Scientific visualization and exploratory data analysis of a large spatial flow dataset

Gou, Zaiyong, Ph.D.

The Ohio State University, 1993
SCIENTIFIC VISUALIZATION AND EXPLORATORY DATA
ANALYSIS OF A LARGE SPATIAL FLOW DATASET

DISSERTATION

Presented in Partial Fulfillment of the Requirement for
the Degree of Doctor of Philosophy in the Graduate
School of The Ohio State University

By

Zaiyong Gou, B.S., M.A.

* * * * *

The Ohio State University

1993

Dissertation Committee:
Duane F. Marble
Morton E. O'Kelly
Randy W. Jackson

Approved by
Duane F. Marble
Department of Geography
DEDICATION

To My Parents
ACKNOWLEDGEMENT

Of my thirteen years of being a geographer, the last four years under the supervision of Prof. Duane F. Marble at The Ohio State University has been the most exciting and most memorable. His wisdom and profound academic insight showed me what geography as an extraordinarily difficult discipline should be and could be. He is always the person who stands very high and sees very far, exploring the most challenging frontiers of geography relentlessly. As my academic adviser, his unfailing encouragement, guidance, patience and professional courtesy have always been my inspiration to work more creatively and more enthusiastically. He leads us to a high intellectual plateau and offers us the unlimited freedom to face the challenges and to pursue academic excellence. I am always grateful for the paths and opportunities you have pointed out to me.

My gratitude also goes to Prof. Edward J. Taaffe, Prof. Morton E. O'Kelly and Prof. Randy Jackson for their support, comments and effort during my dissertation research. I would also like to thank my fellow students, Qin Tang, Demin Xiong, Yong Lao and Lin Liu, with whom I had numerous interesting discussions in the daily gathering and great recreational time in almost every Friday night. Thank you, guys.

Finally I want to express my gratitude to my wife, Guangmin, for her unconditional understanding and support of my work and for keeping me fantastically fed and cared.
VITA

August 14, 1963 ......................................... Born - Sichuan Province, P.R. China.

1984 ............................................................. B.S. in Geography at Peking University, Beijing, China.

1988 ............................................................. M.A. in Geography at the University of Toronto, Ontario, Canada.

1989 ............................................................. Teaching assistant in the Dept. of Geography, The Ohio State University.

FIELDS OF STUDY

Major Field: Geography
# TABLE OF CONTENT

<table>
<thead>
<tr>
<th>TABLE OF CONTENT</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEDICATION</td>
<td>ii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td>iii</td>
</tr>
<tr>
<td>VITA</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>ix</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>I INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>Dealing With Massive Flow Data</td>
<td>1</td>
</tr>
<tr>
<td>Research Context</td>
<td>4</td>
</tr>
<tr>
<td>Visualization and Exploration of Massive Flow Data</td>
<td>5</td>
</tr>
<tr>
<td>SV, EDA and Dynamic Graphics as a Powerful Research Tool to Flow Data</td>
<td>5</td>
</tr>
<tr>
<td>Approaches to the Problem of Massive Flow Data</td>
<td>8</td>
</tr>
<tr>
<td>Implementation</td>
<td>9</td>
</tr>
<tr>
<td>Dissertation Organization</td>
<td>10</td>
</tr>
<tr>
<td>II LITERATURE REVIEW</td>
<td>12</td>
</tr>
<tr>
<td>Scientific Visualization</td>
<td>12</td>
</tr>
<tr>
<td>Scientific Visualization and Cartographic Communication</td>
<td>13</td>
</tr>
<tr>
<td>Scientific Visualization of Spatial Data</td>
<td>16</td>
</tr>
<tr>
<td>Exploratory Data Analysis</td>
<td>21</td>
</tr>
<tr>
<td>Dynamic Graphics for Spatial Data Analysis</td>
<td>24</td>
</tr>
<tr>
<td>Identification</td>
<td>25</td>
</tr>
<tr>
<td>Deletion</td>
<td>26</td>
</tr>
<tr>
<td>Linking</td>
<td>26</td>
</tr>
<tr>
<td>Scaling</td>
<td>26</td>
</tr>
<tr>
<td>Rotation</td>
<td>28</td>
</tr>
<tr>
<td>Scatterplot Brushing</td>
<td>28</td>
</tr>
</tbody>
</table>
APPENDICES

A. Software Usage............................................................................................................. 157
B. The X Window System ............................................................................................. 178
C. The OPEN LOOK Graphical User Interface Conventions ............................ 184
D. The XView Graphical User Interface Toolkit ................................................... 187

BIBLIOGRAPHY .............................................................................................................. 190
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURES</th>
<th>PAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dichotomy of exploratory and confirmatory data analyses</td>
</tr>
<tr>
<td>2</td>
<td>Deletion operation</td>
</tr>
<tr>
<td>3</td>
<td>Highlighting operation of scatterplot brushing</td>
</tr>
<tr>
<td>4</td>
<td>Scatterplot brushing conditional on a single variable</td>
</tr>
<tr>
<td>5</td>
<td>Shadow highlighting operation</td>
</tr>
<tr>
<td>6</td>
<td>MacSpin: Rotation and separation</td>
</tr>
<tr>
<td>7</td>
<td>Graduated shading in the corridors of constant band width</td>
</tr>
<tr>
<td>8</td>
<td>Lines and numbers representing magnitudes of flows</td>
</tr>
<tr>
<td>9</td>
<td>Accumulating vector values according to trips on each path</td>
</tr>
<tr>
<td>10</td>
<td>VAP display of traffic pattern</td>
</tr>
<tr>
<td>11</td>
<td>Continuous flow vector field and flow rerouting</td>
</tr>
<tr>
<td>12</td>
<td>Arrows with different shading patterns</td>
</tr>
<tr>
<td>13</td>
<td>Different two directional arrows</td>
</tr>
<tr>
<td>14</td>
<td>Displaying flows using SeeNet</td>
</tr>
<tr>
<td>15</td>
<td>Theoretical framework of flow data types and research directions</td>
</tr>
<tr>
<td>16</td>
<td>Strategy of flow analysis using SV, EDA and dynamic graphics</td>
</tr>
<tr>
<td>17</td>
<td>Blink Comparison</td>
</tr>
<tr>
<td>18</td>
<td>Forward brushing</td>
</tr>
<tr>
<td>19</td>
<td>Backward brushing</td>
</tr>
<tr>
<td>Page</td>
<td>Section Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>20</td>
<td>Program structure of SEFLOW</td>
</tr>
<tr>
<td>21</td>
<td>Title page of SEFLOW</td>
</tr>
<tr>
<td>22</td>
<td>Choropleth mapping of In-Flows to each region</td>
</tr>
<tr>
<td>23</td>
<td>Controlling data range using slider bars</td>
</tr>
<tr>
<td>24</td>
<td>Directional flows using the lines and numbers method</td>
</tr>
<tr>
<td>25</td>
<td>Net flows using the lines and numbers method</td>
</tr>
<tr>
<td>26</td>
<td>Two way flows using the lines and numbers method</td>
</tr>
<tr>
<td>27</td>
<td>Directional flows using proportional band width</td>
</tr>
<tr>
<td>28</td>
<td>Net flows using proportional band width</td>
</tr>
<tr>
<td>29</td>
<td>Two way flows using proportional band width</td>
</tr>
<tr>
<td>30</td>
<td>Directional flow using graduated shading with constant width</td>
</tr>
<tr>
<td>31</td>
<td>Net flows using graduated shading with constant width</td>
</tr>
<tr>
<td>32</td>
<td>Two way flow using graduated shading with constant width</td>
</tr>
<tr>
<td>33</td>
<td>Polygon collapsing</td>
</tr>
<tr>
<td>34</td>
<td>Double view</td>
</tr>
<tr>
<td>35</td>
<td>Default forward brushing</td>
</tr>
<tr>
<td>36</td>
<td>Identifying outliers with a rectangular brush of 45 degrees</td>
</tr>
<tr>
<td>37</td>
<td>Forward brushing with spatial interactions</td>
</tr>
<tr>
<td>38</td>
<td>Default backward brushing</td>
</tr>
<tr>
<td>39</td>
<td>Backward brushing: define anchor regions as destinations</td>
</tr>
</tbody>
</table>
40 Backward brushing: define anchor regions as origins .................................. 127
41 Backward brushing: lump all regions together ............................................. 127
42 Pattern searching over the entire time span .................................................. 129
43 Pattern searching in the same time period .................................................... 129
44 A Generic Spatial Interaction Visual Analysis Package .................................... 152
45 Black and white screen dump of the title page of SEFLOW ............................ 158
46 Main panel layer of SEFLOW ........................................................................... 160
47 Cartographic representation module of SEFLOW ............................................. 163
48 Double view of SEFLOW .................................................................................. 167
49 Forward brushing module ............................................................................... 169
50 Backward brushing module ............................................................................ 173
51 Pattern searching module user interface ......................................................... 176
52 Hierarchical Structure of the X Window System ............................................. 181
CHAPTER I
INTRODUCTION

Dealing with Large Flow Data

Understanding and analyzing large, complex spatial-temporal flow data has been a very difficult problem both in general geographic analysis and cartography. Such spatial-temporal flows can be exemplified by inter-regional population migration flows, urban traffic flows, various cargo flows or international trade flows.

Geographers have been perplexed by this problem for many decades, and there has not been a feasible solution for it. To be specific, the difficulty stems from three aspects: (a) the voluminous nature of the data, (b) the lack of effective cartographic representations, and (c) the complexity of the spatial interaction mechanisms the flow data contains.

First, flow data usually comes in a staggering volume because flows occur within a complex spatial structure and have a temporal dimension. A typical example is the internal migration of U.S. on a county level. Each origin-destination (O/D) matrix is a 3141 by 3141 matrix with nearly 10 million entries (Tobler, 1987). With a temporal dimension of 20 to 30 years, geographers are easily confronted with 200 to 300 million numbers! In addition, there are many social-economic variables associated with migration flows such as per capita income, crime rate, population density, unemployment rate, to name but few, which are indispensable in understanding the underlying mechanisms of
regional interactions. How can this tremendous amount of information be comprehended and analyzed?

Secondly, flow data is graphically difficult to present and visually hard to understand. Imagine drawing flows among, say, 50 regions on a map of normal dimensions. This map has to portray 2,500 flows (a 50 by 50 matrix) of various magnitudes and different origins and destinations. Such large a number of flows would severely clutter the map, presenting an overwhelmingly confusing pattern. Moreover, geographers are usually encountered with much more complex situations, such as the 3141 county-level migration flows in the U.S. This constitutes a formidable challenge to modern cartography. An extensive review of the literatures by the author indicates that there does not exist adequate cartographic representation methods capable of displaying complex spatial flows cleanly and clearly.

Thirdly, spatial-temporal flows are a classic example of complex spatial interactions, which are driven by various social, economic and behavioral dynamics. Identifying the patterns of interactions and the underlying rationales has been a difficult task for generations of geographers because these insights are trapped inside the numerous rigid O/D tables and other relevant data. Typically, a great number of rows and columns of an O/D matrix continue pages after pages in a tedious fashion, devoid of spatial as well as social-economic context. Traditionally, geographers can only extract limited characteristics of the spatial interactions from the O/D tables through mathematical/statistical modeling, such as the goodness of fit to an entropy model or to a linear regression model. They are
unable, however, to go beyond this point so as to crack the nutshell of the rigid flow data, and explore the inner mechanisms visually and interactively.

The combination of these three factors makes the analysis of a large flow dataset with a considerable temporal dimension a formidable task to tackle. Nonetheless, modern geographers are now equipped more than the traditional abstract analytical methods of scientific inquiry. We now have access to much faster and more powerful computers that are capable of generating complicated graphics at an amazing speed. Scientists are increasingly realizing that, besides our mental abstract thinking, human vision (or more specifically, our visual thinking ability) is another very effective tool to process massive and complex data. Research in this respect has spurred the accelerated development of two intertwined fields of scientific methodology: exploratory data analysis (EDA) and scientific visualization (SV). Since flows are intrinsically complicated and visual, approaching the problem of flow analysis from these new methodological perspectives may prove to be very fruitful. Unfortunately, no one has done any research in this area.

Therefore, it is the intention of this dissertation research to examine the current body of literature pertaining the cartographic as well as computer graphical treatment of the large and complex spatial flow analysis, to cultivate theories and techniques of EDA and SV to solve this problem, and to develop a software system to visualize and explore spatial flows. All these will be further elaborated in later sections.
Research Context

Let us first have a brief overview of previous work about the flow analysis from the stand points of cartographic representation, computer flow mapping and interactive graphics.

Although there are excellent examples of good manual cartographic representation of flows in the history of cartography, which can be traced back to Charles Joseph Minard (1861) (Tufte, 1983) and Ravenstein (1885) (Tobler, 1987), their success has depended largely upon the artistic talent of the individuals involved and the fact that the flow information presented was generally very limited. However, modern times are characterized by the explosion of spatial information on the one hand, and the availability of powerful computers on the other. Therefore, geographers should be able to handle complex, and large amounts of spatial flow data.

Using computer graphics to map spatial flows or spatial interactions was pioneered by, among others, Kern and Rushton in 1969 and Wittick in 1976 using fairly primitive graphics techniques. This was subsequently overshadowed by Tobler’s path-breaking experiments with mathematical modeling (1981) and cartographic representation of migration flow data (1987). Although Wittick and Tobler have created many innovative and effective cartographic methods for presenting flows, they are capable of improving the clarity and reducing the clutter of flow maps to a limited extent. Their work did not solve the problem of portraying and understanding large, spatial-temporal flow data. Simply, there does not appear to be an optimal cartographic representation method which is capable of displaying a large quantity of flows in a clean, uncluttered manner.
Fortunately geographers are not alone in dealing with the analysis and visualization of flows. For example, urban transportation researchers are interested in portraying urban traffic in three dimensional perspective views and presenting traffic volumes in an imaginary transit grid using color scheme or line width (Noguchi, 1977). However, the most important work has been done by Becker and his colleagues at the AT&T Bell Laboratories. They were the first to use interactive, dynamic graphics to map network flows (long distance calls) (Becker et al., 1990a, 1990b, 1991a and 1991b). Regrettably, Becker’s work does not escape the usual pitfalls encountered by non-geographers dealing with inherently spatial data. The cartographic representations are weak, and although it provides many interesting visualization techniques, it is not fashioned as an ideal tool for the spatial flow analysis as expected by a geographer.

This dissertation research should draw valuable experiences from the current body of literature and go beyond the limitations of the traditional cartographic methods and the non-geographers’ usage of interactive graphics in portraying flows. The ideal approach to a large flow dataset is to combine excellent cartographic representation with well-designed dynamic graphics of scientific visualization and exploratory data analysis. The next section provides a more detailed explanation on this approach.

Visualization and Exploration of Large Flow Data

SV, EDA and Dynamic Graphics as a Powerful Research Tool to Flow Data

Scientific visualization (SV) is a rapidly growing field as a result of the tremendous development of applied computer graphics in data analysis and the increasing
recognition that mathematical and analytical methods alone are not enough to cope with
the challenges of dealing with the large amounts of data generated by the modern
information explosion (Maceachren and Ganter 1990). Visualization using dynamic
computer graphics has emerged as a powerful tool for scientific research because we
absorb and process visual information much faster and more effectively than verbal
information or other forms of signals. In our daily activities, we rely heavily upon visual
images to digest, organize and analyze complex, abstract concepts. These concepts are
usually simplified or transformed, and sorted out as inter-related visual objects or
structures in our mind. Sometimes a simple diagram can facilitate our solving of a highly
abstract and complex problem. Since human vision is such a powerful tool in information
processing, the trend is to combine scientific visualization with exploratory data analysis
to deal with very large, complex datasets.

Exploratory data analysis (EDA) is also a relatively new field that has experienced
rapid development in the last decade. It differs fundamentally from the conventional,
confirmatory approach to scientific research. It is especially useful for very large or
complex datasets or phenomena, where we have only inadequate theory to predict what
is going on in the data. Confirmatory data analysis generally follows a simple trajectory
which begins with the creation of a hypothesis based on current theory, and then, through
rigorous statistical procedures and other analysis and testing, the hypothesis is considered
to be either proved or rejected. This methodology cannot be applied to an unknown
dataset of high complexity since we simply do not know enough to generate viable
hypotheses. Consequently, we need first to develop ways of navigating through the
dataset, trying to understand its structures, patterns, and relationships, seeking something that can be related to what is known, or simply something which triggers our curiosity. Then hypothesis can be generated based upon our preliminary comprehension of the data, and confirmatory data analysis can be conducted.

EDA is inherently related to SV, since visual exploration of data is a principal tool of EDA, and the purpose of SV, in turn, is to explore complex data. Combining SV and EDA creates a powerful tool with which to tackle the superficially intractable problem of comprehending huge amount of spatial flow data.

Dynamic computer graphics is a very important tool for SV and EDA. With the availability of fast computers and large amounts of memory, we are able to generate highly sophisticated 2D and 3D displays of data in near real time. This allows the data analyst to interact with data display in a dynamic fashion. Instead of spending extended period of time to generate a static image of the data, we can create hundreds of frames of inter-related pictures portraying complex data in a matter of minutes. We can directly manipulate the data using a variety of methods and can see feedback from our actions on the screen instantaneously. Dynamic graphics to complex data analysis is much like a sturdy four wheel drive or a nimble helicopter to an explorer of a formidable tropical jungle, uncharted by human expedition. We can dissect, rotate, brush, extract, identify, search, isolate, link, zoom in, zoom out in direct data manipulation on a computer screen much the same way as the explorer drives through or hover above the jungle for exciting discoveries. This opens numerous opportunities for understanding the structures and mechanisms contained in the data.
Approaches to the Problem of Large Flow Data

Now let us go back to look at the problem of tackling large space-time flow data. The combination of EDA, SV and dynamic graphics should be very applicable and effective in this respect, since it is generic methodology to large datasets, and flow data is not an exception.

The traditional cartographic methods or computer mapping are inadequate simply because it is virtually impossible to portray the entirety of the flow data on one or several maps. In addition, a spatial analyst may not necessarily be interested in every detail of the spatial flows. He/she may only be interested in a certain portion or aspect of the data, such as flows falling within certain thresholds, or interactions among selected regions.

Alternatively, confronted with an overwhelmingly complex flow dataset, the spatial analyst may not have a clue as to what information to extract because he does not understand what is going on inside the data. Most likely the first thing he/she wants to do is simply to break into the data, and obtain a feel for it by exploring or navigating through the jungle of complex spatial interactions and relations. By doing do, he/she may find something intriguing or unusual and decide to pursue further, or comprehend the entirety of the data by digesting one portion at a time.

Therefore a good solution to the spatial problem is to create a scientific visualization environment using interactive, dynamic graphics to explore the flow data. Within this environment, an analyst should be provided with many visualization tools to navigate through the large dataset, to extract important information, and to view the dataset from different perspective and different levels of spatial aggregation. He/she can search for
patterns, and discover anomalies and relationship through the instantaneous visual feedback of dynamic graphics. And eventually he/she may achieve a comprehensive understanding of the flow dataset and be able to generate interesting hypotheses.

A major task of this research is to develop theories and innovative techniques for exploring and visualizing complex flow data, and to lay a solid theoretical foundation for the spatial flow analysis. The visualization techniques developed here include three cartographic representation methods, as well as interactive visual information extraction, region collapsing, blink comparison, double view, dynamic and static threshold setting, animated display, visual information storage and retrieval, backward and forward brushing, mirror imaging of spatial and aspatial expression of flow data, multivariate linked view, pattern identification and horizontal/vertical searching, interactive data manipulation and dynamic choropleth mapping, etc.

Flow data is fundamentally spatial and temporal. However, the time aspect is left out for the most part in this research because attacking the spatial domain is already a very thorny task. Although the work touches the temporal dimension through pattern searching in the time series, and by allowing flow visualization at different time periods, the treatment has only scratched the surface of a very difficult and perplexing topic.

**Implementation**

A major software system (SEFLOW) is created as a proof-of-concept implementation of these theories and techniques. The name of the software stands for "Scientific Visualization and Exploratory Data Analysis of Large Spatial FLOW Data". It is a five
module program written under the X Window System using XView as graphical user interface toolkit. A combination of Xlib functions and C code is used to generate graphics and manipulate the data. It is a stand-alone program, highly portable over variety of hardware platforms, and is intuitive and easy to use. The five modules of SEFLOW are Cartographic Representation Module, Forward Brushing Module, Backward Brushing Module, Pattern Search Module, Interactive Data Manipulation/Dynamic Choropleth Mapping Module. Together, they constitute a coherent, integrated and powerful toolkit to unveil the intricacies of spatial-temporal flows.

**Dissertation Organization**

The dissertation is organized into six chapters. This initial chapter introduces the problem of understanding and analyzing a large, complex spatial-temporal flow dataset, and the proposed solutions to the problem. Chapter II reviews extensively the relevant literature pertaining to geographic scientific visualization, exploratory data analysis, dynamic graphics and flow mapping by computer-assisted cartography. Chapter Three defines the scope of the problem and provides a systematic treatment of the methodologies proposed to visualize and explore spatial flows.

Implementation details of the dynamic flow visualization system are covered in Chapter IV. This chapter discusses in considerable length how the logical model of the visualization system is mapped to an operational software system. Using a number of computer screen photographs, the fives modules of the implementation program are
discussed extensively with regard to their functionalities and the underlying visualization techniques and theories.

Chapter V revisits the research critically, and tries to provide a viable assessment to the effectiveness of the flow visualization system developed. Chapter VI summarizes the findings of the dissertation research and identifies the limitations of the research and implementation program as well as suggesting further improvement. It also tries to point out the future research directions of spatial flow analysis. A model of a generic spatial interaction visualization and analysis tool is envisioned, which extends two dimensional graphics to three dimensional presentation, includes non-network flows and network flows, and incorporates flow visualization with flow modeling process as well as diagnostic statistics for initial data probing.

Finally the appendices contain supplementary yet important information concerning the dissertation work. Appendix A provides a discussion on system usage, which can be used as a user’s guide to SEFLOW. The structure and significance of the X Window System, which is the software development environment for SEFLOW, is explained in Appendix B. Appendix C discusses in detail the Open Look Graphical User Interface adopted in SEFLOW design. And finally, Appendix D contains information on the XView GUI toolkit used in the implementation.
CHAPTER II
LITERATURE REVIEW

This chapter attempts to put into perspective many fields of study relevant to spatial flow data visualization and exploration. These include Scientific Visualization (SV), Exploratory Data Analysis (EDA), and dynamic graphics for data analysis. Also discussed are previous works on the cartographic representation of spatial flows, and network flow visualization. In short, a thorough, critical review of the body of literature is provided here.

Scientific Visualization

In recent decades, we have witnessed a tremendous growth of interest and research in a completely new approach to scientific investigation: using visual methods to tackle the increasingly complex and large datasets generated in the modern information age. Theories and techniques developed around this approach have crystallized into a rapidly maturing field: scientific visualization. This new field represents a fundamental change in scientific research methodology, from the absolute dominance of analytical methods to the recognition of the power of human visual analysis. The emergence of scientific visualization reflects the inadequacy of analytical methods alone in coping with the massive amount of data generated by our perplexing modern age, advanced computing
technologies, and much improved data collection methods. For example, a sophisticated mathematical model of global environment changes, or a complex aerodynamic model of hurricanes could easily produce many gigabytes of results for subsequent analyses. Multispectrum remote sensed data have been accumulating for decades, waiting for more insightful analysis. Our daily life is constantly deluged with a barrage of information, ranging from urban crime to various financial data.

Often overwhelmed by the enormity and complexity of existing data, some scientists feel strongly that traditional analytical methods must be integrated with human vision to effectively process the information. We can no longer afford to rely on abstract mental thinking exclusively, while ignoring or underestimating our visual thinking (Maceachren, et al., 1990) capability. As a result, scientific visualization emerges as a powerful and promising methodology which values human vision not only as a primary visual data input source, but also as a one of the most fundamental data processing methods when applied in tandem with abstract mathematical/statistical approaches. Geographical analysis is among many fields which will be profoundly affected by this new paradigm of scientific inquiry.

Scientific Visualization and Cartographic Communication

Geography has a long tradition of using visual methods (cartography) to communicate spatial information. There exists a large body of literature aimed at searching for the most effective cartographic portrayal of geographic data, and the understanding of the human-map, interactive communication process. For example,
Dobson (1979) examined this process from the perspective of human visual information processing during map reading. In considering the map reader’s organic and cognitive activities, he proposed important concepts such as the fixation process, short term visual storage, feature extraction, visual conspicuousness and short term memory in explaining how map readers capture and store visual inputs, conceptualize and interpret mapped messages. Analysis of this interpretive process helps to understand human limitations and characteristics of visual information processing, and therefore should help us design maps which will maximally utilize human visual perceptual ability.

However, cartographic communication is quite different from scientific visualization. As pointed out by Maceachren and Ganter (1990), the former concerns communicating what is known, and the latter deals with revealing what is yet to be known. Cartography is oriented toward creating ideal representations of spatial information, while SV is preoccupied with an effective process of discovery. The purpose of SV is to provide the visual information necessary to discover the hidden messages the data contains, as opposed to cartographic communication, which has explicit messages to get across. "Therefore, there is no optimal map!" in SV, as Maceachren and Ganter have claimed (1990).

Scientific visualization represents a profound philosophical change of scientific paradigm. Traditionally, abstract analytical methods (mathematical or statistical) have been the predominant form of scientific inquiry, because it is generally believed to be objective and strict. Visual methods, either in the form of cartographic representation or statistical graphs, have been mainly used to communicate the results derived from
quantitative analyses or to present factual information for the purpose of illustration. Rarely have they been used as a formal approach in "serious" scientific analyses. The power of human vision in data processing had never been fully appreciated. Instead, it is warned against as "a potential source of bias." (MacEachren and Monmonier, 1992).

However, scientific visualization recognizes human vision as one of the most efficient ways to process complex information and as an invaluable instrument for rigorous scientific research. It can be used to extract regularities, rules, patterns, trends, and anomalies from overwhelmingly complex data so that we can generate hypotheses and conclude findings. Our visual thinking and reasoning complements, symmetrically, the traditional abstract thinking. The combination of these two is capable of forging an indispensable new weapon to cope with information explosion of the modern society. This becomes especially clear when we consider the astonishing development of modern dynamic, computer graphics, which enables us to manipulate huge amounts of data and generate complex graphics in real-time.

We actually use much more visual thinking than we normally realize. For example, mental maps have long been recognized by geographers as an important means of organizing spatial information. Human vision captures information, processes it and integrates it with mental abstract thinking. Visual images, structures are ubiquitous in mental data processing, no matter if the data concerned have realistic images or not. Visual representation of data in abstract mental thinking can be drawn from reality or simply assigned according to the data's characteristics. Scientific adventure has been fundamentally facilitated and augmented by visualization. Even mathematics have been
said to be the visualization of structures of numbers (Maceachren and Ganter, 1990). It is our daily experience that when a complex and abstract concept or process is understood, it is usually digested, or transformed in our mental abstract world into a some sort of visual images or structures so that it is more easily retained and manipulated. With images, either mental or realistic, we tend to organize information better, discover relationships and patterns faster, and reason more effectively.

Scientific visualization is a cognitive process, and an act of mental abstraction (Maceachren and Ganter, 1990), based upon which more abstraction can be done easily. A visual image somehow simplifies a complicated set of relationships, and frees us from laboriously generating them logically or from memory, so that we can proceed to our abstract thinking with less burden. Although scientific visualization is applied to a wide range of disciplines, it is especially useful in understanding spatial data, which inherently belongs to a higher order of complexity and has distinct visual structures and images.

Scientific Visualization of Spatial Data

Large spatial datasets contain information on complex spatial interactions, distributions, relationships and a temporal dimension in addition to the aspatial attributes associated with them. This data complexity makes scientific visualization an indispensable tool for understanding and analysis.

Although cartographic communication is quite different from scientific visualization, early works in analytical cartography already began to experiment with computer
animation and interactive exploration techniques in order to visualize complex, spatio-temporal data.

As early as 1959, Thrower saw the potential for animation in cartography to visualize temporal process and forecasted rapid growth in this area. In the early 1970’s, geographers were fascinated by the power of film animation, made using frames of computer generated images, to portray dynamic temporal processes. Tobler (1970) is such a pioneer who made a computer movie of the simulated population change of urban Detroit from 1910 to 1990, followed by Moellering’s work (1973) on displaying traffic crash patterns using the same technique. Nevertheless, Campbell and Egbert (1990) see too little effort devoted to developing the power of animated cartography over the past thirty years. They attribute the disappointing development to the lack of funding in geography, popular disdain of geographic academia toward research on potentially marketable software, and the lack of good dissemination means, such as PC’s and VCR’s.

The animated film, however, does not provide the human-display interactions needed for close examination of data. Therefore, geographers began to probe the possibility of interactively exploring complex spatio-temporal dynamics of geographic processes (Moellering, 1976, 1980a, 1980b). In his famous classification of various cartographic products ranging from tangible maps to digitally stored geographic data, Moellering (1980a) put them into four categories: Real Maps, Virtual Map-Type I, II, and III according to their characteristics of whether they are "permanent with tangible reality" and whether they are "directly viewable". He further observed that the efficient transformation between Virtual Map III (maps in digital forms) and I (maps on computer
screens) determines the "interactive and real-time display capability" of a digital cartographic system. Then he envisioned that powerful computers would manipulate complex geographic data directly and generate three-dimensional displays fast enough for the dynamic exploration of the spatial structure and temporal process. It is not coincidental that his idea of zooming in and moving around a cartographic surface at real time is now an important navigational technique for scientific visualization.

Although this line of work can be dated back to late 1950's, full-fledged pursuit of the scientific visualization of geographic data has been an event of only the past several years. A clear perception of its fundamental impact on geographic inquiry and its philosophical difference from traditional cartographic communication has also developed. Maceachren and Ganter eloquently argued (1990) that scientific visualization emphasizes the effective identification of spatial patterns unknown to us, not the presentation of the end results. Therefore, as they observed, scientific visualization itself is an exploration process, while traditional cartography focuses too much on "glamorous presentational systems". They further noted that the power of SV lies in the visual abstraction process, in which "simplification, approximation and abstraction" of geographic reality tend to effectively discern trends and regularities from chaotic data and achieve insight. With an emphasis upon appropriate abstraction instead of realism in portraying scientific data, the role of modern cartographic visualization, argued Maceachren and Ganter, should be modeled for the purpose of insight pursuit, a "Seeing-that, then Reasoning-why" interactive exploration process. As they observed:
... To do this, we must allow the scientist to interact with what is displayed in a meaningful way. It is not enough to provide a mechanism for allowing scientists to view objects in 3-D, spin them around, look inside of them, etc. The system should permit, indeed perhaps demand, that the user experience data in a variety of modes. The scientists can then iteratively search for, and look closer at, pieces of information...

High interaction visualization techniques or environment can be the key for a successful data exploration. Monmonier (1989a, 1990a, 1990b) explored the idea of geographic brushing and temporal brushing combined with scatterplot brushing for examining the spatial-temporal data. An ideal visualization environment should provide multiple views, control of time and space, and direct interaction with the displays (Monmonier, 1990b).

In the current pursuit of geographic visualization, animation continues to be a major technique for visualization of space and time (Monmonier, 1989b, 1992a, Dorling, 1992 and DiBiase et al., 1992). Monmonier (1989b) envisioned that we might be able to create some sort of graphic script to control the animated visualization process in the near future. Such a graphic script can be written in a macro programming language to create the best sequence of graphic displays of complex data. Using the script, an analyst may control the display sequence by specifying the content and changes on the computer screen, such as the number of variables, orientation, tilt, scale, duration, and manipulation functions like zooming, panning and animation. By doing so, the user may plan an elaborate animated display of perplexing data in his/her attempt to discover patterns and anomalies. To simplify the writing of sophisticated graphic scripts, standard operations can be organized into "graphic phrases" which function as building blocks, much like a
function library in any high level programming language. An expert-advisory system could be designed to help the user to write the most meaningful scripts to his/her particular exploratory intention by selecting variables, design scenes and operations. A user thus can preview a particular script-controlled animated presentation and then decide what to be regenerated or modified for closer examination.

If we can design such a powerful script language, it will allow us to generate very sophisticated animated displays of spatio-temporal data. In comparison to functions provided by a ordinary visualization system such as zoom or rotation, graphic script has the unique advantage of composing highly complex scenes and transitions by controlling many variables and functions at the same time in an elaborate way. This would open numerous opportunities for discovering hidden regularities and interesting deviations. However, so far we do not know how such graphic script can be designed and used, and there is no concrete experiment on this futuristic idea.

A similar idea on designing and controlling animation process was proposed by DiBiase et al. (1992). They defined three dynamic variables crucial for animation in scientific visualization: the duration of a scene, the rate of change between scenes and the order of scene presentation. Therefore, by appropriately fine tuning of these three dynamic variables, it should be possible to generate an animation process with the desired speed and smoothness and sequence. Although chronological order is most common, they argued that sometimes a particular sequence other than a chronological one might prove to be more revealing.
Another approach to visualization of spatial data is selecting and designing effective presentations of multivariate spatial data to exploit the user's visual interpretative capability and thus maximize the user's appreciation of the complex data (Robertson, 1988, 1990). He argued that a natural scene familiar to most people with its 3-D surface structure and other various physical properties such as covering material is the best model of data representation. The visual properties or dimensionalities of a natural scene can be matched with various variables of a complex dataset according to their shared characteristics, such as whether they are nominal, ordinal or ratio data. Such a paradigm is effective because we are accustomed to such natural data presentation, thus can process the information intuitively and efficiently.

To summarize, the recent growing interest in applying scientific visualization to geographic research has only scratched the surface of this potentially very powerful methodology in complex spatial data analysis. More work should be done to explore the theoretical dimensions of this method and to create effective visualization techniques and software systems to solve realistic geographic data analysis problems.

**Exploratory Data Analysis**

Exploratory data analysis (EDA) can be facilitated, and usually is carried out by means of scientific visualization. Like scientific visualization, exploratory data analysis is also a rapidly growing field in data analysis. Much of the body of EDA is attributed to Tukey (1977, 1980, 1988 and Cleveland eds., 1988), who, through his innovative thinking and experimenting, has laid a solid foundation for this discipline.
In short, EDA opens the dichotomy of data analysis, between confirmatory data analysis and exploratory data analysis (Fig. 1). It is a process of exploring, probing the unknown and comprehending, hypothesizing anomalies, relationships and patterns trapped in a large, complex dataset. The emphasis is placed upon data exploration, a navigational process without any preconceived guidelines or plans. The purpose is to set researchers free from any preconceptions so that with free will or whim, he/she can hover above or dive into or drive through the jungle of a complex dataset upon an exciting journey of discovery.

Tukey accurately described exploratory data analysis as "an altitude, a flexibility, and a reliance on display" (Tukey, 1980). It is based upon a willingness to examine the data open-mindedly so that we can obtain a feel for the complex data, and raise interesting questions, before we proceed to conventional, confirmatory analysis. He pointed out that the straight-line paradigm of data analysis as projected by "question --> design--> collection --> analysis --> answer" does not tell the whole story of scientific research, because it neglects the process of generating questions or hypotheses and of monitoring the design and investigation (Tukey, 1980).

EDA complements confirmatory data analysis by filling the vacuum of hypothesis generation through data exploration. It also helps to construct and monitor confirmatory analysis by feeding insights obtained through continuous data exploration. It is aimed at dealing with huge amount of data with considerable spatial-temporal complexity, in which we have absolutely no idea of what is going on. The only thing we can do is to break into the data, "flip through", and get a feel of it. In this initial process, we can explore
The Dichotomy of the Exploratory and Confirmatory Data Analyses

Confirm or Reject Hypotheses

Analysis Design

Statistical Analyses; Model Testing

Confirm or Reject Hypotheses

Complex Data, Curiosity

Reliance on Visual Tools

Free Style Exploration

Patterns, Relations, Anomalies

Confirmatory

Exploratory

Figure 1: The comparison and relationship between the exploratory data analysis and confirmatory data analysis.
general trends and local anomalies, and find interesting phenomena existing in the data. The result of the exploration is at least a partial understanding of the data so that hypotheses can be proposed.

Exploratory data analysis is a major revolution in the research methodology of science and engineering, and is "one of the greatest pendulum swings in statistics, away from formalized model testing towards an informal and open-minded approach to data analysis." (Haslett et al., 1990). Scientific visualization can facilitate EDA tremendously because it is a very effective means of processing complex information. These two concepts are associated so closely that some researchers simply regard exploratory data analysis as the purpose of scientific visualization (MacEachren and Monmonier, 1992) and SV as the principal means of EDA.

**Dynamic Graphics For Spatial Data Analysis**

Dynamic graphics is almost ubiquitous in scientific visualization and exploratory data analysis of spatial data. It provides high interaction between users and computer screen display and the underlying data set, instantaneous visual feedback (Becker, et al., 1987a), and the capability of user-controlled animated display. Modern high performance computing environments provide exciting opportunities for analyzing large datasets visually and dynamically. A data analyst can generate hundreds of frames of sophisticated graphics portraying data in a matter of minutes. Interactive manipulation of graphics and the underlying data sets triggers the complex eye-brain system which processes perplexing and abstract information at an amazing speed and efficiency. Using dynamic graphics can
help effectively discern patterns and irregularities from the chaos of complex data in an iterative process of linking between visual input and visual thinking.

Its ever-growing popularity as a powerful research tool also stems from the fact that high performance computers become increasingly available (at less cost and easier to use) with various interactive tools such as mouse, joystick, track ball, virtual reality gloves and eyeglasses.

Techniques of dynamic graphics for data analysis have come a long way, from the early systems such as PRIM-9 and ORION I to present systems like MacSpin, JMP, EXPLOR4, and SPIDER. Modern dynamic graphics software is widely available on microcomputer or workstation platforms with impressive speed and performance.

Becker et al. (1987a) described some commonly used dynamic graphics techniques for data analysis, from simple identification, deletion, scaling to relatively complex rotation, linking and scatterplot brushing, which are briefly summarized below.

Identification

An interactive technique to identify a particular one or a set of data points, or to separate a subgroup of data points from the rest by highlighting them. This is done through labeling and locating. The former inserts a label that indicates the nature of the selected data point, while the latter highlights all data points with the same label. This is a basic yet very useful technique.
Deletion

Deleting some data points can be an effective tool for discriminant examination. For example, deleting the main cluster of a scatterplot allows close scrutiny of the outliers. On the other hand, deleting outliers helps to present a sharp image of the cluster. Fig. 2 illustrates the improvement in resolution due to deleting a single outlier.

Linking:

Linking corresponding data points in different scatterplots reveals interesting patterns that are often undiscernible otherwise. These linkages can be established by visually relating two or more scatterplots or by dynamically highlighting points using techniques such as scatterplot brushing.

Scaling:

A technique for controlling the aspect ratio of height vs. width of a two dimensional graph. Changing the aspect ratio may yield a surprisingly revealing graph, because the exaggeration or suppression of the vertical or horizontal axis affects the appearance of the graph, a function similar to smoothing or magnification of curves. Altering the aspect ratio dynamically creates a gradual and smooth change of the look of the graph, which helps to "visually decode the quantitative information" (Becker et al., 1987a).
Figure 2: Deletion Operation. Deleting the outlier from the top-right corner, and rescaling the graph produce the bottom graph with very high resolution. [Source: Becker et al. 1987a]
Rotation:

Rotation of point clouds takes place in three dimensional space around a selected axis. Smooth motion can enhance the analysts’ feel of depth and improve his/her comprehension of the 3-D structure of the point clouds. Rotation can be done around any of three coordinate axes, an arbitrarily defined axis, or three fixed screen axes. Becker et al. (1987a) also note that a point cloud does not have a very vivid, intuitive real world 3-D depth, because "real objects are composed of surfaces, not points." Consequently, they suggest that it would be useful to create a stereo vision by superimposing two slightly offset views of different colors for the same point cloud and viewing them using correspondingly colored eyeglasses.

Scatterplot Brushing:

Scatterplot brushing is perhaps one of the most significant innovations in dynamic graphics for data analysis in the past two decades. Regrettably, such a wonderful idea does not occur often enough. If it does occur, it usually has a remarkable impact on the field.

Becker and Cleveland, in a series of papers, describe clearly their idea and implementation of this high-interaction, dynamic, graphical method (Becker and Cleveland, 1984, 1987a, 1987b). It is simply a way to examine multi-dimensional data by dynamically linking many pairwise scatterplots. A rectangular brush is used to move around the scatterplots as a highlighting device. The data points inside the brush are linked visually with corresponding points in all other scatterplots. Four basic brushing
operations were implemented: *highlighting, shadow highlighting, deleting, and labeling*, each of which can be used in one of the three paint modes: *transient, lasting, and undo*.

Figure 3 demonstrates the highlight brushing operation. A scatterplot matrix is used to exhaust all possible pairwise scatterplots from among all variables. The variable "Hardness" controls the vertical axis of the first row and the horizontal axis of the first column. "Tensile Strength" does the same for the second row and column, and so on and so forth. As an analyst moves the brush around the scatterplots, points inside the brush are highlighted, so are the corresponding points in other scatterplots. This establishes instantaneous visual linkage among data clusters or points.

If brushing is in transient paint mode, some data points originally inside the brush are no longer highlighted as the brush moves away from them, while new data points entering the brush are highlighted. The same thing happens to the rest of the scatterplot matrix. If brushing is done in lasting mode, the formerly highlighted points remain so as new data points are highlighted. Undo mode would remove the highlighting as the brush moves along.

Changing the shape of the rectangular brush may be used to inspect the dependency of variables. This can be done in two ways: conditioning on either one or two variables. If a long brush is used to highlight a certain range of a variable in the active panel, then we can see the dependency of other variables upon this particular range (Fig. 4). Similarly, a square brush can fix ranges for two variables in the active panel, the analyst can inspect the dependency on this range by looking at the distribution of the highlighted points in all other scatterplots (Fig. 5).
Figure 3: Highlighting operation of scatterplot brushing. The data points corresponding to the points inside the brush are highlighted as black dots. [Source: Becker et al., 1987b]

Figure 4: Scatterplot brushing conditional on a single variable: the lower range of hardness is fixed to see the behaviour of other variables. [Source: Becker et al., 1987b]
Figure 5: Shadow highlight operation. Only the data points corresponding to the brushed points are displayed in the nonactive panels as black dots. [Source: Becker at al., 1987b]
Shadow highlight deletes the data points which are not highlighted in the nonactive panels to increase the clarity of the graphs (Fig. 5). Deleting operation removes the enclosed points, while the labeling operation prints labels for them.

Becker and Cleveland compare rotation with bushing (1987a) and conclude that while they are both valuable, analytical tools of dynamic graphics, they are most useful under different circumstances. Rotation is good to discover 3-D clustering, which may be extremely difficult to identify in brushing. Nevertheless, the perceptual difficulty increases dramatically as the dimension of data moves beyond 3-D to 4-D and higher. Brushing can handle a large number of variables and portray pairwise dependencies succinctly. Rotation also has the drawback of sometimes insufficient 3-D feel since a point cloud does not have clear surfaces and hence does not possess vivid feeling of depth.

Based on the notion of scatterplot brushing, Monmonier (1989a, 1990b) proposed two concepts to extend this technique to spatiotemporal data analysis: geographic and temporal brushing. By adding a geographic map to the scatterplots and highlighting corresponding polygons while the brush is moving inside the scatterplots, analysts now can examine geographic correlation (Monmonier, 1989a). This has been realized successfully in Tang's dissertation work (1993). Monmonier also suggested to append a time scrollbar so that the temporal change can be also reflected (1990b).

These references are intended to provide only a glimpse of the wide array of dynamic graphics techniques used to visualize and explore multi-dimensional data. There are many other techniques that are less relevant to the flow visualization but nonetheless quite interesting, such as kinematic display (Donoho, et al., 1982), hyperspace interpo-
lation and hyperspace residualization (Young, et al., 1988), viewing pipelines (Buja, et al., 1988), depth separation (Papathomas et al., 1987), to mention but a few. The spectacular growth in this field, fueled by the availability of modern high performance computing environments, will undoubtedly produce a fundamental impact on how scientists look and analyze large amounts of observational and model-derived data.

**Selected Dynamic Graphics Systems for SV and EDA**

Since early 1970’s, many innovative dynamic graphics systems have been built with various functions. Most old systems were constructed on highly specialized, expensive hardware platforms due to the then limited computer performance, such as Varian 620/i and IMB 360/9 used for PRIM-9 (Fisherkeller, et al., 1988), while modern dynamic graphics systems are mostly based on popular yet high-powered platforms, such as high-end PCs or Macintosh, or Unix workstations. This section briefly reviews some of these systems and the visualization techniques utilized.

**An Early System: PRIM-9**

PRIM-9, the most influential early system was developed by a group of innovative pioneers of dynamic graphics and EDA (John Tukey, J. Friedman and M. A. Fisherkeller) in the early 1970’s at the Stanford Linear Accelerator Center. It utilizes several dynamic graphics techniques to manipulate data of up to 9 dimensions: rotation, picturing, projecting, isolating, and masking (Fisherkeller, et al., 1988). By applying one or combinations of these functions to the nine variable data space interactively and through
continuous motion, data analysts are able to view the perplexing data from different angles, dissect a certain subsample of data points from the 9-D space for closer examination, isolate important structures, and pursue the projections which best illustrate the inherent characteristics and complex nature of the data. Here is a brief summary of the major functions of PRIM-9 as implemented by Fisherkeller, Friedman and Tukey.

Picturing

The capability to plot a two dimensional scatterplot portraying any two of the nine variables. By assigning a variable to the vertical axis, and another to the horizontal axis with a push of a button, the data analyst can generate 36 (combinations of 2 from among 9 variables) unique pairwise scatterplots pertaining to different aspects of the data set in two dimensional space.

Rotation

To change the orientation of data presentation by rotating continuously around a fixed axis in the 9 dimensional space. Ideally, the analysts should be able to define an arbitrary axis to rotate, but this is too complicated computationally and very cumbersome for the analyst to operate. Instead, the rotation axis is specified using 2 coordinate axes which actually participate the rotation, which is analogous to common 3-D space rotation around X, Y, or Z axis. Suppose X is the rotation axis, then Y and Z are the participating axes which will be actually moving continuously in the operation. Therefore in the abstract 9-D space, specifying the two "participating" axes is an easy way to define a
rotation axis in an otherwise incomprehensible situation. Rotation at an accelerated speed was accomplished by pressing a button and could be reversed at a slow speed to locate precisely the desired projection position once an interesting pattern was perceived.

**Masking**

To find and define a subregion in the 9-D space, and focus on examination of the enclosed data points by eliminating the rest from the computer screen. The mask is specified by the coordinates it is associated with, its dimensions (front edge F, back edge B, and Joint J) and can be driven back and forth along a constant direction for desired locations.

**Isolation**

To dissect a subsample of data points from the total population, and then focus on this subset of data for indepth scrutiny. It is usually used in conjunction with masking, which progressively removes unwanted samples until only the targeted samples are left. Then the samples are successfully isolated.

The difference between masking and isolation is that masking is defined with respect to coordinates, while isolation relates to a specific subset of the data. The data points in a mask change as rotation moves them in and out of the mask, while they remain unchanged in isolation.
Automatic projection pursuit

The capability to search automatically through all possible 2-D projections of the multidimensional data and find one that contains the maximum number of interesting structures. This search can start from the current projection on the screen or from any chosen projection. Pursuit is performed by fixing one projection axis and varying the other until a local maximum is found. Then the second axis is fixed and search is performed on the first axis. This process repeats until an interesting projection is found or convergence is reached. This is the most interesting function in PRIM-9, allowing a systematic, iterative and gradual close-in search for patterns and clusters.

The exploratory power of the system should be exploited by painstakingly preconceived sequences of operations by combining different functions, from random viewing for the slightest signs of interesting clusters to effective close-in to isolate them through progressive removal of irrelevant data, from a vague image to a sharply, vividly presented structure, from chaos to discoveries, all through smooth, interactive manipulation of dynamic graphics. This is the true essence of exploratory data analysis.

ORION I

ORION I is the direct descendent of PRIM-9, among others (PRIM-H by Harvard University, PRIM-S by Swiss Federal Institute of Technology, all in late 1970's) (MacDonald, 1988). MacDonald (1988) provides a clear description of ORION I. It used much more advanced hardware and some improved EDA techniques. PRIM-9 utilized a IBM 360/91 mainframe computer, a Varian minicomputer for data transfer, and a IDIOM
vector display. Orion I, on the other hand, was implemented on a fast graphics workstation with a high resolution display.

Three techniques were developed: high-dimensional views, projection pursuit and multiple views. By adding color, shape and size to the data point portrayal, it could represent dimensions far more readily than the featureless data point method (a plain dot). Multiple views provide simultaneous, connected pictorial portrayals of different aspects of the data. Projection pursuit, manual or automatic, was also implemented to reduce the complexity of high dimensional data by projecting to lower dimensions (<=3) optimally so that the hidden structures are most vulnerably exposed in a space within which human vision can manage easily.

**SPIDER: Spatial Interactive Data Explorer**

This is an EDA software system implemented on a MacII with very nice graphical user interface (Haslett et al., 1990). The emphasis was on creating dynamic linkages between geographic views and statistical views. Spider has a quite powerful overlay function through its GUI shell, which can put data layers such as satellite imagery, point and line data together to generate a sophisticated map view. Statistical graphs, including scatterplots and histograms, can be also generated interactively to portray relevant variables such as mineral chemical elements available the mapped region.

By selecting a certain portion of the histogram or scatterplots, geographic correlation can be examined with the corresponding highlighting on the map view. The
dynamic linkages can help to verify or refute theories by separating and focusing various subgroups of samples relating to different portion of the histograms.

MacSpin

This is another Macintosh-based implementation with an emphasis on viewing three dimensional data through continuous rotation (Donoho et al., 1988). 3-D images of point clouds can be best viewed through continuous motion in order to generate a feeling of depth since they do not inherently possess a vivid 3-D visual appearance.

Rotation can be operated on any axis in counterclockwise or clockwise direction in an animated process. A data point can be identified with a label by clicking on it. Data points also can be separated by predefined categories interactively for close inspection (Fig. 6).

MacSpin provides very comprehensive data transformation functions, which include all arithmetic and trigonometrical operations. It also provides very nice supporting functions such as a user-friendly data input system and graphical hardcopy printing system. The latter is invaluable to users who wish to retain the image of the screen for subsequent analysis or documentation.

One very obvious drawback of MacSpin is its lack of a interactive point cloud identification and separation function on the screen using a pointing device. Data points can only be separated according to predefined categories. And it basically is a visualization system, analytically insufficient.
Figure 6: MacSpin. Rotation of a point cloud (top) and separation of clusters according to predefined categories (bottom). [Source: Donoho et al., 1988]
Conclusion

Dynamic graphics has been recognized as a fundamental tool for scientific visualization and exploratory data analysis. The ability to manipulate computer graphics and underlying data directly and instantly provides numerous possibilities of visual and dynamic analysis of large and complex data: multiple views, visual linkages, data dissection, isolation, generation of optimal projections, ..., this list can go on and on. With the ever-accelerating advances in modern computer science and technologies, scientists are empowered with more powerful and faster computers at less cost. What seems today impossible or too expensive hardware- or software-wise may turn out to be entirely feasible very shortly. Scientists cannot overestimate the future development of dynamic graphics. The aforementioned dynamic graphics systems are only a selected few, for the purpose of demonstrating the enormous potential of this new technique and its far-reaching impact on data analysis. Spatial data, with clear visual structures and immense complexity, is a natural candidate for utilizing such a powerful tool.

Combining SV, EDA and Dynamic Graphics
for Analyzing a Large, Complex Flow Dataset

It is very clear that SV, EDA and dynamic graphics are so closely intermingled conceptually and methodologically that they must be considered as a single inseparable tool box in modern data analysis. EDA has a great reliance on visual methods to sift through huge amount of data, to observe what is underneath the surface, and to discover the unexpected. SV explores the enormous power of human visual thinking to perceive
structures, patterns and anomalies from chaos. Dynamic graphics, with high interaction and instantaneous visual feedback, is the best vehicle to effectively visualize perplexing data. The fundamental purpose of SV and dynamic graphics is not to find the best way to present known information, but to provide a mechanism by which the hidden structures are most vulnerably exposed and therefore discovered. There is no better combination of research methodologies.

Combining EDA, SV and dynamic graphics provides an ideal tool for tackling the complexity of large spatial temporal flow data. It should be realized that putting all spatial flow information on a single map is usually neither necessary nor desirable. It is not necessary because a user usually is interested only in certain portions of the information but seldom all of the information at a single time. For instance, a data analyst may only want to investigate all flows into a region larger (or less) than a certain threshold. It is not desirable to deal with all the data because our mind and our vision can only process a certain amount of information effectively and will be overwhelmed and confused by a quantity of information that exceeds that capacity. It is also impossible because of the staggering volume of the flow information. Therefore, the single-minded pursuit of a "perfect" cartographic method of flow portrayal will not yield satisfactory answer to the problem of analyzing and comprehending a large flow dataset, not merely because of the cartographic difficulties, but more fundamentally because we do not know exactly what to present.

The solution would appear to be exploratory data analysis aided by scientific visualization and dynamic graphics, since the user of the EDA tool is left to decide what
to be presented. Dynamic graphics can generate displays with varying information content easily and in near real-time. Therefore, it is possible that we can achieve comprehensive understanding by putting together many "pictures" from various perspectives.

Creating an interactive graphics environment is vital to deal with the complexity of spatial flow data. A computer can store great quantities of information. However, the graphics on the screen have the advantage of displaying only a portion of the data. This at least gives us a way to reduce the complexity to a manageable level. Complexity of the graphics can be interactively controlled by an exploring scientist such that his/her data processing capability is optimally utilized. In such an environment, information can be extracted visually and dynamically. The whole dataset can be understood by viewing it in different aspects within his/her capacity of processing information. The data analyst is granted the freedom of choosing information and determining interactively his saturation level with respect to the graphics. Moreover, the dynamic graphics provided in such environment offers great intuitive power to relate, to link and to explore potential regularities and anomalies in the data by manipulating the graphics directly and getting instantaneous feedback.

**Computer Cartographic Techniques of Mapping Spatial Flow Data**

Using lines or bands (with arrow or without) to map spatial flow data is not a new technique. Tufte (1983) traced the flow mapping back to the classic of Charles Joseph Minard (drawn in 1861), which depicts the process of Napoleon's invasion to Russia and the subsequent withdrawal from Moscow during 1812-1813. Tufte (1983) gives a very
enthusiastic interpretation of the map. This is indeed a very nicely designed flow map, conveying considerable amount of information in a yet quite simple, clean fashion. The map not only shows the flows (movements of the army), but also displays other data such as temperature, and important geographic features. The magnitude of the flows is represented by both the width of the bands and some strategically placed numbers. Therefore the reader can see the relative sizes of the flows and the absolute values. Arrows are not used in the map because the map only displays two flows (movements into and away from Moscow). Instead, the shading is used to differentiate them. The geographic map is somewhat distorted for the convenience of portraying flows at a better clarity. However, the flow data presented is very small in comparison to the size of the data dealt in this research. In another track, Tobler (1987) also traced flow mapping back to a similar period (1885) when Ravenstein employed a flow map to show the currents of migration.

However, computer graphics utilizing arrowed lines and bands are events of recent decades. An extensive search for works of this nature has been conducted by the author in the literature of cartography, analytical geography, fluid dynamics, computer graphics applications, transportation research, and migration studies. It was found that the number of works specifically dealing with mapping spatial flow data was very limited. Tobler, however, made substantial contributions by experimenting with different techniques of migration mapping (1987), and Becker and his AT&T Bell Lab colleagues (1990a, 1990b, 1991a, 1991b) applied, for the first time, dynamic graphics to visualize flow data.
Regrettably, the accomplishments of other works are marginal. In this section, some selected pieces of work will be reviewed in detail to provide a broad perspective.

Using Lines to Represent Spatial Interaction

Kern and Rushton (1969) seems to be among the early users of computer graphics to show spatial movement. In an effort to produce flow maps, the numbers of customers to beauty-care centers were mapped by connecting the locations of the customers and the centers using single lines without arrows. The technique is very simplistic, but it does show clear patterns of spatial interaction.

Constant Band Width with Variable Shadings in the Corridor

A more mature treatment of spatial flow mapping can be found in Wittick’s work of 1976. He developed a computer system for transportation data mapping and analysis. In dealing with the magnitude of transportation flows, two options were adopted. The first was to use constant band widths with variable shading according to the volume of flow. The shading scheme required flows to be divided into intervals, which introduces increased aggregation into the data. However, Wittick preferred this technique than the band width proportional to volume because he believed the former could better reduce the clutter of the map (Fig. 7).

A second option was to use single lines with numerical labels to identify the actual flow volume (Fig. 8). Furthermore, the nodes could be labeled by using names, a numerical sequence or graphic symbols.
Figure 7: Graduated shading in the corridors of constant bandwidth to display flows. [Source: Wittick, 1976]

Figure 8: Lines and numbers representing magnitudes of flows. [Source: Wittick, 1976]
However, in both cases the flow direction was not treated, and bands connected two nodes without arrows or any other directional indicator. Although Figure 7 and 8 look very neat, they definitely will become much more cluttered when the number of nodes to be portrayed increases.

**Vector Accumulation Program for Urban Transportation Pattern**

Noguchi (1977) developed three computer graphics programs for displaying urban transportation data: CENVUE(S), VAP (Vector Accumulation Program) and TDN (Time Distance Network).

CENVUE(S) is a graphics program which can generate 3-D perspective maps, with the data being viewed as stacks. The program shows the physical route of urban transportation and uses the height of stacks to show the actual magnitude of traffic indicators (average number of passengers on the buses, night time passengers, average speed of buses, etc.) at various locations along the route.

TND can produce the urban transportation network according to the actual travel time on the network, instead of using actual physical distance. The result is a distorted form of the original physical network. The TDN is overlayed with the actual network to compare the physical distance and the actual travel time distance.

Most relevant to our interests is the VAP (Vector Accumulation Program) which is used to portray traffic flow patterns. VAP generates the route of travel between an origin and a destination by looking for the shortest distance on an imaginary grid system substituting for the actual network. This imaginary grid is constructed on the urban area
so that shortest paths can be approximated by tracing the grid. A line can pass through a grid cell by tracing its edges or going diagonally (at 45 degrees). When the lines overlap with each other, the computer accumulates the vectors (Fig. 9). Then a color or line weight scheme is made based on the vector accumulation records. Finally, the traffic pattern is plotted according to this scheme. Figure 10 illustrates this technique. The first four graphs represent four levels of trip densities, each plotted using different line weight according to its density level. The fifth is produced by plotting the four levels at the same time to generate an overview of the traffic pattern.

This is very important work which significantly reduces the complexity of spatial flows while clearly presenting the spatial pattern. However, it ignores the actual
Figure 10: VAP display of traffic pattern. The first four graphs represent four levels of trip densities, each plotted using a different line weight according to its density level. The fifth, on the lower right, is produced by plotting the four levels at the same time to generate the overview of the traffic pattern. [Source: Noguchi, 1977]
transportation network, and trips from a particular origin to a destination cannot be determined from the plot when multiple origins and destinations are involved.

Continuous Flow Model

Tobler (1981) developed two mathematical models for geographical movement: a network model and a continuous model. The network model is discrete while the other is continuous. In the discrete model, interaction occurs between any two regions. In a continuous model, however, the interaction of a region with others is restricted to only its immediate neighboring regions.

Tobler argued that the interactions among regions other than neighbors were "an artifact of data collection procedure" (1981). If the movement occurs on the surface, he contends that the move from Maine to California is actually a sequence of movements between neighboring states. Therefore, adopting this approach would greatly reduce the number of entries in the N by N matrix of geographical movement, since the entries between regions other than neighbors would be zero. Thus, he suggests that the original N by N matrix is simplified substantially by parsing all movement into movements between neighboring states. The continuous model was used to estimate the actual interactions among neighboring states and then mapped (Fig. 11).

This is an excellent method for complexity reduction of flow data. Maps produced using this model have much better clarity. However, as strong as Tobler's argument is, the fact that geographical movement has a clear, final destination is not satisfactorily reflected in his model.
Figure 11: When the number of regions $N$ approaches infinity, spatial flows become a continuous vector field. The map on the top shows a flow field generated using this method. The map on the bottom shows migrations rerouted to go through the nearest neighboring state. This method significantly reduces the number of flows needed to be displayed. Only the top 50% flows are displayed here. [Source: Tobler, 1981]
Experiments with Graphical Display Methods and Complexity Reduction

Tobler (1987) conducted an extensive program of experiments on mapping migration data. The experiments focused on two aspects: (1) different graphic display methods; (2) reducing the complexity of the migration matrix. These are summarized below.

Graphical display methods

For non-directional flows such as total flow between two places, Tobler (1987) suggested following symbols for showing the interaction:

1. Width of a flows band increases with the volume, and linking the origin and destination. As for the scale of the width to volume, he argued that there is no standard and that the default is to set the width of the largest flow the same as the distance of the two closest centroids.

2. Constant band width with graduated intensity shading corresponding to the volume.

3. Shading multiplied by area of the band proportional to the volume. The underlying assumption is that visual intensity is linearly associated with the combination of these two factors.

4. The total area of the band corresponding to the volume. Again, his assumption is that visual intensity corresponds to size of the area.

5. Other non-linear relation with magnitude, such as logarithmic.
After experimenting on all the above, Tobler felt that the proportional width is a better choice because this representation is most visually intuitive.

For directional flows associated with specific pairs of origins and destinations, arrows are needed for proper representation. Tobler listed many options for the arrow forms and shading patterns (Fig. 12). The arrows can be the classical or barbed ones. The shading, in Tobler's program, includes four patterns: lines parallel to the band, or perpendicular to the band, cross-hatched or chevron. Alternatively, the edges of the arrows can be omitted to have a fancier look.

In order to display two-directional flow on the same path, Tobler experimented with 3 options: putting the smaller arrow on the top of bigger one, or half-barbed arrows with or without gap in between (Fig. 13). Nevertheless, none of them were proven to be very satisfactory.

Tobler also proposed other forms of arrows. They could be "circular, elliptical, splined flow bands", or arrows flying "through the air", etc. But they were not implemented in Tobler's experiments.

**Complexity reduction methods**

The second aspect of Tobler's experiment (1987) concerns how to reduce the complexity of the movement table. The following are a list of what he suggested:

1. Overlap deletion. When arrows or bands overlap, the hidden band will be removed. Tobler recommended that putting bigger bands on top of the
Figure 12: Arrows with different shading patterns. [Source: Tobler, 1987]

Figure 13: Different two directional arrows. [Source: Tobler, 1987]
smaller ones because this reduces the number of arrows present and emphasizes the important flows.

2. Reducing the complexity by showing only the flows from a region or into the region, or only the net movement.

3. Deleting flows below some threshold. Tobler believed that the data of the movement table roughly follows a Pareto distribution: a lot of small numbers and only several large numbers. The average flow volume was considered as an ideal threshold since experiments showed that at least 75% of the entries could be removed.

4. Collapsing the movement matrix by aggregating regions into bigger ones. This is achieved "at the expense of spatial resolution". However, Tobler suggested that "regional variance of resolution" can be reduced in the case of U.S. data if eastern states are aggregated more than western states, since the former have much smaller sizes of states.

5. Using the continuous model (discussed in last section) to reduce the number of entries by restricting movement only among neighboring states.

**Suggested further research**

Based on his experiments, Tobler suggested several area for further investigation:

2. Mapping comparisons of two or more matrices of spatial movement from different time periods.

3. Experimenting animation displays of large movement data of a considerable time span to show the temporal process, especially if the movement is registered in a very short time interval in the data.

4. Breaking down the migration table according to "age, sex or other characteristics" and mapping the movement for comparison.

Tobler's work is the most comprehensive so far. Together with Noguchi's VAP technique, it represents the state-of-the-art of displaying spatial flow data in terms of representation methods. However, these methods do not realistically solve the problem of large, complex flow data. They can only improve the clarity of the flow maps to certain extent. No matter how good the representation methods are, when faced with the staggering volume of flow data, they become ineffective because maps still get cluttered.

A User-Friendly Package for Displaying and Analysis of Flow Data

Teeffleen's work (1990) is another recent attempt at flow mapping. The work is aimed at building an efficient and user-friendly, interactive software system to display and analyze spatial flow data for GIS professionals in developing countries, who only have access to inexpensive hardware and software. Therefore, the program is designed to work on IBM XT or AT machines. Atlas*Graphics and Atlas*GIS are used as important extensions of the core program.
The software allows users to select regions among which interactions occur. Flow data then can be displayed using colors in a simple non-directional manner with the possibility of using bandwidth to differentiate flow magnitude. It also provides 2-D frequency distribution graphics.

This is an interesting attempt to build an "interactive environment" for flow mapping. Unfortunately, it lacks sufficient theoretical guidance and is quite crude in terms of cartographic techniques and software quality. It is far from being interactive, since the main program can only display lines with colors but not width. Therefore, the map files have to be exported to Atlas*Graphics or Atlas*GIS to display in variable width. This makes the process time-consuming and non-interactive (in the sense of immediate feedback). Only non-directional flows could be displayed. In addition, the package provides no other display options for the users other than what has mentioned above.

**Interactive Graphics for Network Flow**

The most important development of visualizing flow datasets using dynamic graphics has been the work of Becker and his Bell Lab colleagues (1990a, 1990b, 1991a, 1991b). In four articles, Becker et al. discussed techniques for the visualization of network flow data, namely telephone call data using dynamic graphics. This effort was subsequently developed into a software called SeeNet.

Becker et al. pointed out that dynamic graphics provides a highly interactive environment, therefore real-time manipulation of data can help to comprehend the network flow data. A concept called "parameter focusing" is proposed, which treats a graphical
display as being controlled by a set of parameters, and thus fine tuning of these parameters by users interactively could give them the desired graphic information in a clean presentation (1990a).

The following are a list of parameters which Becker et al. (1990a) suggested to be used by a user to control the graphic display of network flows:

1. Statistics: The users should be able to choose appropriate information to be presented. The program permits easy selection of data, for instance, links and nodes and their attributes can be interactively selected and displayed. It also provides a set of numerical transformation for data manipulation, such as square root, logarithm, and so forth.

2. Levels: This is a parameter which allows interactive specification of the range of data to be presented.

3. Area: This permits users to restrict displays to a particular geographic area of the network by a zoom-in function or selection of nodes by activating and deactivating nodes.

4. Time: Dynamic selection of time by moving a line in a time scrollbar. And a animated play of node info (continuously varying the dimensions of the in-calls, out-calls boxes) through time. The display speed can be controlled.

5. Aggregation: Allowing users "to aggregate statistics over geographic regions or logical subsets of a network," (Becker et al., 1990a).
6. **Size**: Interactively changing the size (width, length, etc.) of display symbols to maximize presentation clarity.

Two modules were developed in the computer implementation, which is an interactive graphics environment. The first module displays network flows, and the second module presents node information (such as total in-calls and out-calls of each node). Users can interactively manipulate the graphics and fine-tune it to extract desired network flow information. Fig. 14 is an example of such displays.

Becker and his colleagues' work represents the first attempt to use dynamic graphics to visualize network flow data. Users now are provided with tools to manipulate displays directly and extract information at real time. It is a relatively powerful analytical tool for visualizing flows and statistics of nodes involved. This work is very significant because no one has ever done this before.

There are, however, several obvious drawbacks which merit much improvement. First, it has poor cartographic representation of flows. Different colors are used presumably to help convey flow magnitude information, but the color scheme (dark gray -> dark green -> light green -> orange -> red, representing from light to heavy) is poorly designed and does not convey a clear feeling for increments in magnitude. There is no symbol to express directions, which eliminates the possibility to display directional flows (origin -> destination flows, net flows, two way flows). And only one display method exists: color plus width corresponding to magnitude, which omits the flexibility of using other schemes (line and number, color shading). The overall quality of displays is not visually appealing. Second, an interesting option was mentioned but not implemented:
Figure 14: Display flows using SeeNet. This graphics shows overflowing calls for selected nodes. Color and line width are used to show magnitudes. [Source, Becker et al., 1991b]
data aggregation according to geographic area. Third, although the research is targeted at network visualization, it actually does not include any specific network. Links are simply straight lines connecting nodes. How realistic network flows can be visualized is yet to be done in the future.

We might categorize the type of spatial flows like inter-city calls as "non-route based" network. That is, the links of the network are simply hypothetical straight lines linking the network nodes. This is only a portion of the spatial flow data in the theoretical framework presented in Fig. 15. There are three distinct categories: network flow data, non-network flow data; and random dynamics data.

Network flow data is concerned with flows in a network composed of nodes and links. It can be further divided into route-specific and non-route network data. The former relates to flows on a clearly-specified network such as a transportation network, where flows must occur over the routes. The latter ignores the actual routes with only nodes and direct geometric links left.

The inter-regional flows (migration, commodity flows) dealt with in this dissertation represents "area-based flow data", and are quite different from non-route network data. First, actual routes of flows are not clear or not very important in comparison to network data. Second, spatially, all regions make up a seamless geographic area such as a country, while the network which does not occupy a whole region but rather represents spatially discrete points and lines. Area-based flow data represents characteristics of a region as a whole; while network node data only represents a geometric point, although this point has size in reality (such as a city). The spatial
Figure 15: Theoretical Framework of Spatial Flow Data Types and Research Directions. The shaded area represents the data domain and analytical approaches of the dissertation work.
aggregation of area-based data represents a clearly-defined larger region, while that of network only represents a collection of points, which actually does not cover the whole region.

Summary

Scientific visualization and exploratory data analysis emerge as a very powerful methodology to analyze large, complex datasets. This provides direct reference to the spatial flow problem. It can serve as a set of general guidelines or principles that could be applied to our situation, because spatial flows are just another case of large datasets which could be adequately dealt with using this approach. Although this new methodology has attracted much attention from geographers in the recent decade, there do not exist works specifically dealing with flow data using this new approach in the geographic literature.

Previous attempts on the problem using cartographic methods remain largely experimental and contribute only marginally to the solution, because improvement of flow representations can only increase the map clarity to a very limited extent, but still cannot cope with the sheer size of the data. On the other hand, the work using interactive graphics by Becker et al. lacks more mature and comprehensive visualization tools and systematic theoretical guidance. They also suffer from lack of sensible cartographic techniques because they are not geographers but dealing with fundamentally spatial data.

Therefore, the present study should draw valuable experiences from the previous works, and extend far beyond their limit by providing a systematic treatment of the problem from the perspective of SV and EDA, and by incorporating innovative, effective
visualization techniques with good cartographic designs to explore the complexity of large spatial flow data.
CHAPTER III
AN APPROACH TO VISUALIZING AND EXPLORING SPATIAL FLOWS

After an extensive examination of the literature involving spatial flow representations and analyses, it is clear that the complex problem of understanding and analyzing large spatial flows could not be satisfactorily dealt with using a traditional cartographic approach. Many years of research indicate that the pursuit of an ideal cartographic display strategy of complex spatial flows is futile mainly because the data is too large. This calls for a different strategy of attack: building a dynamic visualization environment to interactively and visually explore the intricacies of spatial flows, so that the enormity of a flow dataset can be handled by methods such as dissection, extraction, searching, linking and navigation, utilizing the immense capacities of human information processing.

This chapter provides a systematic and indepth discussion of the methodologies proposed to tackle this problem. They are effective analytical strategies by combining scientific visualization, exploratory data analysis and dynamic graphics.

It is worth noting that although most of the features mentioned here have been implemented in the flow visualization system, this chapter only describes an ideal system and visualization techniques, which may not match precisely with the actual implementation.
Building a Dynamic, Exploratory Visualization Environment

Presenting the entirety of complex spatial flows without overwhelming and confusing the user appears to be an unachievable objective. However, if we consider the realistic situations of a spatial analyst who wishes to study the spatial process of geographic movement, we will soon realize such an objective is really not necessary, nor intelligent.

Normally the analyst may not be so ambitious as to attempt to understand the entire flow patterns at once. Most likely, he/she would choose to investigate different aspects of the data and at different levels of detail one after another. Consequently, a comprehensive understanding of the process can be gradually achieved by putting all the pieces of information together.

Furthermore, often the analyst may be interested in only some specific aspects of the flow data, for instance, the spatial flows lying within a certain threshold level. In this case, displaying all the flow information is obviously not necessary. What is more important is to provide the necessary tools by which the required information can be extracted efficiently.

Thirdly, an even more likely scenario would be an analyst attempting to tackle a large, complex flow dataset which he/she know very little about. In the preliminary phase of data analysis, what is desperately needed is a tool to navigate through the data and explore its complexity, and try to make some sense of it. Such an exploration may yield interesting discoveries of patterns or relationships, which can stimulate further probing and eventually may lead to the generation of hypotheses.
Figure 16: Strategies for understanding and analyzing complex spatial flow datasets by combining exploratory data analysis, scientific visualization and dynamic graphics to build a flow visualization environment.
Therefore, an dynamic visualization environment is essential to understand and analyze a large, complex spatial-temporal flow dataset (Fig. 16). This requires a set of highly comprehensive, flexible, intuitive visualization tools that can encourage an analyst to explore the dataset and extract desired information easily and efficiently. Within this environment, the analyst can extract, manipulate, shift through massive amounts of information effortlessly and reveal insights into the data. He/she can navigate through, fly over or drive pass the intriguing jungle of a large flow dataset, examining each portion and aspect, observing streams of linkages and possible webs of intertwined relations. Hopefully, a more comprehensive understanding of the data can be systematically established.

The visualization environment has to be dynamically responsive to the analyst’s action. This is because our intuitive grasp of information and discovery of patterns depend largely on the instantaneous feedback (Haslett, et al., 1991). One of the innovative use is to provide a multivariate linked view (Haslett, et al., 1991) of the data in a fashion such that many views of the same data can be examined simultaneously and manipulated instantaneously.

**System Requirements**

This section describes the general characteristics of the scientific visualization environment for exploratory analysis of large flow data using dynamic graphics.

Our knowledge of human-machine interactions defines a friendly and effective software system. Based on this existing knowledge, the flow visualization system should
be an interactive, multiple-window system utilizing a graphical user interface. It should be easy to learn and operate, intuitive in graphic presentation, and manipulated on a click-and-go basis. The user should not be burdened by complex macro language programming work. User keyboard inputs must be minimized or avoided whenever possible. It should be a multiple-layer system, modularized such that various visualization operations can be grouped to present a clear structure, which is user-friendly and conducive for exploration.

Except requiring the user to spend several minutes to get familiarized with the graphic user interface convention, the system should be easy enough that the user is expected to just start the flow visualization system and be ready to go. Following is the design objectives the visualization system should meet:

1. Easy-to-use, self-explanatory graphical user interface;
2. Typing to be minimized. An attractive, simple click-n-go user manipulation system;
3. Multiple-window, multiple-layered system clearly structured, encouraging users to explore, instead of distracting them by a complex maze of disorganized windows and layers;
4. Judicious cartographic representation of flows, conveying directions, magnitude, origins, destinations, and general patterns of spatial interaction clearly and elegantly without excessively cluttering flow maps.
5. Providing efficient interactive visual information extraction methods specifically suitable for displaying spatial flow data.
6. Allowing the user to control thresholds with regard to the range of data displayed;

7. Displaying changes in screen graphics smoothly using animation technique in response to continuously user input, such as dragging on slider bars. This helps the user to observe small details of changes in flow patterns, while pursuing and locating the precise image desired.

8. Permitting convenient storage of flow images and retrieval for closer examination and comparison;

9. Interactive spatial aggregation at any level through user selection of polygons and collapsing;

10. Providing a means to link dynamically many views of the flow dataset, creating a powerful tool to reveal relationships.

11. Establishing instantaneous linkages among spatial flows, spatial distributions and aspatial attributes;

12. Capability of identifying clusters, trends, outliers, anomalies and reveal the spatial correlation;

13. Capability of identifying a pattern and searching for similar patterns in the database within the same time period or throughout the available time span;

14. Interactive manipulation of underlying flow data and aspatial attribute data. Defining new variables by combining existing data using common mathematical operations;
15. Displaying areal variables (income per capita, net flows of a region, etc.) using dynamic choropleth maps.

The next section elaborates the visualization techniques to be used in the flow exploratory system, providing theoretical discussions for these techniques.

**Visualization Techniques**

A series of visualization techniques are proposed, on the basis of careful considerations and error and trial, to tackle the exploration of a large, complex spatial flow dataset. These include interactive visual information extraction, three cartographic flow symbolizations, animated display, dynamic/static threshold setting, blink comparison, double view, visual information storage and retrieval, snapshots, region collapsing, forward brushing, backward brushing, mirror imaging of spatial and aspatial expressions of flow data, multivariate linked view, cluster and outlier identification, flow pattern identification and horizontal/vertical searching, interactive data manipulation, dynamic choropleth mapping of areal variables, etc.

These techniques are aimed at providing a set of comprehensive navigational tools to maximally expose the structures and intricacies that lie beneath the surface of a large flow dataset.

**Interactive Visual Flow Information Extraction**

A fundamental part of a scientific visualization system is the capability to allow a scientist to extract any desired information easily and efficiently. When this is applied
to flow visualization, a user should be able to select specific flows amongst a specific set of regions. Therefore, the following options should be provided for flow information extraction in an interactive manner:

Directional (O/D) flows: the user must be able to select and define polygons as either origins or destinations, so that directional flows can be displayed;

Two way flows: When a group of regions are selected, the spatial interactions among them in the form of two way flows \((\text{Region}_i \rightarrow \text{Region}_j \text{ and } \text{Region}_j \rightarrow \text{Region}_i)\) are presented; or alternatively the user can decide to draw net flows among them as described below.

Net flows: This option displays the net flows among any given regions. For instance, the net flow between region \(i\) and \(j\) equals \((\text{Flow}_{i\rightarrow j} - \text{Flow}_{j\rightarrow i})\) if the first flow is larger than the second. Otherwise, the net flow would be from region \(j\) to region \(i\).

**Three Cartographic Representations of Flows**

One of the difficult problems is how to use appropriate symbolization for flows in order to represent their directions, magnitudes and general patterns succinctly without cluttering the map. Therefore, based on the experiment of flow representation by Kern and Rushton (1969), Wittick (1976), Tobler (1987) and Becker et al. (1990a, 1990b, 1991a, 1991b), three different methods were selected and implemented in an improved fashion:
Arrow band width proportional to flow magnitude

With the same shading, an arrow's width is drawn proportionally to its magnitude. To determine the exact width for a flow, we can use a linear function, or some other mathematical function such as a logarithmic transformation. The user should also be able to control the maximum and minimum width of the arrow as an aid in controlling the clarity and aesthetics of the representation.

Constant band width with graduated solid shading

Using graduated saturation shading, the magnitude is made proportional to the depth of the color. A single color should be used for maximum clarity and contrast. In addition, solid shading should be selected instead of cross-hatched pattern shading in order to achieve a substantial reduction in computation complexity.

Lines with numbers

A single arrowed line is used to link an origin with a destination. The magnitude of the flow is indicated by a number associated with the line.

The modifications introduced here upon the three traditional cartographic representations of flows involve two aspects: first, the position and shading of arrow heads, and secondly in the use of graduated solid shading. An arrow head is placed at the end of a flow if there is only one flow involved. Otherwise it is placed 80% of the way from the origin unless the number of flows into a region exceeds 25, in which case arrow
heads are placed 20% of the way along the line. This strategy can reduce possible confusion of flow directions. To further increase clarity, hollow arrow heads are used without shading. Finally, a single color with different saturation levels is used to indicate flow magnitudes, instead of line patterns or using different colors. This is more simple and effective and there are no hidden line removal computations required.

Each of the three display methods has its own strength and weakness. The proportional width method is best when the distribution of flow magnitudes is not concentrated on the top portion. Otherwise, the map will be very cluttered with many thick bands. It is, however, very effective in communicating both exact magnitude and general pattern. The constant band width approach is generally very good in many situations because the complexity of a map is much better controlled since the band widths are the same. Of course, the number of regions still impacts the complexity. Comparatively speaking, it is quite effective in displaying a great number of flows, because it retains the general pattern very clearly. The line and number method is an efficient way to examine a small number of flows where the exact flow information is desired. However, it is not suitable to show large number of flows and does not represent general patterns of many flows well.

By offering these three different methods, the visualization system permits the user to experiment with them easily in order to find the most informative one.
Dynamic Threshold Setting

Controlling the range of data to be displayed by setting thresholds is a rather common technique in scientific visualization. Becker et al. have implemented 2-sided slider bars in SeeNet for this purpose (1990a, 1990b, 1991a, 1991b). Using the both ends of a slider bar, analysts can set an upper and/or lower threshold. This eliminates non-essential information and allows the analyst to concentrate on the interesting portion of the flows. Therefore in our flow visualization system slider bars should be made available to the user in order to interactively extract patterns within a certain data range. By dragging along the slider bars, a new flow map corresponding to the setting of each particular moment should be displayed dynamically so that the user can visually monitor changes of flow pattern and decide when to stop. In this way, the complexity of a flow map can be controlled interactively and the optimal representation with respect to the user’s interest is determined more easily.

Blink Comparison of Two Flow Images

Scientists in physics and astronomy etc. have long been using this technique to establish differences between two images. Equipped with much advanced computers, we now find it is also quite useful to compare two flow patterns using our temporary visual memory. When exploring the flow data set, a user may find an image insightful, and thus decide to save it. When a second interesting image is found, the user can try to compare, contrast and relate these two images by using a blink comparison approach. That is, by clicking a mouse button continuously, these two images are alternatively displayed on the
screen. This rapid visual shift between the two images creates two "flashes" which can be kept in memory for a very short period of time. This provides a unique opportunity to identify the similarities and differences between two interesting maps. The speed of the blink comparison can be controlled manually (Fig. 17), or alternatively, we can be set the speed at a desired level and allow the computer to create the blink effect automatically. This way the analysts can concentrate on the changes in the images.

**Double View**

Another intuitive way to compare and contrast two related images is to simply create two windows side by side so that the user can visualize flows in either windows.
To conserve space, both windows can share the same control panel. The panel can be activated with respect to either one of them, or for both windows, if appropriate. By manipulating graphics in both windows, a user can fine tune them until the most revealing results occur.

**Visual Information Storage and Retrieval**

It is imperative for a visualization system to be able to store "pictures". When the user explores the flow dataset and feels that some images created during the process are of particular importance, then these images must be able to be retrieved during or after the exploration for the purpose of comparison or re-examination. The system should be able to save the images in memory at run time for temporary storage, or/and to a hard disk as files for permanent storage.

One very useful function which a flow visualization system should have is to support hardcopy printing capability so that the "snap-shots" of screen displays can be manipulated and transformed to popular printable files (Postscript, Tiff files, etc.) for sharp printouts. The high quality printout then can be used for more careful study or for documentation purposes. MacSpin, as mentioned before, supports such an operation (Donoho et al., 1988).

**Region Collapsing**

This is a function to group several contiguous polygons into a larger region, so that the flow dataset can be examined at a higher level of spatial aggregation. A user
simply selects polygons from the map using a pointing device, and then they are collapsed into a region with a click of a button. The reverse "uncollapsing" operation should also be supported. Region collapsing can be beneficial in two situations. First, a user may want to reduce the complexity of the flows in an effort to better comprehend the overall pattern of spatial interactions. Second, only the interactions between the aggregated regions are essential under certain circumstances. For instance, a researcher might want to study the flows among New England, the Midwest, the West Coast and the Southeast of U.S. using a flow matrix in which states were the data collection unit.

A typical criticism of this approach is that many important details are lost in the generalization process. For example, when suggesting to collapse a migration matrix by grouping polygons in order to reduce complexity, Tobler (1987) also warned that it is "not a desirable procedure and is to be avoided if at all possible."

However, this spatial restructuring of the data can be done according to any scheme of grouping and can be dissolved interactively. The user has virtually numerous ways to define new regions. Thus the flow data can be viewed using many different schemes of regionalization. Furthermore, the merged original polygons can be recovered by a click of a button. This definitely reduces the drawback of this approach because the region collapsing is not a static, one-directional process. It is reversible and changeable. Many surprising insight may emerge as we look at spatial flows from different levels and schemes of spatial aggregation.
Forward and Backward Brushing: Mirror Imaging of Spatial and Aspatial Expressions

Brushing is a powerful visualization tool for exploring multivariate data. It creates many interrelated views of the same dataset through the thread provided by a continuously moving highlighting device: a rectangular brush. This technique has been expanded in this research to a comprehensive technique for the study of spatial flows: forward and backward brushing, and an accompanying idea: mirror imaging of spatial and aspatial expression.

Related works

The brushing technique was first proposed by Becker and Cleveland (1984, 1987a, 1987b), and it was reviewed in Chapter II. Basically, a special scatterplot matrix is created to present exhaustive bivariate plots of many variables. A brush is used to identify a single data point or a cluster of data points in the active panel, while in the non-active panels the corresponding data points are highlighted in their respective scatterplots. The highlighting pattern is dynamically changed as the brush is moved. It is easy to see that brushing is a useful tool in examining outliers, clusters, or any interesting data point distribution pattern. Besides highlighting, brushing can be utilized in a variety of other operations such as shadow highlighting, deleting and labeling with three different paint modes.

Monmonier then proposed to add a geographic map to the scatterplot matrix (1989a, 1990b). When a brush moves within the scatterplots, he suggested that the
corresponding polygons should also be highlighted. Tang (1993) subsequently implemented this idea.

Haslett et al. (1991) describes such dynamically connected, multiple portrayals of geographic data and related statistics as linked views. Such multivariate linked views offer us an opportunity to look at the relationship between many variables in an entirely new and insightful perspective.

**Forward and backward brushing**

The idea of forward and backward brushing in this research builds on top of the aforementioned Tang's implementation, and extends the brushing technique to a new level of completeness and complementarity.

Under this approach, not only can the pairwise scatterplots be brushed for the corresponding geographic correlation, but also the geographic space can be brushed to reveal patterns of the highlighted data points in the scatterplot boxes. This establishes a complete and reversible relationship between the geographic space and the multivariate scatterplot space.

The process in which a brush moves in a scatterplot box with the corresponding data points in other scatterplot boxes as well as the corresponding geographic polygons highlighted is called "Forward Brushing". Furthermore, it also includes the portrayal of the corresponding spatial interactions among the highlighted regions in terms of two way flows and net flows. Fig. 18 and 19 illustrate the parallelism of the spatial domain and the aspatial domain of a flow dataset. The scatterplot matrix is comprised of interesting
Figure 18: Forward Brushing. The brush moves in the aspatial domain, while the mirror or secondary images are displayed in the spatial domain.

Figure 19: Backward Brushing. The brush moves in the spatial domain, while the mirror or secondary images are displayed in the aspatial domain.
social-economic variables deemed relevant to the underlying mechanism of spatial flows, such as income per capita, crime rate, population density, unemployment rate, etc. This allows us to examine not only the linked views of these attribute variables amongst themselves, but also the geographic implication and spatial interactions.

In order to effectively identify interesting clusters, the brush geometry, size and orientation should be determined interactively by the user. Different geometry has its different definition of a cluster. It should include circular, elliptical, triangular, or any other arbitrarily defined shape, in addition to the traditional rectangles. Sizes ought to be adjusted precisely and conveniently. Orientation is quite important when trying to encircle a particular cluster. For instance, if a scatterplot shows a conspicuous linear relation, then it is interesting to highlight that linear point-cloud precisely by rotating the brush’s orientation, and examine its spatial correlations. At the same time, we can also examine the geographic distribution of the outliers because those polygons which are not highlighted belong to the outliers. Permitting the user to adjust the exact angle of the brush becomes very helpful in this case.

"Backward Brushing", on the other hand, is to brush the spatial domain, with the corresponding attribute values highlighted in all scatterplots. Again, the size of the brush should be determined interactively so that it can highlight a single region or multiple regions at a time. The highlighting in the scatterplot not only tells the exact values of the attribute variables of a brushed region, but also its values relative to other regions. For instance, for a brushed region, we can not only find precisely its value of per capita income, but also how this income stands in comparison to other regions by simply
looking at the location of the highlighted data point relative to the other points. This will tell us whether the region is an outlier or not.

Spatial interactions (flows) can be also displayed in backward brushing. If a large brush is used so that it can encircle several regions, interactions can be easily portrayed among the brushed regions. Nevertheless, when an analyst wishes to examine one region at a time using a small brush as a highlighting device, we can examine the flows between the brushed region and other "anchor" regions. Such anchors are pre-selected regions which are of particular interest to the analysts.

The idea of setting an anchor actually originates from Becker et al. (1990a) in visualizing link statistics between two nodes of a network, although a remarkable difference exists between his and what is used here. In his SeeNet, as the cursor walks from one node to another, the link statistics between the current region and the anchor is displayed. In backward brushing, multiple regions can be designated as anchors instead of only one region. Then the analyst can choose to define them as either origins or destinations with regard to the brushed region. Consequently directional flows (O/D flows) are displayed between the brushed region and the anchors. Alternatively, the anchor regions and the brushed region are lumped together as an interacting group, and net flow or two way flows amongst them can be presented.

Adding flows into the brushing technique makes it inclusive of typical geographic inquiries: spatial aspect (geographic distribution and spatial interaction), and aspatial aspect (social-economic characteristics and mechanism).
It should be emphasized that the brushing process must be dynamic and instantaneous in refreshing the graphics. The combination of Forward Brushing and Backward Brushing achieves a remarkable degree of self-containment and complementarity by dynamically linking both the spatial expression and aspatial expression of the flow dataset.

**Mirror imaging of spatial and aspatial expression**

In the broadest sense, the brushing technique is analogous to a mirror reflection process (see Fig. 18, 19). Consider Becker’s scatterplot brushing. All non-active panels are like mirrors reflecting the highlighted points in the active panel from different perspectives. Forward brushing differs in that it has one more mirror: the spatial correlation and interaction.

Backward brushing, on the other hand, uses all pairwise scatterplots as "mirrors" to reflect the social-economic characteristics of the brushed region(s). The combination of forward and backward brushing constitutes a very interesting "mirror imaging" process between the spatial elements and aspatial elements of a complex flow data.

**Identification of Outliers, Anomalies, Trends, Clusters within a Flow Dataset**

By brushing the scatterplot or geographic space, and controlling the size and geometry of the brush, the user can identify outliers, anomalies, clusters, trends of the flow dataset easily and examine their implications.
The size of the brush is very important, since it defines the dimensions of outliers, or interesting clusters. Typically, a small brush is used to identify individual data points or a very small group of points, while a larger brush is usually used to encircle a cluster.

The geometry of a brush determines how data points are included or excluded from a certain cluster. It should be statistically significant and meaningful, and the shape could be any of the following: square, rectangle, circle, ellipse, triangle, or irregular. Each has a different notion of a cluster. Obviously, a square considers a cluster as equidistantly expanding in four cardinal perpendicular directions, while a circle constitutes a cluster by radiating equally in all directions. A rectangle differentiates expansion distances in two major directions, which is ideal for variable isolation and trend detection. An ellipse encircles a space in which any point has a distance to the two foci less than a predefined measurement. An irregular brush can be defined freely by a user, with any arbitrary shape. The dimensions of a brush can represent some important statistical properties of the data points, such as standard deviation, mean, median, etc.

Flow Pattern Identification and Searching

The flow visualization system should also enable a user to identify a spatial interaction pattern, and conduct a search for similar flow patterns within the same time period or throughout the available time span. The former is called a "horizontal search", and the latter a "vertical search".

Defining a pattern is the first and also most difficult task in pattern recognition. However, dealing with a specific type of data (spatial flows) is not intractable, since there
are only three fundamental characteristics in a flow pattern: the magnitude of flows, their relative ranking, and directions. Given two related flow patterns, we can match up and compare corresponding flows, and then calculate the dissimilarity, which, in turn, is compared with a maximum dissimilarity possible under that particular circumstance. Consequently, a similarity index can be calculated which ranges from 0.0 (the most dissimilar) to 1.0 (the most similar).

However, three different cases of flow patterns merit separate consideration, since the formula to compute the similarity index will have to be specifically tailored. These three cases are discussed separately here.

One-to-many or many-to-one flows

This is a case where there is only one origin (an one-to-many flow pattern) or one destination (a many-to-one flow pattern) involved. Due to its simplicity, We can conduct a horizontal and a vertical search for such a flow pattern. For example, in the one-to-many flow situation, we can fix the destination regions but change the origin, so that we can compare the original flow pattern with any pattern generated by flows from any other origin to the same destinations. This is equivalent to ask the question: "Is there any other region in the same time period which has similar flows into this same set of destination regions?" This is the horizontal search, which would generate pseudo-code such as the following:
For each region

Compare the flow pattern generated by flows from this region to the same set of destinations with the original flow pattern

End do-loop ||

The mathematic equation for computing the similarity index in a horizontal search for an one-to-many flow pattern is given below:

\[
S_{\text{Index}}_k = 1 - \frac{\sum_{j=1}^{n} |M_{ij} - M_{kj}|}{\text{Max. D_Index}}
\]  

Where

- \(i\): The original origin region;
- \(j\): One of the original destination regions;
- \(k\): The new origin region;
- \(n\): The total number of destinations;
- \(M_{ij}\): Flow from region \(i\) to region \(j\) (normalized), one of the original flows;
- \(M_{kj}\): Flow from region \(k\) to region \(j\) (normalized), the flow from a new region to one of the same destinations;
- \(S_{\text{Index}}_k\): Similarity index of region \(k\);
- Max. D_Index: Maximum Dissimilarity Index;

The formula of computing Max. D_Index for this equation and Equation 2 to 5 will be given shortly by Equation 6. Obviously, many S_Indices can be computed. Then they can be ranked from the highest to lowest, indicating a descending order of similarity with the original pattern.

For the case of a many-to-one flow pattern, a horizontal search can be similarly constructed by simply fixing the origins and varying the destination. This is equivalent to ask "Is there any other region in the same time period which receives similar flows
from the same set of origin regions?" The following pseudo-codes can be used for such a search:

\[ \begin{align*} 
|| & \text{For each region} \\
& \quad \text{Compare the flow pattern generated by flows from the same set of origins into this region with the original flow pattern} \\
& \text{End do-loop} || \\
\end{align*} \]

The equation for computing similarity index for a many-to-one case is given here:

\[ S_{Index_k} = 1 - \frac{\sum_{i=1}^{m} | M_{ij} - M_{ik} |}{\text{Max. } D_{Index}} \quad (2) \]

Where

- \( i \): One of the original origin regions;
- \( j \): The original destination region;
- \( k \): The new destination region;
- \( m \): The total number of origins;
- \( M_{ij} \): Flow from region \( i \) to region \( j \) (normalized), one of the original flows;
- \( M_{ik} \): Flow from region \( i \) to region \( k \) (normalized), the flows into a new region from one of the same origins;
- \( S_{Index_k} \): Similarity index of region \( k \);
- \( \text{Max. } D_{Index} \): Maximum Dissimilarity Index;

To search vertically is to compare the original flow pattern with flow patterns generated by the same sets of origin(s) and destination(s) throughout the available time period. An equivalent query would be "Has this type of a flow pattern happened before?" The pseudo-code is:

\[ \begin{align*} 
|| & \text{For each time period} \\
& \quad \text{Compare the flow pattern generated by the same set of origin(s) and destination(s) at this time period with the original flow pattern} \\
& \text{End do-loop} || \\
\end{align*} \]
For an one-to-many flow pattern, the similarity index in a vertical search is given below:

\[
S_{\text{Index}_{t_k}} = 1 - \frac{\sum_{j=1}^{n} | M_{ijt_k} - M_{ijt_0} |}{\text{Max. D_Index}}
\]  

where:
- \( i,j \): The original origin and destinations;
- \( t_k \): Time \( k \);
- \( t_0 \): Time 0: the time of the original flow pattern;
- \( n \): The total number of destinations;
- \( M_{ij0} \): Flow from \( i \) to \( j \) at the original time (normalized);
- \( M_{ijtk} \): Flow from \( i \) to \( j \) at time \( k \) (normalized);
- \( S_{\text{Index}_{t_k}} \): Similarity index of the flow pattern at time \( k \);
- \( \text{Max. D_Index} \): Maximum Dissimilarity Index;

Similarly, for a many-to-one flow pattern, the similarity index in a vertical search is specified in Equation 4:

\[
S_{\text{Index}_{t_k}} = 1 - \frac{\sum_{j=1}^{m} | M_{ijt_k} - M_{ijt_0} |}{\text{Max. D_Index}}
\]  

where:
- \( i,j \): The original origins and destination;
- \( t_k \): Time \( k \);
- \( t_0 \): Time 0: the time of the original flow pattern;
- \( m \): The total number of origins;
- \( M_{ij0} \): Flow from \( i \) to \( j \) at the original time (normalized);
- \( M_{ijtk} \): Flow from \( i \) to \( j \) at time \( k \) (normalized);
- \( S_{\text{Index}_{t_k}} \): Similarity index of the flow pattern at time \( k \);
- \( \text{Max. D_Index} \): Maximum Dissimilarity Index;
Many-to-many flows

In this case, the flow pattern is generated by multiple origins and multiple destination. Similar patterns can only be searched throughout the entire time span, and a horizontal search would be computational impossible and theoretically meaningless. Therefore the vertical search compares the original pattern with the flow patterns generated by the same sets of origins and destinations in other time periods. The method for computing $S_{\text{Index}}$ is given in Equation 5:

$$S_{\text{Index}}_{k} = 1 - \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} | M_{ij_{k}} - M_{ij_{0}} |}{\text{Max. D\_Index}}$$ (5)

where:

- $i,j$: The original origins and destinations;
- $t_k$: Time $k$;
- $t_0$: Time 0: the time of the original flow pattern;
- $m$: The total number of origins;
- $n$: The total number of destinations;
- $M_{ij_{0}}$: Flow from $i$ to $j$ at the original time (normalized);
- $M_{ij_{k}}$: Flow from $i$ to $j$ at time $k$ (normalized);
- $S_{\text{Index}}_{k}$: Similarity index of the flow pattern at time $k$;
- $\text{Max. D\_Index}$: Maximum Dissimilarity Index;

Maximum Dissimilarity Index

$max. D\_Index$ measures a hypothetical extreme situation in which a flow pattern is maximally different from the original pattern. Equation 6 can be used to compute it. This equation is applicable to the computation of $\text{Max. D\_Index}$ of Equation 1 to 5.
$Max. \ D_\text{Index} = \sum_{i=1}^{m} \sum_{j=1}^{n} |X - M_{ij}|$ \hspace{1cm} (6)

Where

- $i,j$: An origin and a destination;
- $n$: Total number of destinations;
- $m$: Total number of origins;
- $M_{ij}$: The original flow from $i$ to $j$, normalized;
- Max. $D_\text{Index}$: Maximum Dissimilarity Index;
- $X$: A flow maximally dissimilar to the original flow, and
  - $X = 1.0$, if $M_{ij} \leq 0.5$;
  - $X = 0.0$, if $M_{ij} > 0.5$;

Net flows

If the flow pattern is generated by net flows among regions, then, similar to the many-to-many flow case, only a vertical search is possible. This is given in Equation 7.

$$S_{Index_{tk}} = 1 - \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} |Net_{M_{ij_{tk}}} - Net_{M_{ij_{t0}}}|}{Max. \ D_\text{Index}}$$ \hspace{1cm} (7)

where:

- $i,j$: The original origins and destinations;
- $t_k$: Time $k$;
- $t_0$: Time 0: the time of the original flow;
- $m$: The total number of origins;
- $n$: The total number of destinations;
- $\text{Net}_{M_{ij_{tk}}}$: Net flow from $i$ to $j$ at time $k$ (normalized);
- $\text{Net}_{M_{ij_{t0}}}$: Net flow from $i$ to $j$ at the original time (normalized);
- $S_{Index_{tk}}$: Similarity index of the flow pattern at time $k$;
- Max. $D_\text{Index}$: Maximum Dissimilarity Index;

In this case, the maximum dissimilarity has to be computed differently from Equation 5. A maximally different net flow would be one that is pointed toward the opposite direction with the largest possible value. Equation 8 shows how to compute it:
\[ \text{Max. D. Index} = \sum_{i=1}^{n} \sum_{j=1}^{m} | X - \text{Net}_M_{ij} | \]  

(8)

Where

- \( i, j \): An origin and a destination;
- \( n \): Total number of destinations;
- \( m \): Total number of origins;
- \( \text{Net}_M_{ij} \): The original net flow from \( i \) to \( j \), normalized;
- \( \text{Max. D. Index} \): Maximum Dissimilarity Index;
- \( X \): A net flow maximally dissimilar to the original net flow, and
  - \( X = 1.0 \), if \( \text{Net}_M_{ij} > 0 \);
  - \( X = -1.0 \), if \( \text{Net}_M_{ij} \leq 0 \);

In any of the above three cases, the top ranking S_indices are selected as the most similar patterns and can be displayed side by side with the original pattern for comparison.

**Interactive Data Manipulation and Dynamic Choropleth Mapping**

Interactively manipulating the data allows a user to define new variables so that the original raw data can be viewed from a new perspective. Variables and operators can be selected from menus or lists using mouse buttons. The result then is assigned to a new variable whose name has to be typed in. Simple text editing fields should be provided in the graphical user interface for entering the equations if the user choose to do so. For example, if the data set only contains total migrations from one state to another, new variables can be computed by dividing the numbers by the origin population or the destination population, resulting in two standard measurements of spatial interactions. Viewing and comparing these three variables can be quite interesting.
For some attribute data of the flow dataset, which are associated with geographic areas, we can use choropleth maps to view spatial variations of these variables. These variables typically include social-economic attributes of each region (income per capita, population density, crime rate etc.) and areal variables such as the total in-migration and total out-migration, as well as the net migration of a region.

The user can control the range of the data displayed by setting the upper and lower thresholds. Slider bars can be used to change the thresholds interactively with instant refresh of the choropleth map. Through progressively dividing the remaining data range into several intervals of a choropleth map, the user can have a much closer look at the regional pattern of each variable at a finer scale and see interactively how the number of regions within the data range shrinks or expands as the thresholds change.

Animated Display

To achieve good interaction between the user and the graphics, screen displays should be refreshed and redrawn instantly in response to user actions such that a series of smooth changes in the graphics give an impression of virtual animation. This technique should be implemented for the dynamic choropleth mapping, interactive flow threshold setting and brushing.

Consistent and Intuitive User Interface

The visualization system should adopt a widely accepted user interface convention, such as OPEN LOOK GUI. It should be consistent throughout the system, providing an
elegant and intuitive shell for the user to access the power of the program. Extensive online help should also be provided for concise and informative instructions.

**Conclusion**

This chapter outlines the system requirements for the flow visualization software and proposes a series of techniques to explore a large flow dataset. It should integrate good cartographic representation with comprehensive, and effective flow visualization techniques. It also should be a user-friendly environment based on multi-window graphic user interface. The visualization techniques are tailored to take full advantage of modern dynamic graphics and specifically designed for exploring a large flow dataset. Ideas such as forward and backward brushing, flow pattern searching, animated display, etc. should be quite powerful to reveal the hidden regularities of a flow dataset.

In the next chapter, we will see how the system requirements are mapped to a realistic flow visualization system and how the proposed visualization techniques are implemented on a Unix workstation.
CHAPTER IV
SOFTWARE IMPLEMENTATION

Implementation Overview

In last chapter, the system requirements and visualization techniques for the proposed flow visualization system have been clearly specified. This chapter discusses how the logical model of the visualization system is mapped to an operational software system. The discussion is organized into two main sections: first, implementation considerations and secondly, an extensive discussion of the proof-of-concept software: SEFLOW, which stands for "Scientific Visualization and Exploratory Data Analysis of Large Spatial Flow Data".

The first section addresses following topics: hardware and software platform considerations, graphical user interface issues, system design and program structure, implementation techniques, data structure and quality.

The second section details the results of implementation using a number of plates of screen photographs. Each of the five modules of SEFLOW is discussed from the perspective of mapping the specified logical system to an operational system.

In short, the dynamic visualization system for the exploration of complex spatial flow data which was developed in the last chapter has been implemented under the X
Window System\textsuperscript{1} running on a Sparc 2 Sun workstation using XView\textsuperscript{2} as a graphical user interface. The end product is a highly interactive, insight-pursuing software called SEFLOW. It is structured into five interrelated modules: a Cartographic Representation Module (CRM), a Forward Brushing Module (FBM), a Backward Brushing Module (BBM), a Pattern Searching Module (PSM), and a Dynamic Choropleth Mapping/Interactive Data Manipulation Module (DCM/IDMM). Each module performs a clearly defined task and permits visualization and exploration the flow dataset from different perspectives. The modules are organized into a multi-layered environment, in which the user can navigate from one module to another with a click of a button. Each module is a complete, multi-window environment, divided into control panel window(s), and display window(s). The X Window System provides a very versatile and powerful programming environment for multi-window management, interactive 2-D graphics, and the writing of event-driven software. The software access is via the XView graphical user interface, offering an intuitive, elegant and consistent look and feel within the working environment.

Implementation Considerations

This section discusses various aspects of converting the system specifications and visualization techniques into a flexible, user-friendly, and effective exploratory system. There are many issues involved in a complex system design and implementation, such as choosing the appropriate software and hardware platforms, structuring the system, dealing

\textsuperscript{1}See Appendix B on the X Window System.

\textsuperscript{2}See Appendix D for more detail on XView.
with data input and output, programming techniques and languages, designing the user interface, etc. These issues are examined here.

**Hardware and Software Platform Considerations**

Choosing the appropriate hardware and software platforms is perhaps one of the most important decisions in software implementation. The hardware environment determines, to a large extent, many critical aspects of the performance of a program: the speed at which computation, drawing, and data access are accomplished, the amount of memory available, the number of colors which can be created, and screen resolution, etc. The software "platform" is the programming environment, which determines what software development tools are available to the developer. This usually includes the underlying operating system, programming languages, computational and mathematical libraries, 2-D and 3-D graphics function libraries (such as graphics primitives and transformation pipelines), mechanisms to acquire, or to be notified of user input and other significant events, user interface tools and styles, extensibility, and portability.

A wise choice of the software environment can help tremendously during the development process and in the critical problems of program maintenance. A nightmare in software engineering occurs when an environment is found, half the way down the road, to be so severely limited and inflexible in certain unforeseen respects that it consequently impedes a successful implementation. Unfortunately this cannot be easily avoided because sometimes decisions have to be made under some degree of uncertainty, especially when it involves a graduate student with limited time, experience and
knowledge of various software platforms. A software developer has to know exactly what the platform is capable of doing and how flexible and extensible it is in order to make a well-informed decision and to avoid disastrous consequences occurring somewhere down the road. Therefore, great caution must be exercised before making a final commitment regarding the platform choice.

In making such decisions for developing SEFLOW, the following are a summary of the guidelines used in choosing the hardware platform:

1. It should run fast enough in computation and drawing to support near instantaneous, complex graphics generation;

2. It must have a sufficient amount of internal memory and be capable of utilizing it to store data and pixmaps for fast visual feedback and animation purposes;

3. The UNIX operating system should be used with fairly large volume of hard disk space for storing large spatial flow data;

4. The platform, while powerful enough to deal with very large, complex spatial-temporal flow datasets, should be popular enough to be found in ordinary academic institutions. This would broaden the software acceptance in terms of hardware environment.

Obviously, a logical choice would be a Unix workstation. In this case, a Sparc 2 Sun workstation was selected because it was available and is a very popular platform in a majority of academic institutions, and met the technical requirements. It is excellent in terms of processing speed, memory, disk space, screen resolution, available colors (256
with 8 bitplanes at one time). Undoubtedly it is much better than any microcomputer platform in every respect, and not as exotic and expensive as Silicon IRIS graphics workstations, which are very powerful yet not often seen in ordinary academic environments when this research began.

On the other hand, the guidelines for choosing an appropriate software platform can be summarized as below:

1. The platform should provide extensive mathematical and 2-D and 3-D graphical function libraries to alleviate programming difficulties;
2. It should run under the Unix operating system and provide access to the C programming language, which is currently the most versatile high level language;
3. It must offer excellent development tools for building the graphical user interface;
4. It must be portable over a variety of hardware platforms. That is, the program should be written in a form which can be transplanted to other platforms with minimal programming effort.
5. It should provide the state of art in every aspect mentioned above, yet be highly extensible such that the system is transparent to the programmer, and supports mechanisms and necessary low level tools for the programmer to design and develop their own libraries and GUI widget sets. In short, programmers should not be narrowly confined in the existing capabilities of the system. Instead, they should have access to the "insides"
of the system and to fundamental programming tools so that they can descend to low level operations if necessary. This requirement with respect to the software platform can not be over-emphasized because we should have the best development tools that modern computer science has to offer, as well as providing a high degree of flexibilities to extend beyond what the existing tools provide. It must help to avoid "re-inventing wheels" as well as not subjecting programmers unduly to the limitations of the platform.

After careful study and comparison, it was concluded that SEFLOW should be written in the X Window System environment, to utilize its rich assortment of development tools including 2-D and 3-D graphics primitives and transformation pipelines, window management functions, event mechanism, graphical user interface toolkits, and its advantage of hardware-independence.

The reasons for choosing the X Window System as the programming environment are three-fold. First, it is a widely accepted and well implemented software environment on almost all workstations, such as Sun, HP, Sony, DEC, IBM, etc., a de facto workstation industry standard. It is designed to be hardware independent by establishing communication between programs and the underlying hardware through a shell provided by the X protocols. That is, the program does not need to deal with the idiosyncrasy of a particular hardware. This ensures its portability across a variety of hardware platforms.

Second, it is currently the most powerful and most comprehensive programming environment available in the public domain, that supports development of event-driven
programs, 2-D graphics (Xlib) and 3-D graphics libraries (Phigs, XGL, graPHIGS, Phigs-Plus), a wide range of elegant graphical user interface packages (XView, Motif, Xaw, Olit, etc.), and C code binding. The event driven program is a revolutionary program style, as opposed to the traditional program-controlled interface. A traditional program dictates the flow of control and regards users as participants for inputs and other responses. In an X program, however, users' actions and interactions among X applications are translated into "events", which are captured by an X server and dispatched to the relevant programs for appropriate responses. A typical X program contains elaborate event procedures as the core of the program, responding to various events of interests, and behave in response to users' actions, instead of asking users for input. Therefore the program is structured around the user, not the other way around.

Third, the X Window System is very flexible and extensible. It not only supports numerous software development tools, but also provides mechanism for programmers to develop their own customized features and incorporate them into X. For instance, the access to the C language guarantees the flexibility to write any functions a programmer wishes; he/she can also write any GUI widget using X intrinsics instead of being limited by the existing widget set.

**Graphical User Interface Issue**

The X Window System also makes available many excellent graphical user interface (GUI) toolkits, such as Motif, XView, Olit and Xaw. This substantially reduces the amount of programming work. A good graphical user interface provides a consistent,
attractive and intuitive feel and look to the program. This helps to alleviate the anxiety, fear and difficulties of the novice user during the learning process. The factors of simplicity and consistency are pivotal in attracting users to experiment and explore the functions of the software. When the interface looks familiar, the user can effectively utilize knowledge gained from previous programs of the same GUI style, which speeds up the learning curve considerably.

The XView GUI toolkit is chosen to develop the interface portion of SEFLOW, because of its excellent GUI style, elegant appearance and easiness of programming. XView conforms to the OPEN LOOK\(^3\) graphical user interface conventions, which is a widely accepted and well-implemented interface style. For example, the window system OpenWin developed by Sun Microsystems for Sun workstations uses this GUI style. All programs conforming to OPEN LOOK behave remarkably similarly. Therefore a user can change from one program to another without being forced to learn many new approaches, and the knowledge gained from working with previous programs can be completely transferred to the new program as long as the GUIs are based on the OPEN LOOK.

In comparison to other GUI toolkits such as Motif and Olit, XView is much simpler to program. It is built directly on Xlib and provides very easy mechanisms to write the interface, while other toolkits are built, instead, on top of X Intrinsics which in turn is built on Xlib (Heller, 1991). This requires dealing with Intrinsics functions and generally working in a much more complicated programming environment.

\(^3\)See Appendix C.
Nevertheless, the decision of which X GUI toolkits to use is not a final one, since the interface can be, and indeed should be, changed easily. Modern software engineering requires that the program be written so that its graphical user interface is detachable from the core of the program. This means that the core is relatively independent of the GUI, and that the GUI can be replaced with minimal programming effort, much like a person changing clothes. Therefore the GUI serves as a flexible shell, providing a certain unique look and feel of the program, but it is not an inseparable part of the program.

**Data Structure and Data Quality**

The spatial flow data used for demonstration purposes in the present project shows the internal migration among U.S. states for three time periods: 1955-1960, 1965-1970, and 1975-1980. These data came from the population census of 1960, 1970, and 1980. Since digital files are not available, the data were manually entered.

In order to generate a database large enough for a meaningful spatial pattern searching, the O/D matrices of the two intermediate time periods (1960-1965, 1970-1975) are estimated by averaging the flows of their respective previous and subsequent time periods. For instance, the O/D matrix for 1970-1975 is computed by averaging flows of 1965-1970 and 1975-1980. Together with the true flow data of the three time periods, this generates 5 matrices for the time period from 1955-1980.

Then based on these data from 1955-1980, additional 15 O/D matrices, prior to 1955, each corresponding to a 5 years of population migration data, were extrapolated
using a simple weighted average scheme using "time distance" as the weighing factor. This makes the total number of flow matrices to 20, with total time span of 100 years.

Obviously, the interpolation and extrapolation schemes are very simplistic and definitely are not optimal in flow estimation. However, the purpose here is not to establish elaborate flow estimation formula, but to generate a big enough database so that the flow visualization system can be demonstrated and tested.

The second type of data involves social-economic variables associated with each region for the time period of 1975-1980 only, and were manually keyed in. These variables included population density of each state, averaged from population data of 1975, 1978, and 1980; unemployment rate averaged from data of 1975, 1978, and 1980; crime rate (cases per 100,000 population) estimated from data of 1978, 1979 and 1980; per capita income in current dollars from 1975, 1977, 1980. These data were taken from the Statistical Abstract of the United States.

The last category of data involves other areal variables associated with each state, including total in-flows and total out-flows, centrality and population potential. Equations 10 and 11 show how the last two were computed using two well-known equations in geographic literature:

\[
Centrality_i = \sum_{j=1}^{n} Distance_{ij} \tag{9}
\]

Where
- \(Centrality_i\): Centrality of region i;
- \(Distance_{ij}\): Physical distance between centroids of region i and j;
- \(n\): Total number of regions;
\[ P_{Potential_i} = \sum_{j=1}^{n} \frac{Population_j}{Distance_{ij}} \]  

(10)

Where:
- \( P_{Potential_i} \): Population potential of region i;
- \( Population_j \): Population of region j;
- \( Distance_{ij} \): Physical distance between centroids of region i and region j;
- \( n \): Total number of regions;

The O/D matrices are stored as a three dimensional matrix, with the first dimension representing time, the second and third representing rows and columns of the matrix. Social-economic variables are stored as simple arrays. All areal variables are also normalized so that they range from 0 to 1.0 for convenient scatterplot display when using brushing techniques. However, in order to deal with very large flow datasets, well-designed data structures and a good database management system are indispensable for efficient data import, storage, retrieval and manipulation. This painstaking design work was not done because the top priority of this research was to develop and test visualization techniques for flow data. Therefore this work was decided to be left for subsequent researches.

**Implementation Techniques**

There are several basic rules followed closely during the development of SEFLOW in order to keep the program robust, easy to maintain and efficient:
1. Maximize the use of high level Xlib and XView functions and, where possible, avoid low level C programming in order to alleviate coding difficulties and increase program efficiency;

2. Structure a procedure so as to perform a single, clearly defined function and be highly generic in declaring arguments to deal with a wide variety of situations;

3. Use callback functions to respond to events of GUI widgets, such as button pushing, slider bar dragging, etc., and use more sophisticated event procedures to handle events in the drawing area;

4. Design algorithms so that screen refresh during dynamic display occurs at maximum speed and smoothness, and avoid uncomfortable flickering due to slow drawing;

5. Keep the graphical user interface simple and informative, use consistent layout with regard to the locations of control panels, drawing area, etc. Place GUI buttons in the control panel areas according to their importance: primary buttons are placed first, and utility buttons such as "clear", "help", "quit" are always kept toward the right end.

**System Design and Structure**

The software are structured according to visualization and exploratory tasks. Such task groups are identified and organized into five modules of distinct functionalities and purposes. Each task group is constructed as an independent module in a multi-layered
Figure 20: The structure of SEFLOW. Five modules and its hierarchical access.
window system, with its unique functions and user interface. Fig. 20 illustrates the structure of SEFLOW and its five modules. The five modules are accessed through a reversible, hierarchical path (top-down, or bottom-up) and a horizontal path (size-ways).

The first module, Dynamic Choropleth Mapping and Interactive Data Manipulation, resides in the Main Panel layer, which provides access to any of the other four modules. Each of the four modules can access the Main Panel layer, thus each has access to any of the other modules. Ideally, five modules should be directly accessible horizontally; however, only FBM and BBM, PSM and CRM are bridged due to the time constraint.

The principal functions of the five modules are summarized briefly here:

1. **CRM (Cartographic Representation Module):**
   Experimentation with the three representation methods for spatial flows, together with a rich assortment of interactive visual flow information extraction techniques;

2. **FBM (Forward Brushing Module):**
   Implementation of the forward brushing by constructing a two canvas window environment, brushing in the scatterplot space and highlighting geographic space;

3. **BBM (Backward Brushing Module):**
   Implements the backward brushing with a very similar window layout, while brushing in geographic space instead of in the scatterplot space;
4. PSM (Pattern Searching Module):

Identifying a flow pattern and searching for similar ones in the database in the same time period or over the entire available time span;

5. DCM/IDMM (Dynamic Choropleth Mapping and Interactive Data Manipulation Module):

Examining areal data, such as social-economic variables, using choropleth maps. Controls the data range dynamically using two slider bars. Manipulates data by using common mathematical operations, and creating new variables.

The next section discusses each module in detail with regard to implementation details and functions, using a number of screen photographs.

**Implementation Results**

SEFLOW was developed over a time period of about 6-7 months from July, 1992 to February, 1993 as a proof-of-concept implementation of ideas relating to visualization and exploratory data analysis of spatial flow. It is structured into five inter-related modules, each with distinct functionalities, in a multi-layered, multi-window dynamic graphics environment. Modules are accessible through a hierarchical path and partially through horizontal paths. The following provides an indepth discussion of each module of SEFLOW.
Dynamic Choropleth Mapping and Interactive Data Manipulation Module (DCM/IDMM)

This module resides in the Main Panel layer of SEFLOW, which serves as the gateway to the other four modules through four buttons: "Cartographic Representation", "Forward Brushing", "Backward Brushing", and "Pattern Searching".

The purpose of this module is to give a user the chance to examine the spatial variation of areal variables using dynamic choropleth maps. Areal variables include social-economic variables such as income per capital, population density, crime rate, unemployment rate, and others such as total in-flows and total out-flows from the region, etc. Thus a clear preview of the regional characteristics related to spatial flows can be
achieved. The social-economic dimensions are essential to understand the underlying mechanism which drives migration flows and other flows.

Choropleth maps are generated by dividing the data range of any variable into five equal intervals. Solid shading of a single color with graduated saturation is used to represent a clear and smooth spatial differentiation. Three primary colors (Red/Green/Blue) and gray are available for the solid shading. The saturation levels of a color are created by assigning different RGB values. For example, following RGB values may be used to generate five red colors with increasing saturation:

\{255, 190, 190\}, \{255, 140, 140\}, \{255, 90, 90\},
\{205, 0, 0\}, \{139, 0, 0\} /*red level 1,2,3,4,5*/

Obviously the first three levels are created by fixing the red value at 255 (maximum) and simultaneously decreasing the green and blue values thus reducing the overall brightness. The last two levels keep GB values as zero and reduce the red values to generate increasingly darker red colors.

A new choropleth map is drawn instantly by selecting any variable from a pull-down menu (Fig. 22). The user also has a choice of using either normalized values or just the raw data.

The choropleth map can be dynamically refreshed by changing the upper and lower thresholds on two slider bars. The new maps only shade the regions within the thresholds. New shadings are done according to the new data interval scheme by dividing the remaining range into five equal intervals, while the regions whose values are outside the range set by the two slider bars are left blank (Fig. 23). When the user drags along
Figure 22: Choropleth map of In-Flows to each region. The Main Panel area of SEFLOW with four buttons providing access to the other four modules.

Figure 23: Controlling the range of a variable displayed by using two slider bars to set upper and lower thresholds to produce choropleth maps dynamically.
on the slider bars, the value of each interval is changing constantly through dividing the current range into 5 equal intervals. This allows the user to zoom in and take a closer look at the spatial variations of the variables in virtually any value range.

In addition, the user can process the raw data available in the existing data set, so that new variables are created. Computation of variables is done by selecting from pull-down menu the variables involved and the computation to be performed. Each user action is echoed by the program, confirming a selection and prompting the next action, or sending an error message in the case of incorrect user input.

**Cartographic Representation Module (CRM)**

This module visually explores the flow dataset using many different techniques, which include three cartographic representation methods for spatial flows, interactive visual information extraction, animated display, dynamic threshold setting, double view, blink comparison, as well as region collapsing.

There are two control panels: at the top of the frame and at the bottom. Together, they offer a rich array of options that the user may use to extract interesting information from the flow data base.

**Three representation methods**

Figure 24, 25, and 26 demonstrate the use of the line and number flow representation methods to display directional flows (Fig. 24), net flows (Fig. 25), and two way flows (Fig. 26).
Figure 24: Use lines & numbers to represent flows. Origins and destinations can be selected interactively and directional flows are drawn between them.

Figure 25: Net flows among selected regions using lines & numbers. Notice the position of arrow heads: if there are multiple flows into a region, arrow heads are drawn 80% of the way. Single arrow head is drawn at the end.
Figure 26: Two way flows using lines and numbers representation method. Lines are broken at the middle and drawn toward respective destinations.

Figure 27: Directional flows using proportional band width representation method. Notice that hollow arrow heads are used to increase clarity.
Figure 28: Net flows using proportional band width presentation. Also the displayed flow data range is between 6 to 91\% of the total range. Dragging the slider bars at the bottom control panel dynamically redraws flow maps according to the thresholds.

Figure 29: Two way flows using the proportional band width representation.
In this module, origins and destinations are interactively selected and directional flows are drawn accordingly (Fig. 24). When a group of regions are selected indiscriminately, then net flows and two way flows can be displayed amongst them (Fig. 25, and 26).

In the line and number representation, arrow heads are solid, and the numbers are printed on top of the flow lines using a constant font and sizes. Therefore, a small portion of a flow line is over-shadowed by the number. This representation is clear and informative only when the number of regions involved is quite small; it is most suitable for examining the exact flow magnitude for several regions.

Figure 27, 28, and 29 illustrate the implementation of the flow representation of arrow band width proportional to flow magnitude (O/D flows, net flows and two-way flows). A constant, maximum flow band width is used to display the largest flows, with the widths of other flows being determined linearly. In order to show two-way flows (Fig. 29), the flow band connecting the centroids of two regions is divided into two equal segments, with each segment displaying the flow from one region to the other.

This representation has a high information content because line width differentiates very minute changes in magnitude. The use of hollow arrow heads considerably reduces the overall weight of the presentation, making the graphics more clean and attractive. Nonetheless, if many flows are concentrated in the high end of the magnitude range, the maps generated become very cluttered.
Figure 30, 31, and 32 demonstrate the third representation method: constant width with graduated solid shading. The same strategy is used to exhibit two-way flows by drawing flows from the middle to destinations. Flow maps generated with this method is of substantial utility, especially in the case of a large number of flows, since the band width is strictly kept the same. Therefore the graphics tends to retain a clear overall pattern. The obvious drawback is the use of five intervals in differentiating flow volume, blurring the fine differences of flows. Five intervals were used because, after repeated experiments, it seemed that more intervals with different depth of the same color did not present easily perceivable differences between them.

All three representations can be drawn in three primary colors (RGB) and gray and in one of two drawing orders: default ascending sequence which draws the smallest flow first, and the opposite descending sequence. The default tends to retain a clear overall interaction pattern, while the latter emphasizes small flows.

**Dynamic threshold setting**

Figure 28 also exhibits the effect of changing the upper and lower thresholds. Two slider bars at the bottom control panel determine the range of the displayed flows. The screen display refreshes dynamically as the slider bars are dragged. SEFLOW actually redraws each new flow graphics in a pixmap, and then simply copies it to the screen so that the visual change is smooth and natural.

Since the pixmap copying operation is quite time consuming, the screen refresh speed is not as fast as the speed at which the user moves the cursor. Consequently there
Figure 30: Directional flows using constant band width with graduated solid shading.

Figure 31: Net flows using constant band width with graduated solid shading.
Figure 32: Two way flows using constant band width with graduated solid shading. Flows are drawn from the middle point to respective destinations.

Figure 33: Polygon collapsing. Polygons are selected interactively and collapsed into a megapolygon by clicking a button. Flows then reflect the interactions of the megapolygons.
is a temporal lag between the cursor's location and screen graphics display. Nevertheless, the locations of buttons on the slider bars which indicate the current data range are in unison with the graphics. Therefore the lag time does not affect the accuracy of the presentation.

Region collapsing

Regional aggregation of polygons is realized by selecting any number of contiguous polygons and collapsing them into one megapolygon. Interactions (net flows and two way flows) as well as directional flows can be visualized among the megapolygons (Fig. 33). The polygon boundaries inside a megapolygon are dimmed to very light black lines, while the boundaries of megapolygons are highlighted using thick, violet lines. The underlying data matrices actually are not reorganized. Flows between the megapolygons are computed at run time.

Double view and blink comparison

Using a pull-down menu, two images can be "dumped" into memory and saved temporarily. Then by consecutively clicking on a button, these two images of flows can be swapped out of and into the screen very quickly. As a matter of fact, the speed of swapping is virtually as fast as the user can click on the mouse. Therefore the user can adjust the speed and repeatedly compare and contrast two graphics, hoping to find something interesting.
Double view is realized by creating two windows for flow exploration (Fig. 34). When the "Split Window" button is pressed, the frame of the CMR layer expands to fill in virtually the whole screen with two drawing windows side by side. A button activates or deactivates each window. The control panels only affect the activated window, while the deactivated window does not respond to any event. The activation status of two windows can be changed by pressing the button. The user can then explore the flow data in any window and be able to compare graphics in two windows very conveniently.

In addition, the user can select spatial flows of certain time period by using a pull-down "Time" button to visualize and explore. This allows convenient traversal of the time domain.

**Forward Brushing Module (FBM)**

As one of the two mutually complementary components of the brushing technique, forward brushing regards the scatterplots of the aspatial domain as the primary image and the spatial expression in the form of spatial distribution and interaction as the mirror or secondary image. Brushing any of the scatterplot boxes would trigger multivariate linked mirror imaging, not only in the other scatterplot boxes, but also in the spatial domain.

The layout of forward brushing involves a two window format, with the scatterplot matrix on the right, and the representation of geographic space on the left (Fig. 35). Scatterplots are constructed by selecting variables from a pull-down menu and pressing a "Scatterplot" button. Scatterplots are stretched to fill in the whole left window, capable
Figure 34: Double view. Two windows are built to explore flows. They can be activated and deactivated with a button. Commands go to the activated window.

Figure 35: Default forward brushing. A circle is used as a highlighting device. The corresponding data points in other scatterplot boxes and geographic areas are dynamically highlighted.
of presenting two to nine variables selected from a pull-down menu in the current implementation.

Each variable in the scatterplots controls the vertical axis of its row and the horizontal axis of its column. For instance, in Fig. 35, box (1, 3) is the scatterplot of InFlow (y) and Income (x). The two variables defining a scatterplot box can be interactively identified by clicking a mouse button in the box. A pop-up window will appear with appropriate information describing the variables that represent the x and y axes.

Dragging or clicking a mouse button highlights the encircled points and the corresponding points in other boxes, with the corresponding map polygons highlighted. Highlighting of data points is done by changing the color to blue. In another version of SEFLOW, tiny circles replacing the points are utilized for this purpose. There are four choices of brush geometry: circle, square, rectangle, and ellipse. The size of the brush can be set by choosing "small", "medium", or "large" from the pull-down menu. The exact size of a brush also is affected by the number of variables involved in the scatterplots, i.e., smaller number of variables corresponds to a larger brush, and vice versa. Therefore, a medium size brush decreases in dimensionality as the number of variables increases. The size of a brush can also be interactively determined by selecting the upper-left corner of the brush and dragging down and right to define a bounding box. Then, the brush of desired geometry is drawn to fit the box.

The brush can also be rotated in 1, 5, 10, or 45 degree increment in both clockwise and counter-clockwise directions. This allows the brush be positioned in any direction and to cover a particular cluster with a specific orientation. This becomes very
Figure 36: Identify outliers by using a rectangular brush with a 45 degree orientation. Brush size can be determined by dragging on the screen. Several brush geometries are available: circle, square, rectangle, ellipse. (Forward brushing)

Figure 37: Forward brushing with the spatial interactions shown among the highlighted polygons. Net flows are displayed. Two way flows can also be viewed.
useful when covering all points in a perceived linear relationship and identifying the outliers (Fig. 36).

Brushing can be carried out in two modes: dynamic or static. The dynamic brushing refreshes display continuously while static brushing only redraws it after the brush stops. A button is made to easily shuffle to the backward brushing module.

Spatial interactions among the highlighted polygons can be viewed by selecting either net flows or two way flows between them. Fig. 37 shows the net flows among the six states. Any one of the three flow representation methods can be used.

**Backward Brushing Module**

The second part of the brushing technique is symmetrical to the forward brushing technique, in which the spatial expression of flow data is treated as the primary image while scatterplots of attribute data become the mirror image.

The default backward brushing highlights the brushed region and reflects the values of its attribute variables by enlarging the data points and enclosing it in a conspicuous blue circle (Fig. 38). This not only reflects the exact values of the region’s attributes, but also its value relative to other regions. The location of the attribute data point with respect to the data clusters reveals whether it is an outlier.

The highlighting of scatterplots is refreshed dynamically, as the bright-colored, circle-shaped brush moves in the geographic space. The center of the brush circle indicates the position of the mouse cursor.
Figure 38: Default backward brushing. A bright colored brush serves to highlight any polygon with corresponding attribute values circled in the scatterplots.

Figure 39: Anchor regions can be selected on the screen and defined as destinations with respect to the brushed region. Flows are drawn instantaneously as the brush moves from one region to another. (Backward brushing)
Figure 40: Anchor regions can also be defined as origins with respect to the brushed region. Flows are then drawn into the latter. Any of the three flow representation methods can be used. (Backward brushing)

Figure 41: The brushed region and anchors can be treated the same, then net flows and two way flows among them can be visualized. (Backward brushing)
Polygons of special interest to the user can be selected interactively as "anchors" to examine their interactions with the brushed region. The brushed region then can be defined as the origin or destination with respect to the anchor regions. Consequently, directional flows from origin(s) to destination(s) can be displayed dynamically as brushing is carried out (Fig. 39, and 40). Alternatively the brushed region and anchor regions can be treated as a group of interacting regions, to view the net flows and two way flows among them (Fig. 41).

Another visualization technique implemented is the ability to identify multiple regions and highlight their attributes. Instead of refreshing the scatterplots and spatial map continuously, brushed regions and their attribute data points simply remain highlighted to achieve a smooth string of changes.

**Pattern Searching Module**

This module is built based on the cartographic representation module to solve the problem of identifying an interesting spatial flow pattern and then searching in the database for similar patterns within the same time period or over the entire time span.

Figure 42 presents a two window environment, in which any window can be used to display the original flow pattern, while the other presents the search results. The search results are the top N most similar patterns. N can be adjusted interactively from 1 to 5. In Figure 42, a search through the entire time span indicates that the most similar flow pattern occurs in 1970-75 with a similarity index of 0.982 (in the right window).
Figure 42: Pattern searching over the entire time span. The left window shows the original flow pattern; the right one displays the search result. The most similar pattern has a similarity index of 0.982 in the 1970-1975 time period.

Figure 43: Pattern searching in the same time period. The right window shows the most similar flow pattern as having a similarity index of 0.931.
Pattern searching can be also conducted in the same time period if either the number of origins or destinations is one. In Figure 43, the most similar pattern is found (in the right window) to be from Oklahoma to the same set of destinations with a similarity index of 0.931.

**Conclusion**

The five modules of SEFLOW implement the basic methodologies of flow visualization and exploratory analysis. The combination of three representation methods prove to be clean, and effective to extract flow information. The use of graduated shading and proportional band width achieves clarity of conveying flow magnitude. Other techniques such as blink comparison, double view and dynamic threshold setting are quite useful in pursuing insights from rigid O/D matrices. On the other hand, the animated display of choropleth maps helps to present a clear picture of the underlying social-economic dynamics of the spatial interactions.

Moreover, the forward and backward brushing exhibit good capabilities of data analysis dynamically. The software responds to user's action instantaneously, which creates a smooth visual effect conducive to insight discovery. The pattern searching results in remarkably similar flow patterns, which indicates the mathematical equations of computing similarity indices are correctly formulated.

Nevertheless, the visualization system is only a proof-of-concept implementation using a sample data of limited size and complexity. In order to build a realistic system capable of dealing with a wide variety of very large flow datasets, much improvement is
needed in areas such as program robustness, capability of handling data of different sizes and formats, extended visualization techniques, etc. These will be further discussed in Chapter VI.
CHAPTER V

ASSESSMENT OF THE FLOW VISUALIZATION SYSTEM

Chapters III and IV discussed in some detail the conceived spatial flow visualization and exploratory techniques and theories, as well as the actual implementation of the SEFLOW system. It is time now to revisit the research critically, provide a comprehensive evaluation of the visualization system and answer several important questions. First, has the research accomplished what it was set out to do? Second, how well is the visualization system capable of solving the problem of exploring a large, complex flow dataset for the purpose of observing the hidden patterns and generating interesting hypotheses? What lessons can be drawn from this research experience that might be illuminating to future researchers engaged in endeavors like this?

This chapter will point out that while a preliminary assessment of the flow visualization system can be furnished based on a limited survey of potential users and the intuitive judgement from the developer’s perspective, a rigorous, systematic evaluation of the effectiveness of this visualization system is very difficult due to both the lack of well-defined methods and criteria for such system assessment, and the substantial amount of time and resources involved in such a process. These constraints are well beyond the reach of this dissertation research, and therefore the formal, structured evaluation will have to be undertaken in the future when substantial resources become available.

132
A Preliminary Assessment

This research was initiated to attack the problem of comprehending and analyzing a large spatial flow dataset. A combination of three inter-related approaches was proposed in answer to this challenge: scientific visualization, exploratory spatial data analysis and dynamic graphics. It supports a process of visual search and thinking within an interactive graphic environment in order to discover patterns, irregularities and obtain insight. The knowledge acquired from this process is expected to eventually trigger the generation of important hypotheses pertaining to the dynamics governing spatial flows, which, in turn, will form the basis for further confirmatory investigations. The research task consequently is to explore and develop theories and techniques of spatial flow visualization and integrate them in an interactive, user-friendly dynamic graphics environment. The goal of the experimental flow visualization system developed here is to serve as a navigational vessel through the oceans of complicated spatial interactions and social-economic mechanisms contained in large flow datasets. It should provide a set of visualization tools which will maximally expose the underlying structures and patterns so that the analysts can uncover them effortlessly. The visual tools must be integrated seamlessly so as to allow the analyst to easily observe different aspects and depths of the data utilizing well-targeted visual techniques.

From the system developer’s perspective, the research has satisfactorily accomplished its objectives given the significant constraints on both resources and time. The implementation was conducted within less than a year, including the system design,
software and hardware platform selection, detailed design, and coding under X Window Xlib and XView.

A wide array of visualization techniques and theories have been developed here to deal with spatial flow data. Ideas such as forward/backward brushing, mirror imaging, vertical/horizontal searching, judicious usage of traditional choropleth maps in a dynamic fashion, dynamic flow threshold setting, animated flow pattern display, good color usage of graduated saturation levels to clearly represent flow magnitudes, appropriate placement of hollow arrow heads to achieve clarity, etc. represent quite innovative and powerful ways of analyzing interregional flows.

These approaches and techniques were implemented and integrated into five modules with distinctive visualization functionalities: interactive flow information extraction, the dynamic linking of spatial and aspatial expression of flows, pattern searching, and the systematic examination of underlying social-economic dynamics. These modules were implemented within an intuitive, friendly and elegant graphical user interface (OPENLOOK). SEFLOW is the first full-fledged visualization system capable of handling complex spatial flow data. Its implementation appears to represent, for the first time in the history of spatial interaction analysis, the incorporation of good cartographic design and flow representation, innovative visualization and exploratory techniques and near real-time, dynamic graphics to forge an effective navigational system for spatial flow data.

The iterative nature of this research enabled the author to go well beyond what was originally anticipated when the proposal was written. Through repeated efforts, the
ideas and techniques were refined, improved and, in a few cases, eliminated, while a number of new ideas were created. For example, the positioning of the hollow arrow heads in such a way as to maximize clarity was the result of a number of experiments. Dynamic choropleth mapping, blink comparison and double views were also added to the system when the tests clearly suggested such implementation during the research process. Furthermore the overall performance of the program is substantially better than it was originally thought feasible in terms of graphics, user interface and speed as the X Window System turned out to be both very usable and very powerful.

During the research process, demonstrations were given to several faculty members and graduate colleagues in order to solicit criticism and suggestions. Several doctoral students were asked to use the program to explore the sample flow dataset. While improvements were made to the program based upon these comments, the responses were uniformly very positive as to the power and usefulness of SEFLOW in dealing with large, complex flow datasets and its potential applicability to other spatial interaction data analyses.

The intuitive, visual nature of this approach was quite enthusiastically received by some graduate colleagues as being attractive, straightforward and very useful, especially from those individuals with little background in GIS. They were excited to see the availability of such a software which enabled them to generate many frames of interesting flow patterns effortlessly, exercise their geographic expertise in understanding them, and to explore the complex spatial data painlessly with only a very short introduction. This ensures an easy integration of such visual techniques into geographical research on flows,
since the tool does not discriminate between geographers on the basis of their computer expertise.

However, as with any research on tools and methodologies, a rigorous, systematic assessment needs to be conducted in order to evaluate its effectiveness. This kind of testing on an exploratory system is very difficult because, as Tang (1993) pointed out in a very similar case, we lack a set of well-accepted criteria to judge the system against, and such an evaluation demands the availability of a pool of expert users and of many carefully selected, large flow datasets. Although such an evaluation is important in concluding the research, it is not feasible to undertake it as part of the present study considering the time and resource constraints. The following sections will elaborate upon this.

**Discussions on a Formal Assessment**

Judging the success of either a specific visualization method or a full-fledged visualization system remains largely an uncharted territory, because the field of SV and EDA is still at an early stage of development, and because of the unique nature of such exploratory tools. Most research has been concentrated on developing new and innovative visual thinking methods to tackle complex data, although the need to create a set of judging criteria has been strongly voiced by some experts in this field (MacEachren and Monmonier, 1992, and DiBiase et al., 1992).
Criteria for Judging a Visualization System

Discussions of judging criteria are presently very general and vague. For example, Becker and Cleveland (1991d) suggest that a good visualization method should allow the user to discover the underlying patterns "effortlessly, without attentive search." But exactly how can the degree of effort or easiness be measured? In addition, how can the success of insight-pursuit and hypothesis-generation be specifically evaluated? There does not appear to be an easy answer.

As MacEachren and Monmonier (1992) point out "the effectiveness of maps intended for exploration" should be treated entirely differently from maps intended for communication. Exploratory displays do not have a clear message to convey. They only try to maximize the opportunities for the analyst to capture the important structures that might be present and perceivable. This is in sharp contrast to traditional cartographic communication where we know exactly what to present but are only concerned with how to best achieve this objective. Therefore the somewhat well-established criteria for judging the effectiveness of cartographic communication are not applicable to visualization tools.

Furthermore, exploratory visualization is an evasive, unpredictable process. It is a free-style navigation with an unforseen sequence of events and outcomes, depending upon the particular combination of visualization operations performed, the characteristics of the dataset, and the state of mind of the analyst. We have little control of what is going to happen and face a complex dataset whose structure is unknown both to the analysts and the system developers. This uncertainty makes the design of a reliable testing procedure very difficult.
The Involvement of Expert Users

Adding to the difficulty of a rigorous testing is the need to involve expert users. The flow visualization system is not designed for novice users but for experts in the area of spatial flow analysis, who possess substantial knowledge and experience in this field. DiBiase et al. (1992) suggest that such users should be "highly motivated viewers" in order to achieve reliable pattern identification results. The same visualization technique may strike a expert user as being interesting and illuminating, but may be so complicated as to overwhelm a novice user and cause a great deal of confusion (DiBiase, 1992). Even among the experts, there may be different sensitivities to the same patterns and structures, and therefore the test results can be quite mixed. In order to achieve reliable results, we need a substantial group of spatial flow experts and we must find ways to deal with the normal deviations of responses among them.

The Selection of Large Flow Datasets

For an ideal testing work, the expert users should be provided with several realistic flow datasets which cover a reasonable number of hypothesis generation scenarios. Alternatively they may wish to use their own flow datasets. Therefore, we need to prepare the datasets and set them up according to each individual since SEFLOW is presently not designed to accommodate a variety of input data formats. The complexity of the datasets will definitely influence the hypothesis generation results. Consequently, comparisons of the results will be quite difficult.
In short, as it was argued by Tang (1993) in a very similar case, rigorous assessment of these systems demands a pool of highly motivated expert users with a substantial level of knowledge in spatial flow analysis, a well-established judging criteria to deal with a fundamentally evasive exploratory process, and a set of carefully selected, large test flow datasets of considerable scenario coverage. While a formal evaluation is imperative to systematically validate this type of visualization system, it is not feasible for this present study to arrange such expensive and time-consuming testing. The most thorny task is the establishment of the criteria and procedures for the evaluation based on a clear understanding of human perceptual and cognitive mechanism operating during data exploration and pattern identification, which itself would justify another doctoral dissertation.

Summary

The primary emphasis of the present research was to develop flow visualization techniques and to link them in an exploratory environment. With respect to this objective, SEFLOW has successfully fulfilled its mission. The preliminary survey has produced very optimistic responses to the system. However, it is recognized that a formal evaluation is required in the future to systematically validate this visualization tool.

Constructing such a flow visualization system has been a very exciting experience as it was improved and polished continuously through numerous trials. The visual nature of this research provided almost immediate gratification for the effort put in. There are two important lessons to be learnt from this development experience.
First, it is vitally important to select a powerful and flexible hardware platform and software environment. The author spent a considerable length of time in assessing the feasibility of the X Window System and other software such as ARC/INFO as a potential software environment. It was concluded that the latter was designed to support specific GIS applications and therefore presented too many restrictions as to what can be accomplished. In addition, visualization software attached to such a huge commercial software package often runs excruciatingly slow because of the amount of overhead involved. In order to construct a fast, friendly and flexible visualization system, it has to be a stand-alone program built on top of a powerful and extensible system, such as the X Window System which is designed specifically to support professional software development. Unix workstations should be the proper development hardware environment, since they are widely supported by many software development tools and thus the amount of work involved to develop an effective visualization system is much less than on a PC.

The second lesson concerns the design of efficient data structures and a data management system to support the manipulation of large flow datasets of various formats. Such considerations before the system is built are essential to ensure fast and efficient data import and retrieval, and robust and flexible coding of visualization operations in handling various flow datasets. To modify the hard-coded software in order to meet these requirements will be a quite time-consuming chore. The early investment in this aspect will prove to be well-worth the effort in the long run.
In next chapter, we will go one step further to summarize the findings of the research, and to examine its limitations that need further improvements. The future research directions of spatial flow analysis will also be discussed.
CHAPTER VI
CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

This dissertation research was designed to tackle the difficulty of understanding and analyzing a large, complex spatial flow dataset by constructing a dynamic visualization system. To this particular end, it is considered as successful given the good results of cartographic representations of spatial flows and other visualization and exploration techniques conceived and implemented. As a UNIX workstation based visualization software, SEFLOW represents the first systematic attempt to combining dynamic graphics, scientific visualization (SV), exploratory data analysis (EDA), and good cartographic techniques in the analyses of large flow data. Nevertheless, the theoretical significance of this research is not limited to flow analysis. Instead, it should be seen as an effort to bring a new kind of analytical methodologies into modern geographic analysis: utilizing the enormous power of modern dynamic computer graphics to visually analyze and explore staggering volume of spatial data in near real-time within a user-friendly EDA and SV environment.

However, SEFLOW is only a proof-of-concept implementation to experiment and demonstrate the underlying idea of visualization and exploratory techniques. It is not a perfectly designed and engineered piece of commercial software and therefore there is substantial room for further improvement. The experience of developing this software has
also triggered many sparks of thought about new techniques, future research directions, and expanding this research into a generic spatial interaction visual analysis tool.

In this chapter, results of the flow visualization research will be discussed, followed by a careful examination of the limitations of SEFLOW. Future research to enhance and expand it into a comprehensive spatial interaction analysis tool will then be elaborated. Furthermore, this chapter will examine the theoretical impact of this research on general geographic research. To conclude, this chapter will summarize the research experiences and point out future perspectives of the methodology adopted in the present study.

Results of the Flow Visualization Experimentation

SEFLOW is the first to incorporate three flow representation methods in a dynamic graphics environment. The results indicate that they are very effective tools in extracting spatial flow information clearly and cleanly. The array of visualization techniques employed in this software is a powerful tool kit in understanding and analyzing flow data. The research has made substantial contribution toward solving the representation problem and significantly broadened the avenues of exploring flow data by using techniques such as forward/backward brushing, interactive information extraction and pattern searching in a near real-time dynamic environment.

For the first time, population migration flows of a substantial volume (state to state movement for many time periods) can be effectively visualized and understood within a user-friendly computer graphics environment. Such a sophisticated system can be used by
any geographer with only several minutes of introduction regardless of his/her computer
background. Spatial analysts are no longer confined by the laborious process of manual
drawing of flows and tedious eye inspection of numerous O/D tables. The dynamic
threshold setting allows the analyst to focus, fine tune and close in for a systematic
examination of spatial interactions. The interactive, animated nature of the flow
exploration process, as demonstrated by forward and backward brushing, and other
techniques presents an entirely new way of looking at static tables of flows. The spatial
and aspatial aspects of flow data are linked dynamically to reveal the hidden relationship
and anomalies. It once again demonstrated the tremendous power and potential of
instantaneous visual feedback of dynamic graphics in capturing complex relations and
details in a unique fashion.

The five modules represent a rather comprehensive (though not exhaustive)
approach to flow analysis. The Cartographic Representation Module provides a rich
assortment of technique for viewing interesting flow patterns. The two brushing modules
enable a multivariate linked view of the underlying mechanism of spatial interaction.
Similar flow patterns can be easily searched from the database within the same time
period or throughout the entire time span in the Pattern Searching Module. The Main
Panel Area can be used to view areal variable such as social-economic characteristics. The
five modules complement each other with their distinct SV and EDA tasks centered
around attacking the complexity of spatial flows. Throughout the development work,
SEFLOW was repeatedly demonstrated to colleagues and several faculty members of
geography, and received very positive reaction in terms of its power and potential.
The choice of hardware and software environment proved to be an excellent one during the process of implementation. Sun workstation has respectful amount of memory, disk space and speed which are fully capable of supporting this type of work. It is well understood that the amount of memory and speed are crucial for graphics software development. The author was pleased to find out that the memory has not been a problem and that the graphics runs fast enough to maintain a highly interactive mode. The X Window System turned out to be a wonderful development environment: it is almost a programmer's dream comes true. It includes every basic tools and some advanced tools for the development of highly sophisticated software. The X protocol enables programs developed under X hardware-independent to be highly portable. The X library provides, in C binding, the most comprehensive basic tools to deal with any basic programming problem: user input, events, drawing, colors, fonts, sound, window management, etc. The availability of well-designed, elegant graphical user interface (GUI) toolkits (Motif, Olit, XView, etc.) substantially reduces the programming difficulty, leaving a programmer to worry only about more substantive problems such as program design, graphic programming techniques and the general research goals that the program is trying to achieve. Added to this is the capability of using a 3-D rendering package like Phigs or XGL, which opens the possibility of volumetric visualization of flow data.

A programmer's dream of a programming environment is one that utilizes the best building tools that modern computer science has to offer while providing him/her the completely free space to develop anything that is not available in the tool kit. It should not be an exotic environment, rather, it should be available in the widest range of
powerful hardware platforms, and better yet, it should be free. The X Window System
does just that. With C binding, Unix OS, comprehensive programming tools (a basic
routine library, GUI toolkits, 3-D rendering libraries), it is simply the state of art, as far
as public domain softwares are concerned. XView, which conforms to the OPEN LOOK
GUI convention, proves to be very handy in creating sophisticated user interfaces.

The theoretical significance of this work is really not how much SEFLOW has
achieved as experimental software, but rather that it unveils the great potential of this
visual analytical approach for the future spatial interaction and general geographic
research.

**Limitations of the Research and Its Implementation**

Although the research and the visualization system are considered as quite
successful with regard to the primary objective, limitations definitely exist in several
aspects. These are discussed below.

**Generic Data Input**

One of the major limitations of the current version of SEFLOW is its limited
capability to read spatial flow data and boundary data in various formats. Ideally, such
a visualization program should be able to read a wide range of O/D table formats and
boundary formats. However, since SEFLOW was originally conceived with the overriding
issue of experimenting with the visualization and exploratory techniques, therefore the
decision was made to reduce programming complexity by limiting inputs to a single
rigidly formatted O/D table and ARC/INFO UNGENERATED ASCII boundary files. Additional programming work is needed (approximately 15 days) to examine the most popular flow data formats and to design and implement a generic data input module.

**Handling Large Quantities of Flow Data**

Although SEFLOW has made substantial contributions to the flow representation and visualization problem, it is confined to a limited set of sample data involving only 50 states over 20 time periods (2500 entries a period). The question that remains unanswered is: how do we tackle a truly huge flow dataset with a large number of polygons, such as a 3000 by 3000 county to county migration table?

It is clear though that when the number of polygons exceeds a certain threshold (maybe 100), visualizing flows becomes very difficult because the limited screen space of a typical workstation (16" to 19" diagonally) will be too crowded with such spatial divisions. The speed of vector graphics generation would also be substantially reduced to a level that might prevent meaningful program-user interactions. However, the exact threshold remains to be determined, and there are potential graphics generation techniques to optimize the drawing and re-drawing, which merit further experiment. Moreover, specialized graphics hardware, such as high-end Silicon graphics workstations, also offers another alternative for fast and quality flow visualization.

Memory is another issue. Loading data directly to memory speeds up a program considerably, as opposed to having to read data from a hard disk. The current version of SEFLOW reads data into the memory immediately after the start of the program. A
workstation typically has at least 16 to 32 megabytes of memory, which is more than enough to deal with the sample data used. In addition, the amount of memory of a workstation can be easily upgraded to over 100 megabytes. This is a very respectable amount, capable of handling huge amounts of data. Nonetheless, even with sufficient memory, the program speed might still be slow because of the time-consuming computation and drawing associated with large flow datasets.

If we have to work with only 16 to 32 megabytes memory, data has to be read during the visualization operation contingent upon needs. A simple way to maneuver around a large dataset is to allow analysts to select interactively a subregion to visualize. The program then can zoom into this smaller region, load the relevant data into the memory and perform effective visualization and exploration operations. This undoubtedly will still slow down the program considerably, but nonetheless is a feasible strategy.

Another method would be to construct a hierarchical visualization strategy, such that the analyst could descend from more aggregated spatial structures to increasingly disaggregated ones, with a shrinking portion of the total geographic domain subject to effective exploration. The regionalization at each level should be accomplished freely by the analyst as desired, and the portion of the total geographic domain should be interactively selected by the analyst. Therefore, he/she could first try to visualize the whole geographic area composed of large polygons appropriately defined by the analyst, then select a subregion from the map, define new polygons at this level interactively for further exploration.
Handling Different Spatial Interaction Data

There are different types of non-network based spatial flow data. But SEFLOW only deals with area based flows. The possible situations would be point to point flows, like inter-city interactions or air traffic; flows between areas and points. These are very common forms of spatial interaction and should be included in the future development.

Efficient Data Structures and a Database Management System

Another aspect of data handling is good data structures and a powerful DBMS to manipulate flow data easily and effectively. For example, the program should allow the user to perform complicated computation and reorganization of the data. SEFLOW has only a very rudimentary data processing capability using basic arithmetic operations. It is incapable of complex data manipulation. Efforts should be made in the future to fill this vacuum.

Other Miscellaneous Problems

There are other improvements which should be mentioned here. The first is the map composition capability. During the process of spatial interaction exploration, an analyst may come across a important flow pattern and decide to retain and print it as a nicely composed map. Currently, it is easy to screen-dump the flow images, but impossible to compose maps for publication. Such composition functions include positioning and design of title, caption, and legend and scales.
The second is zoom-in and out functions. This would allow a user to define interactively the region to zoom in and blow up for further flow visualization. The zoom-out region can be defined on a temporary map of the whole area in a pop-up window that is displayed as the zoom-out function is activated.

Controlling some cartographic representation features by users might represent a third future improvement. For example, a user should be able to control the actual size of the constant band width of shaded flows, and also the maximum and minimum flow band width of the proportional width flow representation.

Finally, further work can be done to allow a user to control the brush size and shape in the backward brushing much the same way as the forward brushing. In addition to rotating brush by discrete degrees, free interactive rotation by dragging the brush to any angle should be added to the forward brushing. And more innovative flow pattern definitions should be explored.

Further Research

Further research should be directed toward two areas: network flow visualization and a generic spatial interaction analysis package involving visualization and exploration, flow modeling, and diagnostic statistical analyses.
Network Visualization

A major area of research is the case where flows are channeled through a specific network. This is quite different from the non-network based flows dealt with in SEFLOW. Three salient research questions can be identified here:

1. characteristics of the network itself, such as gateways, bottlenecks, physical network versus actual travel time network, etc.
2. characteristics of hinterlands of nodes, such as social-economic attributes, commuting populations, nature of the area (residential or commercial), etc.
3. characteristics of network flows: magnitudes, spatial distributions, temporal changes. Innovative visualization techniques appropriate to network flows are to be explored.

A Generic Spatial Interaction Visualization Package

SEFLOW has included three key aspects of general spatial analysis: aspatial aspects in terms of attribute data, spatial aspects in terms of spatial distributions and interactions. Flows are a very important research topic in geography, including not only realistic physical flows such as freight and migration, but also abstract interactions among geographic entities, such as attractions or push-pull forces.

Instead of concentrating on visualizing only area-based O/D tables and their aspatial attributes, SEFLOW has the potential to expand to a very generic spatial interaction visual analysis tool (Fig. 44). To achieve this goal, several key components
Fig. 44: The components and structure of a generic spatial interaction visual analysis package. Three major functions, dealing with general spatial flows in 2D and 3D representations.
must be added. It should include modeling capability, utilizing existing spatial interaction models and allowing users to generate new models.

Second, the modelling process should be connected with the visualization process to display the modelling results and to examine the validity of the model and modify interactively its formulation. The user can define an interaction model and have the results which are derived from the model displayed dynamically. Then judicious judgement could be exercised through visual inspection to see the validity of the model. The user can then modify the model interactively and see the changes in the results immediately, and then decide what further modification should be made, if any. Combining modeling with visual verification in an iterative process enables analysts to fine tune, calibrate and close in for the ideal model.

Third, for the purpose of data exploration, the visualization system should incorporate some diagnostic statistic analysis functions, such as computation of correlation matrices, frequency distribution, square of R and other simple and intuitive statistic indices. For example, the S-Plus statistical analysis system could be easily linked. This would allow some preliminary examination of the data from a formal statistical viewpoint.

Extending 2D to 3D Flow Visualization

SEFLOW is basically a 2D flow visualization tool. Extending it to 3D representation might greatly enhance its power to reveal spatial relationships and patterns. A 3D perspective view could be used to project flows through the 3D space, such as flow lines or bands taking off from the origins, flying in the air and coming down to the destinations.
from above. Viewing angle could be changed so that the analysts could look at the flow pattern from different perspectives or "fly" over the geographic space with a continuously changing 3D "bird's view". In short, adding the third dimension to the representation would give us more space to maneuver around to portray flows. A vivid 3D flow representation tends to present striking images that should leave a strong impression in the user's mind.

Brushing likewise can be three dimensional. In the current version of SEFLOW, scatterplots are on a flat surface. Three dimensional scatterplot cubes can add a time variable or any other relevant variable in the Z direction. Then the 3D scatterplot could be viewed by highlighting clusters of points in a transparent 3D space. Obviously, free rotation and changes of viewing angle should be allowed for optimal information exploration. The program should allow cubes of scatterplots be separated from the major cube and dissected and manipulated like an object for best observation.

**Implications for General Geographic Research**

Although the spatial data dealt with in this research are only flow data, the general methodologies which combine SV, EDA and dynamic graphics are very applicable to other geographic analyses. The key is to apply visual thinking techniques to explore complex spatial data in the modern advanced computing environment using dynamic graphics. Scientific visualization and exploratory data analysis have been developed specifically for the analysis of large datasets, and they are becoming increasingly powerful as faster and better computers and software avail themselves to data analysts. The most
salient feature of this approach is the dynamic, instantaneous and visual nature of the analytical process, which enables us to deal with very large, complex spatial data efficiently.

Another important aspect of this new approach is the user-friendliness. Unlike the traditional quantitative analysis which may be very sophisticated and is usually very demanding not only on the persons who design it but also on those who use it. This has set tremendous hurdles for many not-so-mathematical geographers, because some may believe it is out of their reach. This would obviously obstruct its acceptance and might even brew strong resistance. However, the visual thinking approach is different in that it is extremely demanding for those who design the tool, but extremely friendly to those who use it. This is because the fundamental underlying philosophy is to utilize our intuitive power to tackle the complexities of spatial data. It must be powerful and very user-friendly and easy to use. It does not assume any sophisticated technical background on the user's side in order to conduct the exploration effectively. This user-friendliness will definitely help the dissemination of the visual thinking approach.

Epilogue

There have been much excitement and joy as I embarked the journey of exploring the uncharted terrain of flow visualization. Much of such wonderful feeling comes from the realization that analyzing spatial, temporal flows is one of the very important, and challenging frontiers of geographic research, and also that the research methodology is closely linked to the wonder and "magic" of modern computer science and technology.
Utilizing the state of the art computer tools to deal with a substantive geographic topic is a good example of cross-discipline fertilization. The combination of SV, EDA and dynamic graphics should be a powerful tool not only to flow analysis, but also to many geographic problems. It is instantaneous, visual, intuitive and effective. Therefore it will be widely accepted by many geographers as a very important research methodology with great potential.
APPENDIX A
SOFTWARE USAGE

SEFLOW was implemented under the X Window System in order to demonstrate the concepts and methodologies of scientific visualization and exploratory analysis of a large flow dataset. This chapter discusses system usage in several aspects: the structure of SEFLOW, its functionalities in each of the five modules, the exact procedures of using these functions, user interface conventions, and some potential problems in the system.

The visualization system can be started by typing "seflow" in any command window under the Openwin window management environment of a Sun workstation. Then SEFLOW will take several seconds to initialize, load fonts, get colors, read the boundary file, attribute data of each region and O/D matrices, interpolate the intermediate O/D matrices and extrapolate the remaining 15 matrices from prior to 1955. Then the title page of SEFLOW will appear displaying pertinent information and a "CLICK TO CONTINUE THE PROGRAM" button (Fig. 45).

The SEFLOW graphical user interface conforms to the OPEN LOOK GUI conventions. Therefore if a user knows how to use any Openwin application, he/she should know how to utilize SEFLOW. The left mouse button is used to select things, such as activating a panel button, dragging on a slider bar, or selecting a point or a polygon in the drawing area. The right mouse button is used to bring up a menu. For instance,
Figure 45: A black and white screen dump of the title page of SEFLOW, displaying pertinent information about the software and the typical XView graphical user interface.
clicking this button in the drawing area will pop up a menu with available colors for
drawing the foreground. Alternatively a button with a tiny triangle in the control panel
area indicates a hidden pull-down menu, which can be brought out by clicking the right
mouse button on it. If the menu comes with a pin icon, then it can be pinned down on
the screen by clicking on the pin, and the menu can be moved around in the screen area.
The middle mouse button is not used in SEFLOW.

It should be pointed out that SEFLOW is a proof-of-concept implementation, not
a meticulously designed and robustly engineered software product. Therefore the program
may not behave in a well-polished fashion as it might.

Functions of the Main Panel Layer

The main panel layer appears immediately after the title page. It houses the
Dynamic Choropleth Mapping and Interactive Data Manipulation Module (DCM/IDMM),
as well as providing access to the other four modules of the program through four
buttons: "CARTOGRAPHIC REPRESENTATION", "FORWARD BRUSHING",
"BACKWARD BRUSHING", and "PATTERN SEARCHING" (Fig. 46). The functions
and usages of this module (DCM/IDMM) are discussed below.

Creating Choropleth Maps and Changing Thresholds

The nine attribute variables currently associated with each region are available
through the pull-down menu button "Choropleth Mapping". These variables are "total in-
flows", "total out-flows", "net_flows", "income", "crime rate", "unemployment rate",
Figure 46: The main panel layer of SEFLOW, housing the Dynamic Choropleth Mapping and Interactive Data Manipulation Module. It also provides access to the other four modules.
"population density", "centrality", "population potential". Viewing the spatial distribution of these variables helps to understand the underlying social-economic factors and spatial interaction mechanism.

Selecting any variable from the menu immediately generates a choropleth map of five equal intervals using graduated solid shading in different saturation levels of the same color. The right mouse button can pop up a "GC FG" menu (graphic context foreground, the color used in drawing) with a pin, which presents red, green, blue and gray as possible choices.

The bottom control panel features two slider bars controlling the upper and lower ("left" and "right") thresholds of the data range displayed in a choropleth map. Dragging the buttons on the slider bars dynamically refreshes the screen with a new choropleth map reflecting the new thresholds. Moreover, thresholds can also be set statically by toggling the small square buttons on the right, which sets the status of the slider bar to "static". Then the screen refreshes only when the user stop dragging. Variable values can be normalized by selecting on the pull-down menu button "Choose Values" so that the data range is from 0.0 to 1.0.

Interactive Data Manipulation

Variables can be manipulated through interactive computation and the resulting variable can be assigned to replace an existing variable. This allows the user to view the data from different perspective. For example, the original O/D matrix contains gross migration among states. Dividing it by origin or destination population might provide a
more appropriate look at the migration pattern. Similarly, dividing gross total in-flow, out-flow or net flow of a region by its population or its area might reveal more interesting migration magnitudes.

There are eight variables available for computation through "Select Operants" menu button: "total in_flows", "total out_flows", "total net_flows", "O-D flows", "origin population", "destination population", "origin area", "destination area". Common arithmetic operators can be chosen from the "Select Operators" menu button. The steps in computing a new variable are as follows:

1. Select the first operant from the operant menu;
2. Select a operator from the operator menu;
3. Select the second operants from the operant menu;
4. Select the resulting variable from the operant menu;
5. Click the "Compute" button.

Each selection is echoed and confirmed by a beep and a message of instruction of further action or/and an error warning, in order to make the operations of the process clear. Obviously the user cannot create a true new variable because the result is assigned to an existing variable, which is certainly a drawback. Generally speaking, the variable computation is not a very well-conceived operation.

**Functions in the Cartographic Representation Module**

This module is designed to allow users to extract and explore visual information from the flow dataset interactively using a variety of EDA and SV methods (Fig. 47). It
Figure 47: The Cartographic Representation Module of SEFLOW for extracting and exploring the spatial flows. This shows the user interface and a flow map using the lines and numbers method.
can access both the Main Panel Layer and the Pattern Searching Module through the "Back To Main Panel" and the "Pattern" button respectively.

Creating Different Flow Maps

Given a set of regional origins and destinations, SEFLOW can generate directional (O/D) flows between them. It also displays interactions among a set of regions in terms of net flows and two way flows.

**Directional flows:** The first three buttons ("Origins", "Destinations", and "Draw Flows") are used to produce directional flow maps. Following are the steps for doing so:

1. Click the "Origins" button, then proceed to select origins by clicking inside the polygons. A red dot then appears at the centroid of the polygon and program beeps to confirm the selection;
2. Click the "Destinations" button, then choose destination regions exactly as step 1;
3. Press the "Draw Flows" button to produce the flow map.

**Two way flows and net flows:** In order to view the interactions among regions, first click the "Select Regions" button, then proceed to select polygons. And then the user can generate two different maps by either pressing the "Net Flows" or "Two Way Flows" button.

If the user decides to select all polygons either as origins or destinations or a group of interacting regions, the "Select All" option present in the three polygon-selection
pull-down menu buttons ("Origins", "Destinations", "Select Regions") can be conveniently used.

Flow symbolization and drawing order: Cartographic representation methods for flows can be assigned using a pull-down "Flow Symbols" menu button. Three methods are listed in the menu: "Arrow Width" (arrow band width proportional to the flow magnitude), "Color Shading" (constant band width with graduated solid shading), and "Lines & Numbers". The default is set to color shading.

The "Drawing Order" pull-down menu provides a choice of descending or ascending sequence for drawing flows. The ascending order, the default, draws the smallest flows first, so that the larger flows always appear on top, overlaying the rest of the flows. To examine the small flows, the user may choose a descending order so that larger flows are overlayed by smaller flows.

Setting Thresholds Dynamically

The two slider bars in the bottom control panel area determine the upper and lower thresholds of the flow data to be displayed on the screen. Dragging on the slider bars dynamically refreshes the flow display, immediately reflecting the limits of the data range determined by the new thresholds. Alternatively, the user can type numbers into the threshold fields to change the flow map. Flow maps can also be refreshed only once slider bar dragging stops by selecting "static" from the menu of the small square toggle buttons on the right.
Collapsing Polygons

Regional aggregation is accomplished through the pull-down menu of the "Collapse Polygons" button. The user is advised to follow the exact sequence listed below:

1. Choose the "select polygons to collapse" option from the menu, then proceed to select contiguous polygons from the map;
2. Select the "collapse" option to aggregate the selected polygons. The program dims out the original regional boundaries and draws a thick, purple line to outline the boundary of the new megapolygon.

Now the new regional structure is ready for flow exploration. The "uncollapse" option will recover the original polygons.

Using Blink Comparison

The combination of a pull-down menu button "Save Images" and the "Blink View" serves to produce this function. Once an interesting flow map has been found, the user can save it through the "Dump To Image One", or "Dump To Image Two" menu option. After two images are saved, clicking the "Blink View" button will swap the two previously saved images onto and out of the screen, creating instant flashes of images for comparison and contrast.
Figure 48: The double view function of SEFLOW in the Cartographic Representation Module. Two display windows are used to explore the spatial flows. The user can activate any window and visualize flows.
Using Double View

Pressing the "Split Window" button allows the program to expand the windows to cover almost the whole screen area and presents two drawing windows (Fig. 48). The "Activate" button is used to toggle the status of the drawing windows between being activated and deactivated; the control panels are effective only with respect to the activated window. A user can explore the flow data set in either of the two windows. The "Merge Windows" button ends the double view representation and recovers the original window layout.

Selecting Flow Data From Different Time Periods

Flow data from several time periods are available for visualization and exploration by the pull-down menu button "Time" which lists spatial flow O/D matrices of "1955 To 1960", "1960 To 1965", "1965 To 1970", "1970 To 1975", and "1975 To 1980". Selection of the time period is echoed by changes in the bottom title of the flow map.

Functions in the Forward Brushing Module

The forward brushing module features a two drawing window layout, with the left window displaying the data in geographic space and the right presenting scatterplots. Brushing over the scatterplots highlights the corresponding data points in the other scatterplots as well as displaying the spatial distributions and interactions among the corresponding polygons (Fig. 49).
Figure 49: The Forward Brushing Module: user interface and window layout. The left window is used to highlight the corresponding polygons and the right window is for scatterplots and brushing.
Creating Scatterplots and Default Brushing

The "Select Variables" button presents a menu of nine variables currently available: "total in_flows", "total out_flows", "net_flows", "income", "crime rate", "unemployment rate", and "population density", "centrality", "population potential". Choose variables from the menu and then press the "Scatterplot" button to create scatterplots in the right window.

Dragging in the scatterplots automatically pops up a circular brush (default) as a highlighting device and the corresponding polygons are highlighted in gray (default). The shading color can be changed by bringing up the "GC FG" (graphic context foreground) menu from the drawing area and selecting the desired color.

Displaying Spatial Interactions

The "Select Spatial Flows" button allows the user to choose to display either net flows or two way flows among the highlighted regions by selecting from its pull-down menu. The interactions can also be turned off using this menu. For additional clarity of the spatial flow maps, the "Spatial Distribution" button will switch on or off the shading of the corresponding polygons.

Choosing the appropriate flow representation has to be done by going back to the Cartographic Representation Module to select "Flow Symbols", which is one of the inconveniences of SEFLOW.
Changing the Brush Size, Geometry and Orientation

The "Brush Size" button contains a menu to adjust the size of a brush. Three choices are available: "small", "medium", or "large". Notice that the size is also affected by the number of variables in the scatterplots (more variables corresponds to a smaller brush size). For example, the dimensions of a medium size brush decreases as the number of variables increases.

Alternatively, brush size can be determined using the "Interactive Sizing" button: press the left button in the right window and drag in down-right direction until the desired size is achieved then release the button.

Changing the brush orientation is accomplished using the "Rotate Brush" button. Its menu allows the brush to be rotated in a clockwise or counterclockwise direction in an increment of 1, 5, 10, or 45 degrees.

Furthermore, there are four brush geometry options available from the "Select Brush Geometry" menu button: circle, rectangle, square, ellipse. Each shape represents different definition of a cluster and may be assigned to contain distinct statistical meaning.

Other Functions

To help the user understand the scatterplots, the "Identify Boxes" menu button is available to provide information on each scatterplot box: choose the first item on the menu to start identifying, then click inside any scatterplot box. Then information
pertain to the X and Y axes of the box will be displayed in a pop-up window. Selecting the second item from the menu ends the identifying function.

Using the "Brush Mode" menu button, the user can change the brush action from dynamic to static. Selecting the latter will refresh the highlighting only after the brush stops moving.

**Functions in the Backward Brushing Module**

The backward brushing is the symmetrical operation of the forward brushing. Instead of brushing scatterplots, it moves the highlighting device on the geographic domain with corresponding data points highlighted in the scatterplots (Fig. 50).

The default backward brushing is done by simply dragging the cursor on the geographic map, which automatically shades the selected polygon and enlarges as well as encircles its attribute values in the scatterplots.

**Backward Brushing With Anchor Regions**

Spatial interactions can be viewed between the brushed region and anchor regions. This is done by first selecting anchor regions, then defining the brushed region with respect to the anchors, finally selecting flow types if appropriate.

1. Select anchors: Press the "Select Other Regions" button to notify the program that the user is starting selection of anchors, then proceed to select anchors, finally press the button AGAIN to end the selection.
Figure 50: The Backward Brushing Module: the user interface and screen layout. Contrary to the forward brushing, a circular brush is dragged along the geographic map, highlighting the polygon and its attributes in the scatterplot.
2. Define the brushed region: Use the menu of the "Define Brushed Region" to define it as an origin or a destination with respect to the anchors. Then directional flows can be drawn between the brushed and anchor regions. Alternatively, the brushed region is simply defined "As One of the Selected" so that it is no longer different from the anchors, then flow types can be chosen, as explained in next step.

3. Select flow types: When the brushed region is defined "As One of the Selected", it is lumped together indiscriminately with anchors. Using the "Select Spatial Flows" menu button, the user can define the flows among the selected regions as either net flow or two way flows. However, the default is set to net flows.

Identify Polygons

Instead of highlighting one polygon at a time and refreshing the display continuously, the user can choose to identify multiple polygons and their attribute values by accumulating highlighting without refreshing. This allows the user to see how the screen information changes as more regions are added and highlighted.

This is accomplished by pressing the "Identify Polygon" button to start, then selecting polygons for highlighting, and then pressing the button again to end the identification function.
In order to increase the clarity of display, the highlighting circles can be switched off using a menu button "Circle Switch", as a large number of circles would clutter the scatterplot display.

**Functions of the Pattern Searching Module**

Pattern searching can be done in either single display window mode or double window mode (Fig. 51). The user interface is extended directly from the Cartographic Representation Module. The idea is to search for similar flow patterns from the database once an interesting pattern is found during the process of flow exploration. Any of the two windows can be used for the exploration purpose, or for displaying the search results.

Search is conducted simply by clicking on the "Search" button. The "Display" menu button contains the search results from the most similar to the fifth similar, as well as the original pattern. Selection on the menu automatically generates flow maps in the activated display window.

The number of similar patterns searched may be determined by moving the up-and-down arrows associated with the "No. of Patterns" button. "Search Methods" determines whether search is conducted in the same time period or throughout the available time span. However, if the original flow pattern is many to many flows, net flows or two way flows, search can only be accomplished by looking over the whole time span.
Figure 51: The Pattern Searching Module user interface in two window mode. The user can explore flows in any of the two windows, and search the database for similar patterns, then display the search results in the other window.
Summary

SEFLOW is implemented under the X Window System using XView as the graphical user interface. As a dynamic visualization system, it has very consistent, elegant interface and fast graphics which permit easy, smooth navigation and exploration of a spatial flow data set. Many innovative SV and EDA techniques are implemented and organized for effective hypothesis generation purpose. Since the GUI conforms the OPEN LOOK conventions, anyone with some experience of using Openwin applications should feel very comfortable with SEFLOW. Since it is an experimental software, it may not behave in a well-polished fashion as it might.
APPENDIX B

The X Window System

The X Window System

The X Window System is one of the most exciting and most important development of software engineering technologies in the recent decade. The first release of a full-fledged system occurred in September 1987 by the Massachusetts Institute of Technology. It was the version 11 of the X Window system (Release 1), widely known now as X11. Since then MIT has made several releases incorporating revisions and enhancements: Release 2 (March 1988), Release 3 (February 1989), Release 4 (January 1990), and the most recently Release 5 of X11 (Nye, 1990a).

The birth of the X Window System promised to bring fundamental change in the workstation world. Currently it is the de facto workstation industry standard, which may bring about an explosion of X-based software. The popularity of the X Window System rests on several factors (Nye, 1990a):

1. It is hardware independent. X Window applications are portable across different workstations without substantial additional programming. They can even run on a high-powered personal computer.
2. It is a network-oriented window system. Programs can be run over the network and the results can be displayed locally. This opens the possibility of sharing resources on a network of computers of various power.

3. It is one of the most comprehensive and most powerful window system in workstation environment, which supports 2D and 3D applications and offers a wide variety of well-designed graphical user interface toolkits.

4. It is also fully extensible. While the X subroutine library and X protocol will remain stable for a period of time, the X Window System is designed to support additions to it. Moreover, future development is controlled by the X Consortium, an association of nearly all major workstation manufacturers, many famous universities and software developers. This ensures the quality of future development and its widest possible acceptance and support by the workstation industry and the user community.

Some Basic Concepts of the X Window System

The architecture which enables X applications to run on a network is the server-client model (Nye, 1990a). The program which controls the resources of each machine is a server, and the programs running under any server are clients. Each machine has a server, which is not only accessible to local clients but also to the clients from any other machine on the network. A server is responsible for the communication between the clients and its local resources.
A special client running under the server of a workstation is called the window manager, which is responsible for enforcing window layout policies determined as appropriate by its developers and controlling the general appearance of the applications. The X Window System itself is independent of any particular window management style because its principle is to provide mechanism, not the policy. Instead, it only provides some sample manager programs like *uwm* and *twm*. This leads to the existence of many distinct window managers conforming to different GUI policies by many different vendors.

X Window applications are all "event-driven" programs. An event is anything significant that the server and applications should be informed of. These include user actions from a keyboard or mouse buttons, and interactions among clients. For example, if a window which was originally overlapped by another window is exposed, the program should be notified of this event and perhaps decides to redraw the content of the exposed region. The X Window System is designed to allow a client to listen to user actions and respond promptly to them. This is done by the server stacking up events in a chronological order and the program querying the stack of events in a manner defined by the application program. As Nye pointed out: "Events imply a philosophy that the program should respond to the user's actions, not the other way around" (1990a).

The Hierarchical Structure of the X Window System

The X Window System is a multi-layered programming environment residing usually within a Unix operating system. Fig. 52 illustrates the structure and the relation
The Hierarchical Structure of the X Window System

Figure 52: The multiple layers of the X Window System. Applications sit on top of the system and have full access to any layer. Based on Nye, 1990a, and Sun Microsystems, 1990d, Xt Intrinsics Programmer's Guide.
of an application program to different layers of the system. At the bottom is the X subroutine library (Xlib) which fully utilizes the power of the X protocol and performs extensive functions ranging from querying events, manipulating windows and controlling colors and drawing graphics. The Xlib does not make X programming easy because it performs very extensive yet very basic tasks. A programmer has to deal with virtually everything from assigning window attributes to advanced graphics generation, which makes X window programming overwhelmingly complex and error prone. To alleviate this problem, the graphical user interface toolkits (such as XView, Motif, and Olit, Xaw) are created to modularize the program and handle the interface part of the program. This substantially reduces the programming difficulty by taking away the burden of creating quality user interface from the scratch and by making the program event-handling structure very clear and straightforward.

On top of the GUI toolkits layer lie the 3D graphics routine libraries such as Phigs and XGL, which provide some very useful graphics primitives. The existence of 3D rendering and transformation pipelines makes graphics programming much easier. Finally comes the application on the top of the X hierarchy, which has full access to any layer of the X system. The beauty of it is that the system is entirely transparent to the application, and therefore this feature enables a program to develop any new tool if necessary without being confined by the existing capabilities of the system.
A List of Important X Window References

There are many well-written X window programming and reference books available. But the most comprehensive, definitive is the series published by the O'Reilly & Associates, Inc. They provides indepth discussions of every aspect of the X Window System and numerous examples. Here is a list:

Volume 4 & 5:  X Toolkit Intrinsics Programming Manual,
X Toolkit Intrinsics Reference Manual
Volume 6:  Motif Programming Manual
Volume 7:  XView Programming Manual and its companion book:
XView Reference Manual
Volume 8:  Phigs Programming Manual
Quick Reference:  The X Window System in Nutshell
APPENDIX C

The OPEN LOOK Graphical User Interface Conventions

In an effort to define a consistent, simple and efficient graphical user interface style for applications, Sun Microsystems and AT&T jointly developed the OPEN LOOK GUI conventions which establishes a complete set of guidelines of GUI design, and specifications of the general "feel and look" of OPEN LOOK compliant applications.

The underlying idea of the OPEN LOOK GUI conventions is that if applications have a consistent look and feel conforming to a well-designed GUI specification, users would feel very comfortable in learning new applications once they are familiar with that GUI specifications. And users are more inclined to explore many other programs without spending much time getting used to a new GUI style. This could significantly speed up the users' learning curve and substantially reduce the fear and anxiety typically associated with learning an entirely new and complicated application.

Not only does a GUI have to be consistent over a wide spectrum of applications, but also it must be simple, intuitive, elegant and efficient. It should give a novice user a feeling of comfort and clarity. It should not overwhelm the user by an overly complex design and a too flashy outlook. It should also provide convenience for advanced users to explore more sophisticated functions. The OPEN LOOK GUI is designed to do just that (Sun Microsystems, 1990a).
The feel and look of the OPEN LOOK GUI is defined by establishing the exact appearance of different visual control objects, and the manners in which a user interacts with them (Sun Microsystems, 1990a). It also defines the standard usage of mouse buttons, pointers and a keyboard, etc. For instance, it supports the use of three button mouse and gives each button an explicit functionality: the left button for selection, the right button for bringing up menu, and the middle button for adjusting. Control objects are required to have the same looks and usages in all applications: a simple button is used to perform an action; a menu button with a distinct little triangle indicates it hides a pull-down menu, a button with a "..." sign implies it will bring up a dialogue menu once activated, and a scrollbar has several parts, each performing distinct functions. Typical window manipulations are also defined regarding resizing, closing, opening, panning and scrolling. The organization of workspace, icon characteristics and color usages are also specified succinctly (Sun Microsystems, 1990a).

Conforming to the above GUI specifications does not guarantee an efficient, intuitive and elegant GUI. Therefore, the OPEN LOOK GUI provides a complete set of design style guidelines. The basic principles are (Sun Microsystems, 1990a):

1. simplicity in visual design, labelling of control objects, and the appearance of visual objects;
2. consistency in mouse and keyboard usage, the placement of control objects, and the manner in which they are manipulated;
3. efficiency such as progressive disclosure, appropriate use of pop-up windows and menus;
4. coexistence with applications of other GUI conventions.

These principles provide a general guidelines of GUI design. For example, the progressive disclosure requires that a GUI should present most commonly used functions conspicuously while hiding advanced options so that a novice user would feel comfortable working with it without being frightened or overwhelmed, and an experienced user can still conveniently experiment with more flashy and complicated functions (Sun Microsystems, 1990a).

The guidelines also includes specific suggestions as for when, what and how to use different OPEN LOOK GUI features. This helps programmers in designing window configuration, loading and manipulating data, naming control objects, choosing colors, displaying messages, etc. (Sun Microsystems, 1990a).

The Openwin window manager running on Sun workstations and its utility programs all conform to OPEN LOOK GUI convention, so does the SEFLOW. The following is a list of useful references:

APPENDIX D

The XView Graphical User Interface Toolkit

In order to modularize X Window programs and alleviate programmer's burden of developing high quality graphical user interface, various software developers have created half a dozen toolkits for this purpose. Most notably are Xt (DEC/MIT), the Andrew toolkit (IBM and Carnegie_mellen University), InterView toolkit (Stanford) (Nye, 1990a), Motif (OSF), Olit and XView from Sun Microsystems. However, only Xt, which includes a Xt Intrinsics and Athena widget set is included as a standard part of X11 release (Heller, 1990).

Athena widget set is designed by MIT only for demonstration purpose. Therefore it is not quite suitable for advanced software development. Currently, there are three most popular toolkits: Motif, XView and Olit. Motif toolkit is created and supported by OSF (the Open Software Foundation) conforming to the Motif GUI specification, while XView and Olit are designed by Sun Microsystems, Inc., compliant with the OPEN LOOK GUI specifications.

XView (X Window-System-based Visual/Integrated Environment for Workstation) was developed on the basis of the application programmer's interface (API) for the SunView window system supported by the Sun Microsystems. It serves dual purposes:
to convert old SunView applications into X Window applications and to develop new programs running under X11 (Heller, 1990).

Unlike other GUI toolkit widget sets (such as Athena, Motif, Oliit etc.) for the X Window System, XView does not build upon the Xt Intrinsics layer. Instead, it is directly based on Xlib. However, similar to others, it provides a complete set of pre-built graphical user interface objects and a mechanism through which the core of an X program is linked with the GUI. These objects are frames, canvases, control panels, panel items, pull-down menu and pull-right menus, scrollbars, notices, icons, and other non-visual objects. Callback functions are registered with each object in the control panel area and triggered when the corresponding object is activated. Event procedures are used to respond to any other user actions and incidence. XView also provides functions to simplify X window programming in dealing with colors and fonts, writing helps, coping with selections, drag and drop operations, etc. (Heller, 1990).

A conventional X program contains a complex event handling loop, which forms its core. It continuously queries the server for events, and processes the events, usually by calling subroutines, until an end of program event is detected. However, applications written under XView is quite different from this program flow, because XView is a notification-based system. It installs a notifier which reads events and processes them, and then dispatch them to pre-registered callback functions or event procedures. (Heller, 1990). By doing so, it takes over the complex control loop of an event-driven program, reducing the programming task to simply writing subroutines of callbacks and event procedures. The notifier does quite a bit of work by controlling the program flow, pre-
processing events, screening for appropriate events to be dispatched to the pre-registered callbacks. This removes a very complicated overhead structure of the program control from the application program, considerably reduces the complexity of an XView application.

In short, XView not only provides a comprehensive set of visual objects for constructing graphical user interface, it also simplifies the program by offering many functions to deal with important things like allocating colors, writing help, handling fonts and resources. In addition, it modularizes the application by taking over a complex overhead of the main control loop, leaving application programmers to cope with callbacks only. The following is a list of important references:


BIBLIOGRAPHY


Sun Microsystems, Inc. 1990a. OPEN LOOK Graphical Interface Application Style Guidelines.


____ 1990c. OPEN LOOK Graphical User Interface Trademark Guide.


Tukey, John W. 1977, Exploratory Data Analysis. Addison-Wesley Publishing Company, Reading, MA.


