory can be overridden by using the 'directory' option.</td>
</tr>
<tr>
<td>PUT</td>
<td>dirName &lt;directory&gt; fileName &lt;fileName&gt; objectType &lt;objectType&gt; format &lt;format&gt; [metadataItem &lt;metadataValue&gt;]</td>
<td>Message indicating successful transfer, or error message if item not added to database.</td>
<td>Adds a new object to the database. The metadata items shown are compulsory, but others (see below) may be added in the same way.</td>
</tr>
<tr>
<td>LIST</td>
<td>[asKey</td>
<td>as table] [if &lt;itemList&gt;] [ifID</td>
<td>ifRequired</td>
</tr>
<tr>
<td>REMOVE</td>
<td>&lt;ObjectID&gt; [override]</td>
<td>Message indicating successful removal of item, or error message if unable to remove it.</td>
<td>Removes the given object from the database. Will not remove objects that linked to others (dependants) unless the 'override' option is given.</td>
</tr>
<tr>
<td>EDIT</td>
<td>&lt;ObjectID&gt; [metadataItem &lt;metadataValue&gt;]</td>
<td>Message indicating successful edit, or error message if unable to edit metadata.</td>
<td>Edits the metadata associated with given object. Any number of metadata items (see below) and associated values supplied will replace those stored in the metadatabase.</td>
</tr>
</tbody>
</table>
smooth VRML/EAI development reflect the rapid emergence of these new technologies and the fervent response to the problems augers well for the future of VRML. Opportunities for using virtual reality (Raper et al. 1998) and VRML (Fernandez 1997, Fairbairn and Paisley 1997) for cartographic applications have been quickly identified. Further potential cartographic benefits together with solutions to overcome some of the current obstacles to using VRML in combination with Java have been addressed (Moore et al. 1997). A GeoVRML Working Group (GeoVRML Working Group 1998) has been established within the VRML Consortium to address geographical issues such as coordinate systems, resolution, terrain representations and data interchange with a view to creating integrated geographical VRML nodes for incorporation into the VRML standard.

traVelleR is designed as a shell with generic functionality that can be applied to any geo-referenced model at any scale. The VRML model may be viewed independently, however dynamic links are made from a Java applet using EAI VRML nodes, and thereby the components of the model, may be accessed by the applet using a standard set of defined node names in the VRML code. The Java application can reference values and monitor events as they occur in the scene and can modify the scene directly returning events.

Geo-spatial header data required by traVelleR is defined in the VRML code as a custom or PROTO node. This delimits the spatial extent of the model in terms of a standard coordinate system. Transformations are made in the Java code between the VRML model, real world coordinates and 2D mapped coordinates. Once the model has been defined with real world coordinates other geographical functionality is enabled. Navigating through spatial environments can be confusing in both real and virtual worlds, therefore the ability to track location on a linked 2D map is a major benefit (figure 3 see p. 408). The virtual GPS and compass monitors location by coordinate, bearing and map position. Additionally a 2-way link has been created to allow the user to specify their viewpoint position and orientation in the 3D world from the map. The freedom of movement and bi-directional tracking between the virtual world and the map provides time-space flexibility not available in the field. For instance archaeological sites within a region may be visited in historical order rather than in one imposed by road networks on the ground.

By altering the visual characteristics of a model the power of virtual reality can be employed to make the environment ‘more real than real’ (Gillings 1997). Firstly, changing the drape on a terrain model allows comparison of terrain with derivative information such as slope and aspect and other continuous geographical data surfaces such as soil type, vegetation or population density. A mechanism for ‘sampling’ these multiple layers at any point on the VR surface has been implemented. Secondly, in many locations the vertical range of the DEM is relatively small. Terrain scaling enables interactive exaggeration of the height component of the scene.

traVelleR also provides the means to spatially search a locale in the VR scene for multimedia data and display them appropriately. Queries to the VFC database are made through direct calls to the Java HubClient class methods. Users simply select point locations using the cursor in the VRML scene and a LIST request for a spatial search is sent to the Hub. The returned metadata information is displayed. As items are selected from this metadata list a GET command is sent to the Hub, the items are downloaded and saved locally for subsequent playing or viewing. Keyword and gazetteer searches that take advantage of the HubClient Application Component Connection are also available.
Linked multimedia within the traVelleR framework can take a variety of forms. Figure 4 illustrates the range, interaction and relationship of traVelleR media object types. Current empirical work is assessing the merits of visualization of multimedia embedded within the VR model as opposed to viewing it in external application windows. If viewed externally 3D pointer symbols are added to the VR scene to indicate location and direction of the media. Geo-referenced sound in the form of commentary appears to be a useful aid in understanding the virtual environment. Anchored to a location with a proximity trigger embedded in the scene it can provide a spatially sensitive surrogate for the 'lecture in the field'.

VFC software emphasises student-centred learning where a region is explored in a freeform manner. However pre-constructed tours are useful during the preparation stage of fieldwork (Kent et al. 1997), particularly if 'led' by an expert on the region or topic of interest. At later stages students can use the functionality to produce their own tours or routes through a region to illustrate particular themes. Pre-defined fly-throughs and multimedia presentations can be viewed in traVelleR. The viewpoint in the VR scene is animated to travel between various sites of interest. At each site a proximity sensor triggers a commentary and associated video or photographic image.

Firmly grounding the technical capabilities of visualization software within a sound pedagogic framework is of paramount importance to the VFC. This has been accomplished by structuring a series of Virtually Interesting Projects (VIPS) as worked examples related to the generic Application Components. These can be used as tested demonstrators for teachers and also to guide student's use of the software throughout their fieldwork module. For example, staff may be inspired by VIPS to apply the generic functionality to their own field interest areas and students may learn the software during an initial class-based exploratory VIP before using it more extensively for analytical purposes in later stages of the fieldwork. Examples of VIPS currently being tested using traVelleR include exercises involving route planning, soil sampling and the urban environment.

Route planning can be appropriate at the preparation stage of fieldwork teaching.

Figure 4. Linking between Java program, VRML model and multimedia data objects that creates the traVelleR virtual environment.
Figure 3. Navigation in traVelleR using a 2D map to track movement across the 3D terrain at Haytor in Devon. Results from raster sampling across multiple data layers are shown (right) along with multimedia views of the scene (bottom).

Figure 5. Building an urban land use model by floor level.
Students use the VR model as a priming exercise to assess routes for data collection. In the physical geography domain this might involve using a terrain model to set a route that maximises vegetation cover boundaries crossed in the field, whilst in the realm of human geography an urban area might be assessed for orientation purposes and to identify suitable data collection points.

Soil sampling is a traditional field-based exercise that may be extended within the virtual environment. Students analyse soil samples taken from widely scattered locations in the field and compare them with soil and other environmental variables sampled in traVelleR. Aspects of scale, spatial variability, cartographic representation and accuracy may be critically assessed. This provides an exercise useful at the analysis and debriefing stages of fieldwork.

Urban fieldwork is also being addressed within the VFC using virtual environments. Although the trend in VR is to build high fidelity street scenes, the visualization of more abstract characteristics of urban areas can be especially effective. Land use is usually mapped on a 2D basis which is less than ideal for the 3-dimensional nature of towns and cities. Mapping land use by floor level provides a more authentic representation of the phenomenon. The functionality to produce such 3D maps is being developed to enable model building from field survey data and a 2D vector coverage so that data collected by students can be visualized rapidly (figure 5). The degree of engagement involved in constructing of the model, the appropriate spatial view and the subsequent interaction are examples of the educational rationale behind the VFC software being put to use in the field. The full model may be constructed by combining data and component parts produced by individuals which establishes a degree of collaboration in the project.

6.2. tcf panoraMap

Virtual environments that offer photographic detail at discrete locations through the seamless 360° panoramic image are provided by panoraMap. A sense of reality and immersion are supplied through interaction with the images. Spatiality and navigation are provided through links between image and map and between successive images.

PanoraMap has been designed as a simple geographical user interface to multimedia information that makes the production of a multimedia database with geographical footprints readily achievable. The basis of the geo-referencing is a standard ‘GIF’ format backdrop map. A user determines the extent of a region under study in such a map by providing bounding coordinates. Data files can then be selected from a system file manager and meta data relevant to the file type added. This includes locational information that can be acquired through a text-entry interface or by directly clicking on the map. When a data file is selected a dynamic symbol is added to the map representing that datum. This combination of file manager and interactive map allows a spatial database of multimedia information to be produced rapidly.

Media objects are viewed in an appropriate media display window by clicking the map symbol. A degree of interaction occurs between the dynamic map and the media viewer depending upon the media type and meta-information supplied. For example, images contain a ‘bearing’ meta-field which allows the direction of the image to be shown on the backdrop map whereas text files contain information about the type of text and clicking on ‘data’ text files produces dynamically linked statistical views. Media types are supported in a variety of ways as illustrated in
figure 6 where the solid oval shows the scope of panoraMap and the arrows represent internal and external links. Text, hypertext and imagery are displayed within the application, represented by the bounding oval in figure 6, and so links between map and media viewer are closely coupled. This means that interaction with either one of these windows can result in a response in another. Currently text and hypertext are linked in a one-way relationship (demonstrated by the solid arrows). Sound and video are displayed in external applications. These are user defined with expected defaults. For example a Windows NT machine might use 'mplay32.exe' for video and 'sndrec32.exe' for sound.

The application is designed to be used with familiar systems software and external devices. The dashed arrows show devices and software that might be used to collect and create the media. These include hardware such as scanners, GPS, dictaphones, digital cameras, video cameras and video cards and software such as VideoBrush (VideoBrush Corporation 1998) and PhotoVista (LivePicture Corporation 1998) for capturing and creating panoramic imagery, image editing software, spreadsheets and text editors.

The panoramas operate in much the same way as in a number of other Internet applications. By displaying the field of view on the map as the image is rotated the panoramic photographs can be compared with thematic information displayed on the backdrop map. Comparisons can be made between the realistic photographs and other thematic information by loading, and interactively selecting, supplementary backdrop images of the same area. Examples using historical maps, population density data, elevation and derivatives, and satellite data, are shown in dynamic figure 7.

Some additional features are demonstrated in figure 7, notably tools for reading and displaying GPS tracks and performing exploratory visual analysis with enumerated data. The software plots data downloaded from the Garmin GPS 12XL, showing routes taken and data collection sites marked as 'waypoints'. When a waypoint symbol is clicked a file manager dialogue allows other digital data (photographs,
dictaphone notes, video interviews, text files) to be geo-referenced enabling staff to set up a VFC database and students to add a geographical footprint to their qualitative and quantitative field data. This simplifies and hastens the process of integrating primary and secondary data. Enumerated information might comprise of secondary data obtained by staff from population surveys, or primary data based upon the student’s own observations in the field such as quantitative measures of retail activity (see figure 7). Additionally students can prepare for fieldwork by planning routes using the information in panoraMap, digitising them on the backdrop maps and loading them into GPS which navigates them through the field.

PanoraMap uses the HubClient Application Component Connection to provide communications functionality. This means that the entered data can be stored on a remote VFC database (using the PUT command) and that the panoraMap map interface can graphically define criteria by which a remote server is searched for data (using the LIST and GET commands). The backdrop map features a draggable selection box. This selects and returns data within the specified region from remote Hub Clients and is thus effectively a map window to the WWW implementing the kind of functionality mentioned by Fernandez et al. (1997) and making Cartwright’s (1997) real time ‘ground truthing’ eminently feasible.

A sample data-set located in Dartmoor, UK is shown using panoraMap in figure 8. Collaboration between VFC, the Dartmoor National Park and fieldwork teachers in the University of Leicester, UK, has lead to the production of a rich spatial database for Haytor Down. The location features a variety of areas of fluvial and ecological interest, management conflicts between tourism, conservation and land-owners, significant archaeological sites and spectacular granite tors. Here an aerial photograph backdrop map (top right) shows GPS routes taken on four successive days. The map is linked to the Hub, which has been used to search for media objects. These are shown with green symbols on the map and with icons, file names and other information on the Hub Registry (left). Media objects are displayed by clicking the symbols or object names. Two interactive panoramic images with fields of view displayed on the backdrop map are shown here along with a video clip. The ‘Maps’ window (top centre) shows other backdrop maps that can be viewed in turn.

Considerable effort has gone into ensuring that the functionality outlined here will encompass a broad scope of fieldwork tasks in a whole range of subject areas and example VIPs that provide worked example projects are being tested with the software. By incorporating a variety of types of data file and means of displaying and interacting with them a whole range of projects can be supported. One of the main features that makes the software so flexible is the ability to link between media
Figure 8. panoraMap virtual environment: Haytor Down, Dartmoor, UK.
objects. Links are represented by dotted arrows on figure 6. They are created by dragging selection boxes on the images and then choosing media files to which the boxes should link when subsequently clicked. The way in which the panoramic images used in figure 8 can be linked together is illustrated in figure 9. The images are rotated until they face one another (see symbols on backdrop map, top left). Selection boxes are then dragged on the panoramic images (left). Clicking a box produces an 'Image Annotation Window' (bottom centre). Here the box on the top image has been clicked to reveal a window that has spaces to enter relevant information about the annotation. This may take the form of short caption, or as is the case here, another media object. The 'linkFile' button produces a GUI to select the media file to display when the annotation box is clicked. A system file manager is used to select a local file in this instance (bottom right).

These links are similar to the 'sketching' functionality described by Fernandez et al. (1997). They have a variety of uses including annotation (link to text), increasing detail (link to zoomed image) and 'ground truthing' (Cartwright 1997) (link to hypertext page with WebCam image). Annotation provides the means for a digital student field sketch and the production of routes through the virtual environment. Both of these functions provide possibilities for using one student’s observations as another’s data and for formal assessment.

The two approaches to virtual environments introduced here are complimentary. traVelleR’s VR offers flexibility of viewpoint and spatial information viewed whilst panoraMap’s discrete photographic world offers currency and detail. Efforts are

![Figure 9. Links between media objects in panoraMap.](image)
underway to make the types of communication that occur between the two applications more elegant.

7. Empirical feedback

At the time of writing informal tests using prototype software are taking place in the teaching of a number of staff, and collaboration with the educationalists at the Dartmoor National Park is continuing. The Department of Geography, University of Leicester undertakes an annual Easter vacation Field Course to South Devon. In 1998 both of the software elements described here were incorporated at the analytical stage. Students were guided through the software by staff and then observed and asked to complete a rudimentary feedback form. The empirical evidence gained strongly suggests that the notion that a reduction in the time between the collection and analysis of data has substantial educational potential is legitimate.

Students indicated that the software provided a focus for the day’s work and enhanced the projects where it was used. Encouragingly responses indicated that students saw potential benefit from using the software before carrying out the experimental work, both during and prior to the field course, and the notion that it was beneficial to see collected data displayed in the evening returned a strong consensus. Students were asked whether they could see the utility of the software elsewhere in the curriculum and in other fieldwork projects, which they could, and for criticisms and possible additions, which are being addressed. When asked to identify the most positive aspect of the software many of the initial objectives were identified. Comments included ‘the amount of information’, ‘the ability to compare data’ and ‘seeing what you have achieved that day’. The ‘visual thinking’ paradigm was used by employing visualization techniques for analysis in panoMap. Pleasingly, students responded well to this with the following ‘most positive aspects’ being identified: ‘enables us to see patterns and changes in relationships very well’, ‘simple to use, shows spatial variation’ and ‘very good representation of information’.

The immediacy of loading information that students have recorded during the day into a graphical system on returning from the field certainly appears to stimulate interest. Mapping the routes recorded by GPS and associating them with secondary data was a particular success. By enabling students to interact with their field data immediately they can process their information and integrate primary and secondary data sources whilst making use of the geographical context that is fresh in their minds. Comparisons are readily made between the two, satisfying an identified fieldwork objective, and the kudos granted to digitally mapped primary data encouraged students to question the validity of the secondary data and consider their origins. Students using the software also appeared to make more of their time and efforts in the field, and to appreciate the fact that they could combine the group’s data as a whole. This resulted in the production of a variety of organisational structures to ensure efficient data collection and quality control within groups.

Students also appeared keen to see their work used in future teaching and gained some pride from the knowledge that the fruits of their efforts would endure beyond the lifetime of the field course. A strong and extremely positive sense of the ownership factor was identified in these informal tests.

8. Conclusion

The design and implementation of the software described here provides enormous potential for supporting fieldwork both during the field course and in the ‘preparation’
Virtual environments for student fieldwork

and 'debriefing and feedback' stages of fieldwork that were not incorporated into these initial tests. These stages will be supported with VFC software and example VIPs in the future. Additional VIPs are being produced that show the Application Components applied to very different subject and geographic areas and supplementary Application Components are under development to support other types of fieldwork.

Data that provide a geo-referenced digital library to support field-based activity have been successfully shared between Application Components running on different platforms, using different operating systems. Speeds of data transfer between remote machines are comparable to web-based data transfers, with successful data sharing demonstrated both within Europe and across the Atlantic.

Whilst a more formal survey of staff and student reactions is both required and ongoing, immediate reactions to the software tests were almost wholly positive. The software undoubtedly enhanced the analytical stage of the fieldwork as described above. Whilst some modifications will always be necessary teachers were surprised and impressed by the quality of analysis performed by the students using the visual techniques and the utility of the database and the functionality. The stamina generated by the software was also noted. One member of staff with no previous IT experience went as far as to describe the project that they ran in conjunction with the VFC software as the best day's fieldwork they had ever been associated with.

Acknowledgments

The work reported in this paper was conducted in association with the Virtual Field Course (VFC) project, which is a joint venture between Birkbeck College (University of London), the University of Leicester and Oxford Brookes University and is funded by the UK Joint Information Systems Committee Technology Applications Programme (JTAP). Contributions to the content from the entire team consisting of Peter Fisher, Alan Jenkins, Tim McCarthy, Jonathan Raper, David Unwin, Nathan Williams and the named authors are gratefully acknowledged. Any errors or omissions are, of course, the responsibility of the authors.

This work has only been possible with the advice, ideas, help and enthusiasm of a large number of people including: The University of Leicester Department of Geography; Chris Taylor, Colin Williams, Dominic Kniveton, Jane Wellens, Tony Parsons and David Turnock; University of Leicester First Year Undergraduate Geographers; Jaakko Kähkönen and members of the International Cartographic Association Commission on Visualization; Devon County Council; Dartmoor National Park.

The Census data are copyright of the ONS and are provided with the support of the ESRC and JISC. The images use boundary material provided by the ED-Line Consortium which is copyright of the Crown and the Post Office and backdrop information which is copyright of the Ordnance Survey.

References

Virtual environments for student fieldwork


