255
96

COMPUTERIZED SIMULATIONS FOR INTRODUCTORY GEOGRAPHY INSTRUCTION: MENTAL MAPPING

by

Vincent F. Miranda

Project submitted to the faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

APPROVED:

Dr. Clifford A. Shaffer, Chairman

Dr. Lawrence W. Carstensen

Dr. Edward A. Fox

March, 1991

Blacksburg, Virginia

COMPUTERIZED SIMULATIONS FOR INTRODUCTORY GEOGRAPHY INSTRUCTION: MENTAL MAPPING

by

Vincent F. Miranda

Committee Chairman: Dr. Clifford A. Shaffer

Department of Computer Science

(ABSTRACT)

The study of geography is an important component of a university education. While computer-aided instruction (CAI) techniques offer good means of satisfying the rapidly growing demand for instruction in geography, traditional approaches to tutor construction appear to be of little use in meeting the special requirements of an application based on complex graphical manipulation. This project focuses on the design and implementation of a Mental Mapping Module which explores the mental maps of students by asking them to locate various places in the world by pointing at positions on an outline map drawn on the computer screen. The computer then warps country borders in relation to errors in placement to show the student's "mental map" of that country. The module also gathers data from students on their perceptions of place characteristics such as population size, climate and cost of housing. The interactive graphics design approach for the module is implemented using the X Window System, which allows a "point-and-click" interface with graphical and textual cues to initiate operations. The Mental Mapping Module forms one of the six computerized laboratory modules applicable to several introductory geography courses that have been proposed in project GeoSim: CAI software for the teaching of introductory geography.

ACKNOWLEDGEMENTS

First, I would like to thank Dr. Clifford A. Shaffer for his valuable support and suggestions during the research and writing of this document. His comments and criticisms were constructive and helpful. I would also like to thank Drs. Lawrence W. Carstensen, Jr. and Dr. Edward A. Fox for serving on my committee. Dr. Carstensen has been a source of many fruitful discussions and his constant willingness to help is appreciated. Dr. Lenwood Heath and Sriram Pemmaraju helped me with the triangulation algorithms and I thank them for the same. Thanks are also due to Tom Bailey for providing me with timely data. I owe a great deal to my parents and sisters; without their encouragement and sacrifice I would not have been able to achieve my goals. Working on this project has been an enjoyable experience especially due to a fun group of people at the Quad Tree Lab; Pat Brown, Sheryl Kriss and Mahesh Ursekar have lent support, good humor and outrageous behavior. I [illegible] had many stimulating discussions with Naganand Doraswamy on chips, life, and home cooking, and our friendship will endure long after we both leave this place.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

The study of geography is an important component of a university education. Geography is unique among disciplines in its focus on the spatial aspects of physical processes and human actions. While understanding historical and contemporary events requires knowledge of geography and geographic processes, dozens of studies point out the incompetence of U.S. students in geography [Cars90]. Recognition of the need for improved geographic education is resurging throughout the United States, but is still far behind many other sciences in both universities and secondary schools. Many geographic processes are dynamic, and thus difficult to demonstrate in a traditional classroom setting using static media. At the same time, the manipulation of geographic information by computer has been studied intensively, with many models of geographic processes well understood. Thus, computer simulation of dynamic geographic processes can supplement geography education at all levels of the curriculum. In addition, the use of computer aided instruction allows for decision making and interactive learning by the students.

Members of the Departments of Geography and Computer Science at Virginia Tech have initiated Project GeoSim: computer-aided instruction (CAI) software for the teaching of introductory geography [Shaf90]. GeoSim applies the immense capabilities of Geographic Information Systems (GIS) and simulation to the teaching of geography, beginning at the first geography course. In this manner, the most exciting aspects of geography are made available to the widest audience, and in a manner that stresses dynamic processes over static information.

Our goal is to provide computer-based educational materials for geography that are usable by a wide range of students. A series of computerized laboratory modules applicable to several introductory geography courses have been proposed. These modules are designed to meet several criteria [Cars90]:

1. They must be highly interactive, allowing students to make decisions and manipulate geographic data in a way that encourages learning while keeping the student's interest.

2. The modules must be easy to understand and highly graphical—naive computer users must be able to use them with virtually no training.

3. An effective geography tutoring system must be supported by a Geographic Information System as well as a database system to allow for sophisticated manipulation of geographic data, provide tailoring of modules to the geographic location of the class, and support creation of future modules.

4. The system must run on equipment of as low a cost as possible (under the constraints of the previous criteria).

5. Finally, the modules must relate to the student, i.e., exercises should include real-world data and have sufficient flexibility so that the exercise can be done using data for the student's own city, state or country.

Based on these criteria, the following six modules have been designed.

Mental Mapping: This module explores the mental maps of students by asking them to locate various places in the world by pointing at positions on an outline map drawn on the computer screen. The computer then warps country borders in relation to errors in placement to show the student's "mental map" of that country. Questions on attributes such as population size, cost of housing, and climate of the places are also asked. The computer then provides feedback on the actual values as compared with the student's answers.

Nuclear Accident: This module allows the student to explore the consequences of a nuclear accident at any power plant in the world. The module simulates the movement of the radiation cloud, and computes the costs to human populations, agriculture, etc. for areas affected by radiation.

Reduction of Commuting Time: This module has the student determine optimal paths for automobile commuters living in the suburbs of a major city to arrive at their work places in as short a time as possible. It allows the student to plan the routes by associating neighborhoods with a series of roads, or creating new roads to reduce bottlenecks. The computer then reports on the success of the student's plan.

Population: International: This module allows students to explore a database to compare countries by population growth and age distribution. The student is able to investigate the effect of changing birth and death rates, fertility ratios, etc. on the population statistics of selected countries.

Population: Migration and Political Power in the United States: This module allows students to relate migration patterns among counties of the U.S. to data on place characteristics. It simulates past trends in migration among counties based on census data from the 1960-1990 period in the form of animated maps of the U.S. by county that change population categories over time. Those trends can then be extrapolated into the future based on weightings selected by the student. In addition, the module allows the student to rank counties in a manner similar to that used by "Places Rated Almanac." [Boye85]

Orientation, Position Finding, and Orienteering: This module teaches map orientation skills [Cars90]. The instructor sets up the desired course using a map of the orienteering terrain and placing markers to define the course. The number and location of the markers is determined by the instructor. Instructions are displayed on the computer, and the student begins to find her way around the course with the aid of a hand held map and information presented on the computer screen. Eventually, the student locates the first marker, proceeds to locate the next, and so on. At the end of the session, the time taken by the student to cover the course is recorded and displayed to the student.

To broaden the use of GeoSim modules, the modules are planned to be implemented for three tiers of hardware support. *Tier I* will assume a PC-class machine such as an IBM PS/2 or Macintosh II. Best results will require a color monitor, but basic capability will be possible for most modules using only monochrome monitors. A modest hard disk drive (with 2-3 megabytes of free storage) and a mouse are also required. Module software and databases will be distributed by floppy diskette for *Tier I*. *Tier II* assumes all of the equipment of *Tier I*, supplemented by either a CD-ROM or videodisc drive. Module software and databases will be distributed by either CD-ROM or videodisc as requested by the client. *Tier II* implementations will take advantage of optical storage technology to provide extended geographic databases and visual-based tutorial sections. Finally, *Tier III* will assume *Tier II* hardware plus the availability of Intel's Digital Video Interactive (DVI) multimedia system [Luth89]. This technology allows for real-time decompression and presentation of digital audio, still images, and video clips on the computer, along with a wide range of special effects.

A wide variety of personal computers and workstations will be able to run these modules since implementations for each of the following three popular interface systems will be provided. Microsoft Windows is widely used on IBM PC's and compatibles running either MS-DOS or OS-2. Apple's native programming environment for the Macintosh (known as the Toolbox) allows applications to run on all models of the Macintosh. Finally, the

X11 windowing system (or one of its extensions: DECwindows, Motif, and Openlook) is available on virtually all UNIX workstations. The X Window System is already widely used for educational purposes at MIT as part of Project Athena [Murm89] and at Carnegie Mellon University.

This paper describes the prototype design and implementation of the Mental Mapping Module at the *Tier I* level using the X Window System. The project design was developed by Drs. Lawrence Carstensen and Clifford Shaffer [Cars90]. The author along with the two designers was involved in the implementation of the prototype. The remaining part of the document is as follows. Chapter 2 explains the concept of "mental mapping" and its importance to geographic education. Chapter 3 presents the functional design and Chapter 4 discusses interface issues. Conclusions and ideas for future work are presented in Chapter 5.

Chapter 2

MENTAL MAPPING

2.1 General Description

The purpose of this module is to explore the "mental maps" [Goul74] of introductory students and to teach needed information concerning world cities [Cars90]. The module serves both as a diagnostic tool to test the student's knowledge on places (hopefully motivating the students' desire to learn more) and as a tutor[1]. We first gather data from students on their perceptions of place locations and characteristics. The instructor selects a region of the world and an outline map of that region is drawn on the computer screen. The students are asked to locate cities by pointing at positions on the outline map. Place names from the region are presented to the student in random order (e.g., "Locate Cancun"). The student then moves a cursor to the perceived location of that place within the framework of the map outline. Because knowing the location of a place is only one step in building geographic knowledge, the data collection module next queries the student concerning certain characteristics of that place (i.e., population, climate, and cost of housing).

After a set of 5-20 places has been described by the student, the computer provides feedback on the locational accuracy of the responses in several ways. Distance errors are

1 Quite a few faculty and students (about 25 totally) have come in to "test their wits" against the "geography game".

illustrated on the screen by plotting correct locations corresponding to each place connected by a line to the student's selected location. The actual city will then be "rubber-sheeted" to a best fit with the cities as located by the students. The rubber-sheeting procedure creates linear transformations of local regions which are then applied to the outline map so as to warp the map framework itself and show the student's "mental map" of the region studied. This display includes both the actual shape of the outline and the warped outline for comparison. This analysis is significant because humans are very adept at remembering geographic shapes [Keat82]. More distorted portions of the mental map indicate a lack of knowledge that needs to be corrected. The visual display provides an overview of locational error that is not possible given a table of distance errors alone.

Feedback about characteristics of places is displayed on different screens. Each screen contains an outline map of the country in question along with the actual values and the students' responses for characteristics such as population, climate and cost of housing.

Performance data for each student are stored during the student input phase of the module to allow for assessment by the instructor. Aggregate performance for the entire class can be shared with the students in a subsequent class session. Errors in locations can be summarized, reasons for those errors can be discussed. Students can compare their personal mental maps with those of their peers, and enhanced understanding will result.

2.2 Prototype Description

The implemented prototype allows the students to use the exercise with data from the U.S., India or the Middle East. Necessary data for the outlines has been taken from Goode's World Atlas [Espe88]. The module is designed to run in any X-Windows based environment, having either monochrome or color displays, although it was created on the Macintosh II running A/UX. The interface is window-based and menu-driven with context-sensitive operating instructions. The student's input is via the mouse, and there is no keyboard input. Our intention is to make the module easy to use by students with little or no prior computer experience. All the software has been written in the C programming language [Kern88]. The prototype has four phases of operation: student input, computation, graphical display of results and storage of student responses. These phases will be common to all the proposed modules. The interface has a Macintosh-like "look-and-feel" which is again, common to all modules. This allows easy transition from module to module and, most importantly, focuses the students' attention on the subject matter, rather than on understanding the interface.

2.3 Educational Importance

This exercise provides a valuable introduction to several of the key themes [Cars90] of geographic education: location (position), place (physical or human characteristics), and region (formation of regions of similar characteristics and distribution of these characteristics). This exercise allows students to obtain feedback on the quality and accuracy of their "mental maps" and also to comment on several characteristics of places that should be at least marginally familiar. It illustrates to the student how he or she views the world, and the distortions inherent in that view, helping to correct misconceptions. Aggregate results for a class illustrate both significant similarities and dissimilarities among a group of people.

Chapter 3

FUNCTIONAL DESIGN

3.1 General Description

The Mental Mapping module is an interactive, menu driven, window based software application. The input data for the map outlines will eventually be derived from the public domain World Data Bank II created by the CIA [Cars90], although the prototype maps were hand digitized. An outline map for the prototype version is represented by a set of pairs of (x, y) coordinates (refer to section 4.5.1 for file format descriptions). The drawing program connects the first point to the second, the second to the third and so on, by straight lines. For capturing the intricate contours of the map outline, it is desirable that these points be very close to each other. Alternatively, the instructor may create her own exercise by digitizing a simple map of any desired area, conforming to the input file format described in Section 4.5.1. The input map coordinates are scaled suitably to fit into a 570×365 pixel display window on a computer screen. The locations of cities on the map are stored in a second ASCII text file along with the values of various other attributes such as population size, maximum and minimum temperatures, and cost of housing. The coordinates of the outline map as well as the locations of cities within the map are read into memory and stored in suitable data structures (see Section 4.5). For each city, the specified attribute data are also read and stored. The number of cities to be queried on can be determined by the instructor before the session begins. The instructor is also provided with a facility to select the attributes (population size, climate, etc.) for the session[1]. During each session, the proper number of cities are randomly selected from the city database. Thus, each session is likely to have a different set of cities. The question/answer phase of the exercise begins with the outline map displayed on screen (Figure 3.1). The student is queried for input on the location and attributes of each city in turn. The student's input is stored in a text file (whose format is described in Section 4.5.2) for further analysis.

Feedback on the student's performance is displayed as a warped map (the student's mental map shown by dotted line) contrasted with the original (shown by solid line) in Figures 3.1 and 3.2. Distance errors in locations of cities are shown by lines. Also, the student's answers for attribute values are shown along with the actual values. Finally, the student can view all of the attribute values and the location for a given place as part of a tutoring phase.

3.2 Feedback

Research has shown [Shne87] that users adapt to the system to work more quickly with shorter response time, and that they consistently prefer the faster pace . With shorter response time, user satisfaction is increased. Keeping this in mind, the prototype has been designed to provide an almost instant feedback to the student as a result of his/her action. As soon as the student enters the location of a city on a query, a small solid red circle appears on the screen at that location. Thus, the student knows instantly that the input has been acknowledged. Immediately after the end of the question/answer phase, the student's mental map is displayed. The algorithms (described in Section 3.3.1) are fast enough to display the rubber sheeted mental map within ten seconds after the end of the question/answer phase. More about the interface construction is described in Chapter 4.

1 In the prototype version, the user is allowed to select the number of cities and which of the available attributes are to be queried on prior to each question/response session.

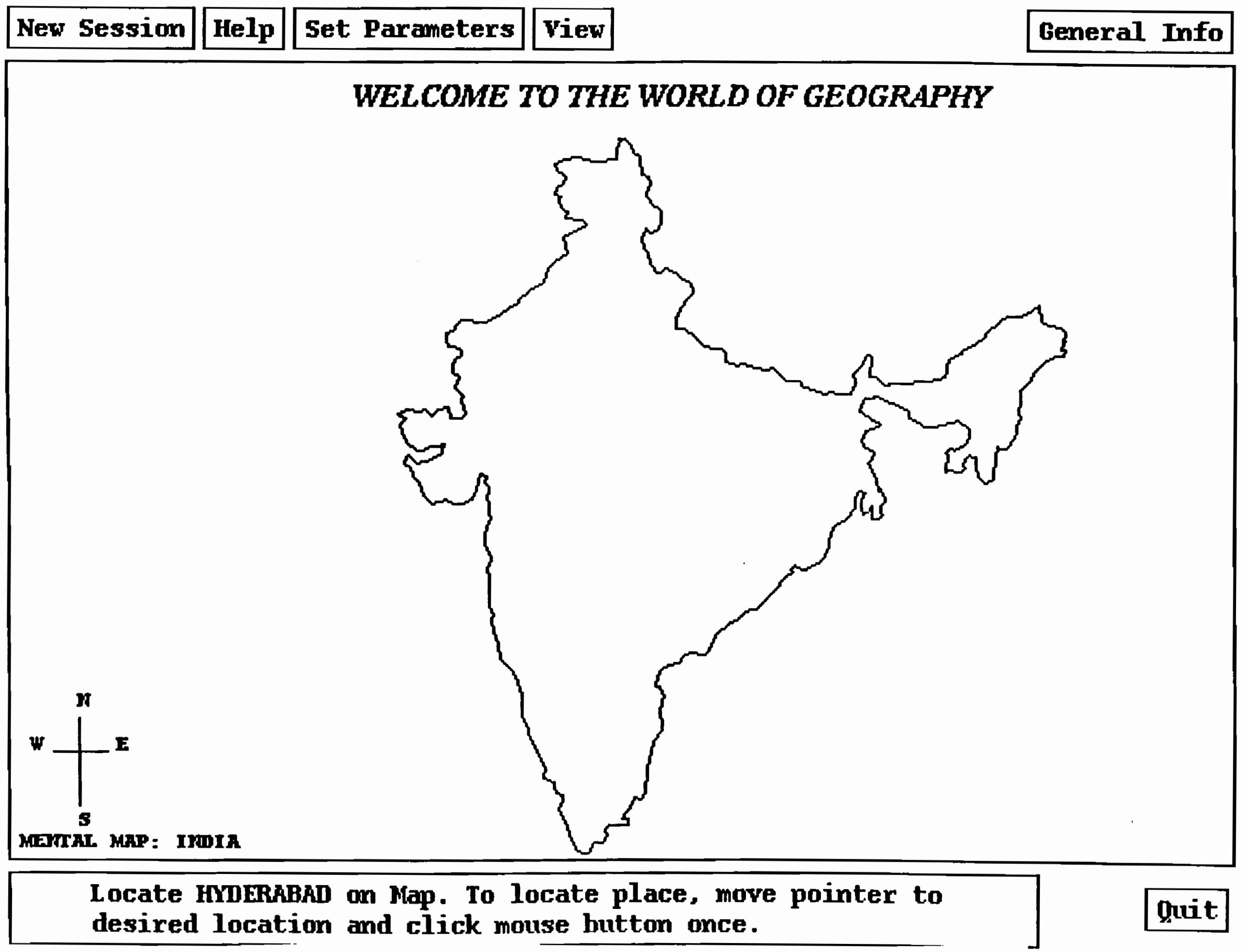

Figure 3.1: Prototype Interface for the Mental Mapping Module (Prior to Question/Answer Phase)

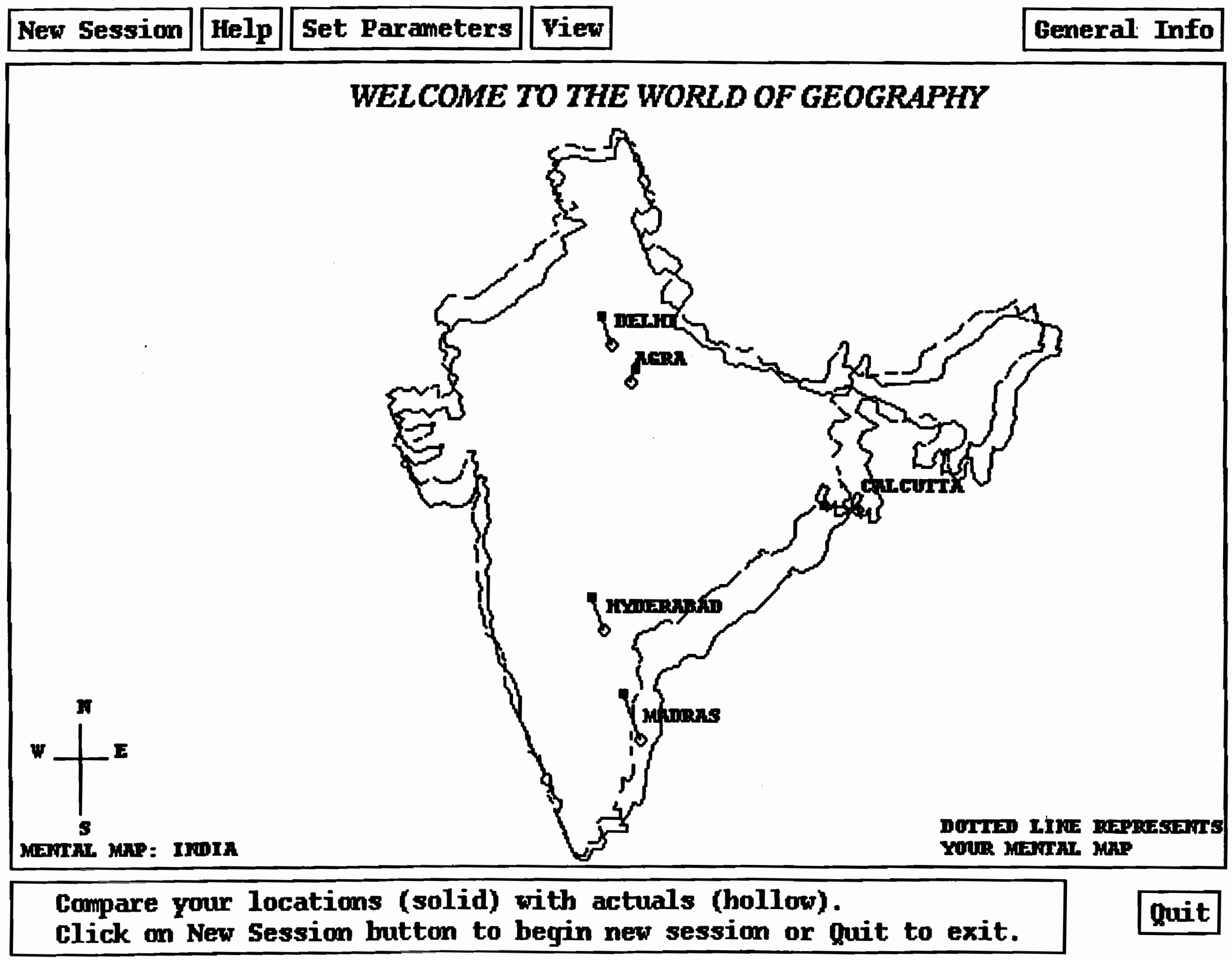

Figure 3.2: Prototype Interface for the Mental Mapping Module (After Question/Answer Phase)

The "mental map" is simply a rubber sheeted version of the original map. The rubber sheeting algorithm has to distort only those areas of the original map where the student has erred on the locations of cities. The other areas of the map should remain unaffected. To achieve this, the map should be divided into areas such that, associated with each area, there is at least one city. This will ensure that if there is an error in the location of some city, only the region of the map associated with that city will be distorted. In general, if the student locates all cities on the west coast correctly but does not do as well with the cities on the east coast, only the regions of the map associated with the east coast would be distorted. The next section describes in detail the rubber sheeting algorithms based on the above criteria.

3.3 Rubber Sheeting Algorithms

3.3.1 Piecewise Linear Transformations

The technique used for the rubber sheeting process is separate piecewise linear transformations applied to each of a set of triangles making up the window in which the map is displayed. The vertices for triangulation consist of the selected set of cities, augmented by the corners of the window containing the map outline (as shown in Figure 3.12). Each triangle is assigned a unique ID number. P_i represents the city number. The outermost rectangle is the window periphery. Since the outline map is drawn within the triangulated region, each point M_i on the outline map will lie within exactly one triangle. A point lying on the boundary line of two triangles is taken to be lying within the first of the the two triangles that encounters it. Every coordinate point along the map outline will be associated with a unique triangle ID number.

The student is asked to locate the randomly generated set of cities on the screen. Since there is a one to one correspondence between the actual locations of cities and the student's locations, a distorted version of the original triangulation can be obtained, with each triangle in the distorted version corresponding with precisely one triangle in the original

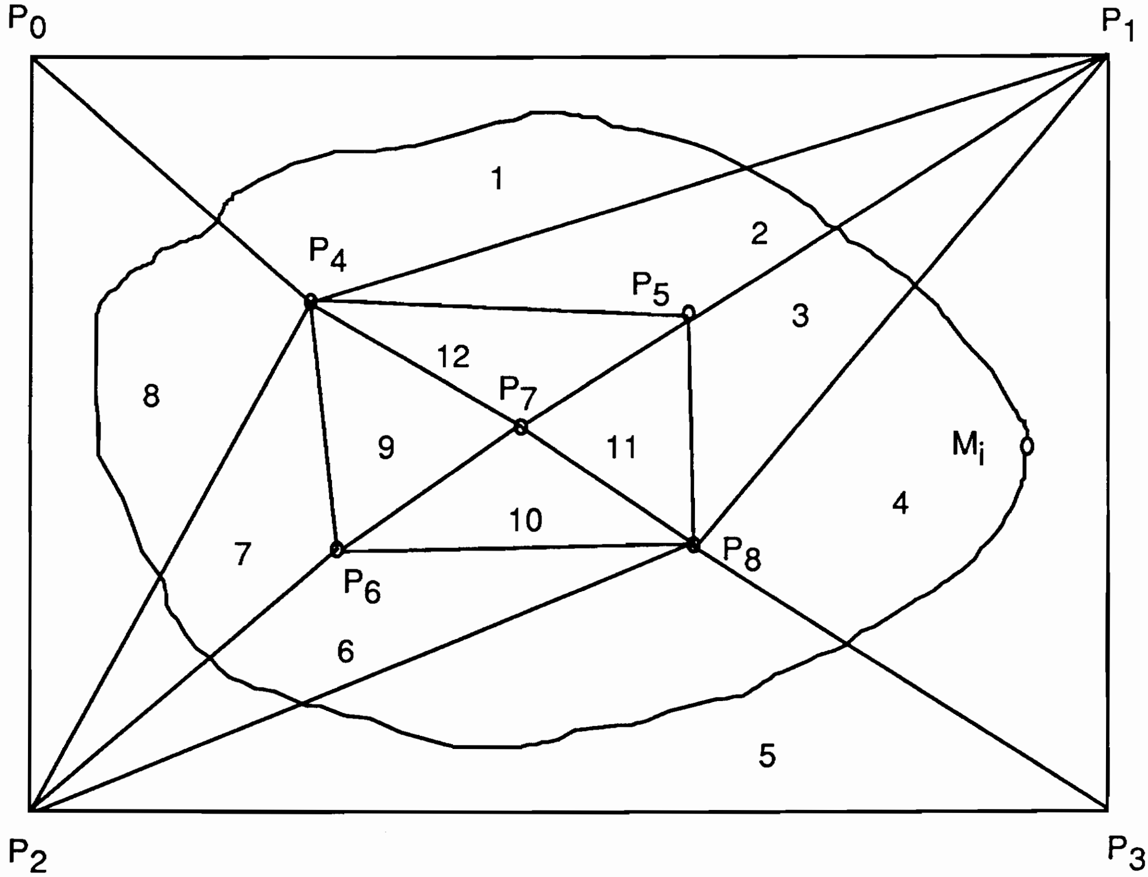

Figure 3.3: Greedy Triangulation on the Cities with Box (Enclosing Outline Map) as Convex Hull

triangulation. The goal of the rubber sheeting algorithm is to transform the original triangulation to exactly match the distorted one. Thus, each triangle belonging to the original triangulation should be transformed to exactly match the corresponding triangle as defined by the student's responses. The transformations applied to achieve this objective are simply a series of $2D$ linear transformations (described in Section 3.2.1) applied to each triangle. The result of these transformations is a 3×3 matrix for each triangle, that stores the cumulative homogenoeous linear transformations applied. Each point M_i (of the map outline) lying within this triangle can then be transformed by applying the transformation matrix to it. The series of transformed points of the map outline make up the rubber sheeted outline.

The initial triangulation is achieved using the greedy triangulation algorithm of Preparata and Shamos [Prep85].

Greedy Triangulation

The method of greedy triangulation is to select shortest available edges first as the edges for triangles. The algorithm is as follows:

Algorithm GREEDY_TRI /* Greedy triangulation on a given set of points */
INPUT: Set of points SP containing all the cities and the vertices of the convex hull.
The rectangular bounding box is the *convex hull* for any given set of points defining
the locations of the cities.
OUTPUT: The set L containing the edges for the triangulation

FOR each point of SP
    ASSIGN a unique ID P_i where $0 \leq i < (n + 4)$, n being the number of cities;
    /* $n + 4$ includes the 4 vertices of the window periphery */
FOR each point $P_i \in SP$
    BEGIN
        CONNECT P_i to every other point $P_j \in SP$ giving a set L;

/* $L = \{(P_i, P_j) | P_i, P_j \in SP, P_i \neq P_j\}$ */
/* L may contain intersecting and non-intersecting line segments.*/
END
FOR each line segment $(P_i, P_j) \in L$
COMPUTE the Euclidean distance between P_i and P_j;
SORT L in ascending order of lengths of (P_i, P_j);
FOR each line segment $(P_i, P_j) \in L$
BEGIN
IF (P_i, P_j) intersects with any other $(P_k, P_l) \in L$ THEN
DELETE all such (P_k, P_l) from L;
/* Line segments with common end points and line segments with point of
intersection lying beyond the end points are considered as non-intersecting. */
END
/* The set L now represents the edges for the triangulation since there are no intersections
and all the edges are connected. (P_i, P_j) and (P_j, P_i) are considered to be the same
edge (P_i, P_j), since the triangulation is an undirected graph. */

The above algorithm is simple and easy to implement. Since the number of cities is limited to 20, it is fast enough for this application.

Numbering Triangles

The set L represents a list of edges that form the triangles, but this set does not tell us which triples of edges (or vertices) form triangles. We divide the region into triangles with unique ID T_k, where $0 \leq k < t$, t = total number of triangles, as follows:

Algorithm NUM_TRI /* Assign a unique ID T_k to each triangle */
INPUT: Set of points SP and set of edges L
OUTPUT: The set TR containing triangles T_k along with its triple of vertices, and t

```
FOR each point P_i ∈ SP
    BEGIN
        LIST the edges (P_i, P_j) incident on the point P_i;
        FIND the positive angle made by each incident edge with the horizontal X axis;
        /* Counterclockwise direction is taken as positive. */
        SORT these edges in the ascending order of angles;
        /* Two such adjacent edges should form two edges of a triangle.
        These two edges contain all three vertices of the triangle since P_i is
        the common point and the other two end points form the third edge. */
        IF this triangle does not already exist in the set TR THEN
            ADD it to the set TR after assigning it a unique ID number T_k;
    END
```

Point Membership

Any linear transformation(s) applied to a triangle T_k, will have to be applied to all the points that lie within that triangle. Each point M_i on the outline map will fall within a unique triangle T_k since the entire region is divided into a set of distinct triangles. Thus, the next step is to determine which points on the outline map belong to each triangle T_k.

To detect whether a given point lies within a triangle or not, a simple point in polygon algorithm is used. Given a point M_i and a triangle T_k, a ray parallel to the X axis is extended from M_i in the positive direction of the X axis. If the ray intersects only one edge of T_k, then M_i lies within T_k, else M_i lies outside T_k. If M_i lies on any edge or vertex of T_k, it is considered as lying within T_k, if T_k is the first triangle that encounters it.

An inefficient approach would be to take each point on the outline map and scan the entire set TR of triangles to find which triangle M_i belongs to. We can speed up the search process since the euclidean distance between given two adjacent points on the outline map will rarely be greater than the width of a triangle since the outline map has consecutive points close to each other to capture the contours of the map. Thus, we check for triangle membership for points on the outline map in a particular order, by moving around the outline. This way, there is no need to scan the entire set TR, but rather simply check to see if the next point is in the current triangle. If not, we check the neighboring triangle in the appropriate direction.

Since we need to quickly find neighboring triangles, we explicitly store along with each triangle its list of neighbors. This is accomplished by taking each triangle T_k of the set TR and checking to see which other triangles from TR have one or more vertices in common with T_k; these will be the neighbors of T_k. We calculate and store these neighbor lists as part of the triangulation process.

The algorithm for assigning map outline points to triangles is as follows:

Algorithm POINT_MEMBER /* Assign a unique T_k to each point M_i */

INPUT: Set of points M and set of triangles TR

OUTPUT: Each $M_i \in M$ associated with a unique id T_k

```
Let M_1 be the first point from set M;
FOR each triangle T_k ∈ TR
    IF M_1 lies within T_k THEN /* using point in polygon algorithm */
        BEGIN
            current triangle = T_k;
            ASSOCIATE M_1 with T_k;
        ENDIF
FOR each point M_i ∈ M
    BEGIN
        IF M_i lies within T_k THEN;
            ASSOCIATE M_i with T_k;
        ELSE
            SELECT the triangle neighboring T_k on the edge through which
```

```
            the line (M_i,M_{i-1}) passes;
            Make this the current triangle T_k;
            ASSOCIATE M_i with T_k;
        ENDIF
    END
```

The greedy triangulation algorithm along with the point membership algorithm take about 10 seconds on a Macintosh II running X Windows to run for a set of 20 cities and a map outline containing about 400 points. This is a satisfactory response rate for the student.

Student Input

The student is asked to locate each city on the map by positioning the cursor on screen to the desired location and pressing the mouse button at that location. Since each city is a vertex of a set of triangles from TR, and since each city has been assigned a unique ID P_i, we can associate the student's location for each city Q_i with the corresponding actual location for the city P_i. Also, we can associate each of Q_i with the vertices of the corresponding triangles T_k. If we join all Q_i with the same rules as those of set L (from the greedy triangulation described earlier in this section), we get a corresponding new triangulation, with a corresponding new set of triangles STR. There will be a one to one correspondence between the triangles of TR and the triangles of STR.

Map Warping

Each triangle T_k from the set TR of original triangles is transformed to be an exact match of the corresponding counterpart from the set STR of student's triangles ST_k. The overall transformation is a series of the affine transformations as described below. We use the traditional method of multiplying together a series of transformation matrices making up

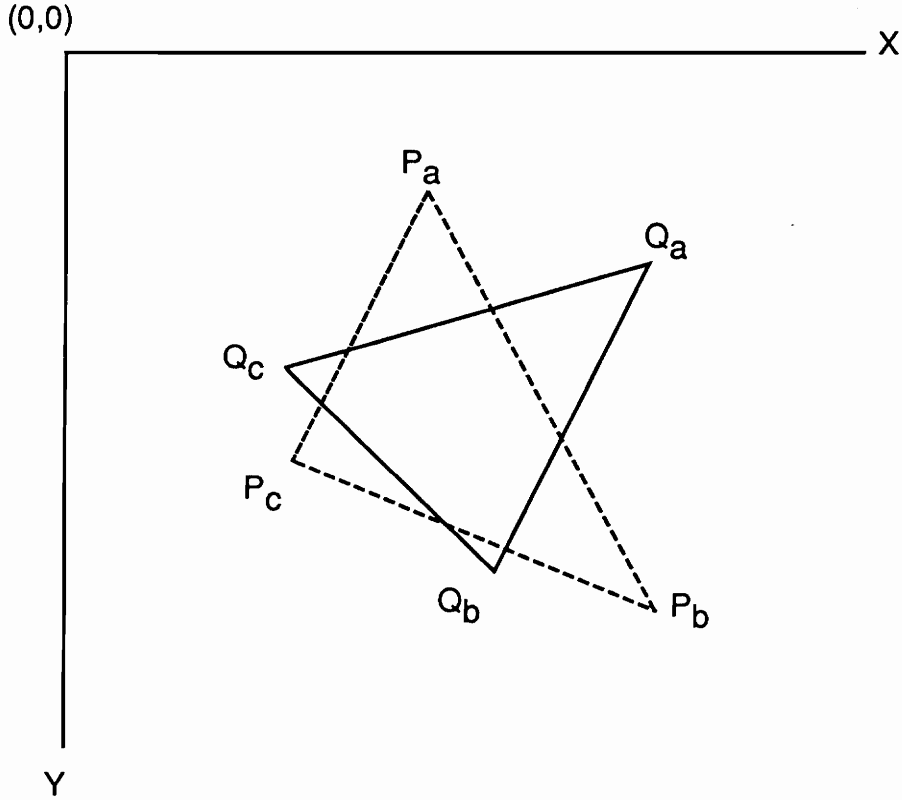

Figure 3.4: A Typical Pair of Original Triangle (Solid Line) and Corresponding Student Defined Triangle (Dotted Line)

the individual steps of the transformation [Fole90, Hear86]. The advantage of this approach is that only a single matrix multiplication needs to be done for each point on the outline map (there are likely to be more points on the outline map than triangles).

Consider Figure 3.4. Vertices P_a, P_b, P_c represent the vertices of the original triangle T_k. Vertices Q_a, Q_b, Q_c represent the vertices of the new triangle ST_k.

Translation:

One vertex Q_a of ST_k is chosen as the reference vertex. The corresponding vertex P_a of T_k is translated to coincide with this vertex of ST_k. The translation coefficients are:

$$tx = Q_{ax} - P_{ax} \qquad ty = Q_{ay} - P_{ay}.$$

The translation coefficients are stored in a homogeneous transformation matrix TM_k associated with each triangle T_k.

$$TM_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ tx & ty & 1 \end{bmatrix}$$

TM_k is applied to vertices P_a, P_b and P_c.

Figure 3.5 represents the effect of the above translation.

Rotation:

Next, one of the edges (P_a, P_b) of T_k incident on P_a, is rotated by angle θ to coincide with the corresponding edge (Q_a, Q_b) of ST_k (Figure 3.5). The point of rotation (pivot) is taken to be Q_a. Again the rotation coefficients are stored in a temporary 3×3 transformation matrix M.

$$M = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ rx & ry & 1 \end{bmatrix}$$

where,

$$rx = (1 - \cos\theta)Q_{ax} + Q_{ay} \times \sin\theta$$

$$ry = (1 - \cos\theta)Q_{ay} - Q_{ax} \times \sin\theta$$

θ is the difference in the angles made by edge (P_a, P_b) and edge (Q_a, Q_b) with the horizontal X axis (Figure 3.5).

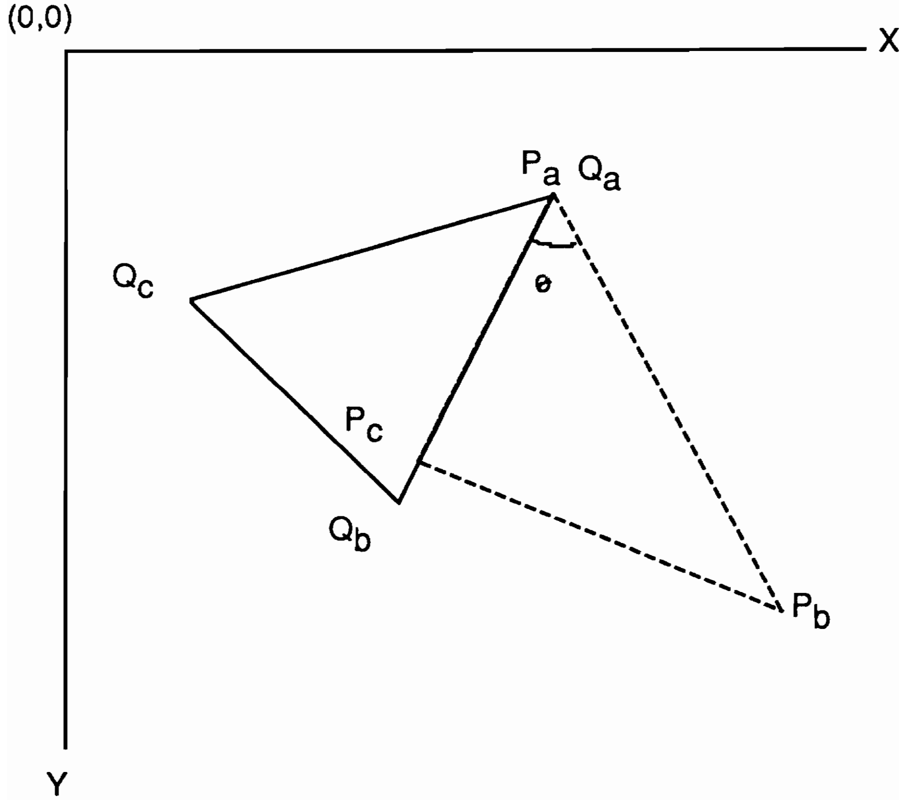

Figure 3.5: Effect of Translation

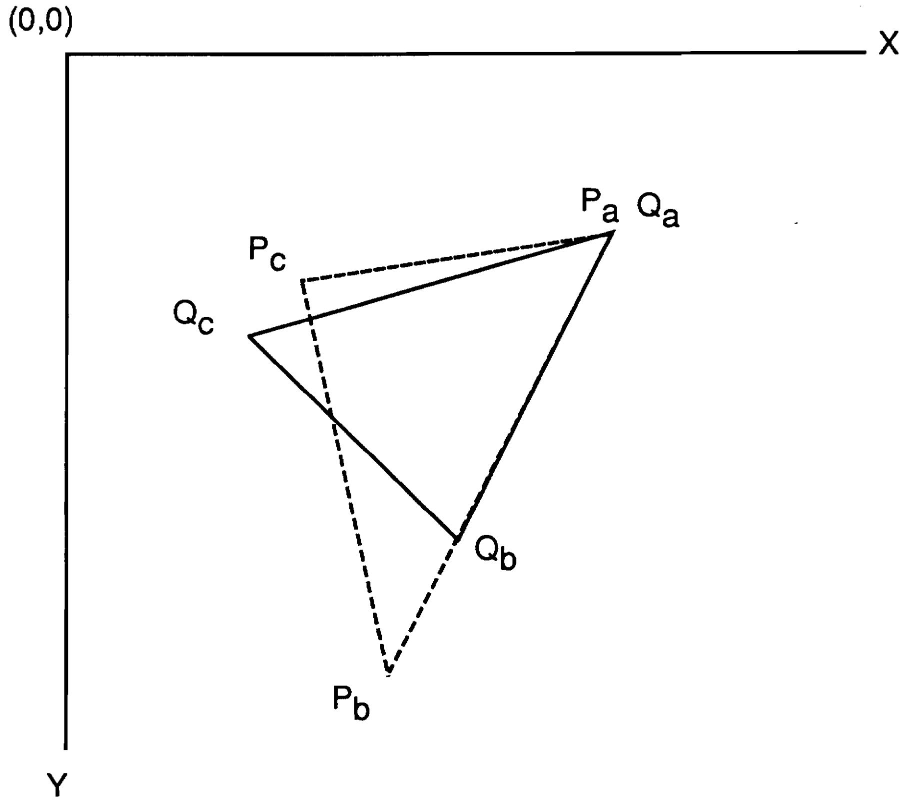

Figure 3.6: Effect of Rotation

Matrix M is applied to P_b and P_c to give the new position. Matrix M is then multiplied by TM_k to give an updated cumulative transformation matrix TM_k. The cooordinates of every point of the outline map lying within triangle T_k are multiplied by TM_k to give the transformed coordinates.

The constants needed for TM_k at this stage are,

$$trx = tx \times \cos\theta - ty \times \sin\theta + rx$$

$$try = tx \times \sin\theta - ty \times \cos\theta + ry$$

Figure 3.6 shows the effect of this rotation.

Scaling:

The edge (P_a, P_b) of T_k is now scaled up or down, as the case may be, to completely coincide with the corresponding edge (Q_a, Q_b) from ST_k. The scaling is a fixed point scaling (where the object is scaled with respect to a fixed point) with Q_a as the reference point. Triangles T_k and ST_k are both translated to the origin, and scale factors $sx = Q_{bx}/P_{bx}$ and $sy = Q_{by}/P_{by}$ computed, and then T_k and ST_k are retranslated back to the original position. The scaling coefficients are stored in a temporary matrix and this temporary matrix M is multiplied by TM_k to give an updated overall transformation matrix TM_k.

$$M = \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ (1-sx)Q_{ax} & (1-sy)Q_{ay} & 1 \end{bmatrix}$$

M is applied to P_b and P_c. The constants needed for TM_k at this stage are,

$$tsx = trx \times sx + (1 - sx)Q_{ax}$$

$$tsy = try \times sy + (1 - sy)Q_{ay}$$

Figure 3.7 shows the effect of this transformation.

Stretching:

Now, we have P_a coinciding with Q_a and P_b coinciding with Q_b. We need to make P_c coincide with Q_c and then we are done with the transformation. First, we translate the points P_a and Q_a to the origin and then rotate edges (Q_a, Q_b) and (P_a, P_b) by angle θ made by them with the X axis such that they coincide with the X axis.

The transformation matrix M for this set of transformations is as follows:

$$M = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ (-Q_{ax} \times \cos\theta + Q_{ay} \times \sin\theta) & (-Q_{ax} \times \sin\theta - Q_{ay} \times \cos\theta) & 1 \end{bmatrix}$$

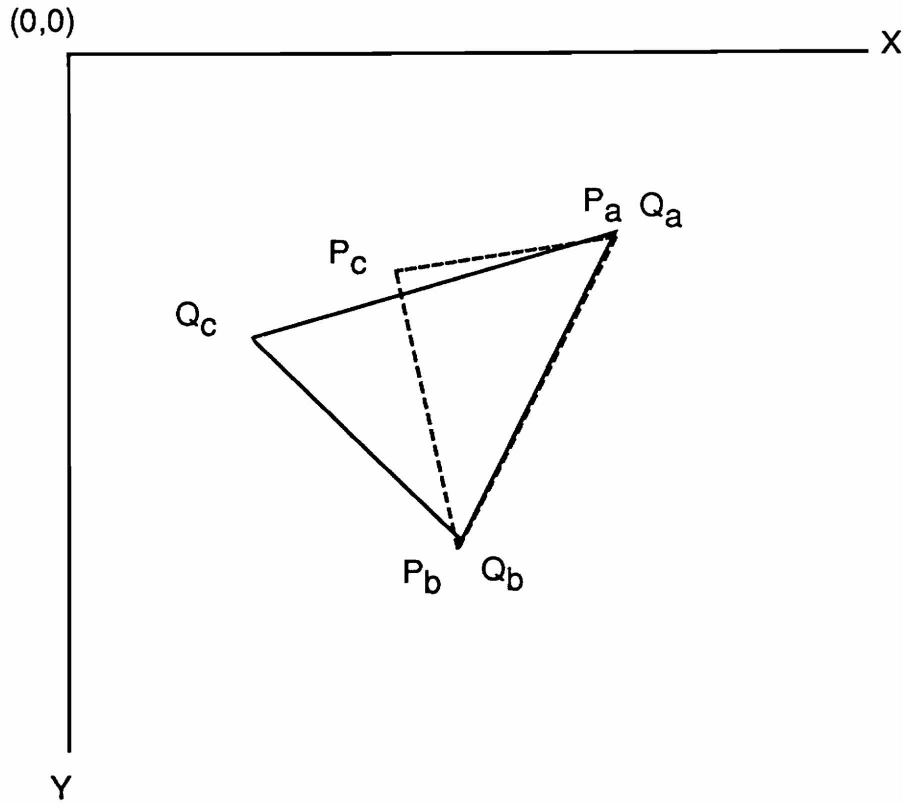

Figure 3.7: Effect of Scaling

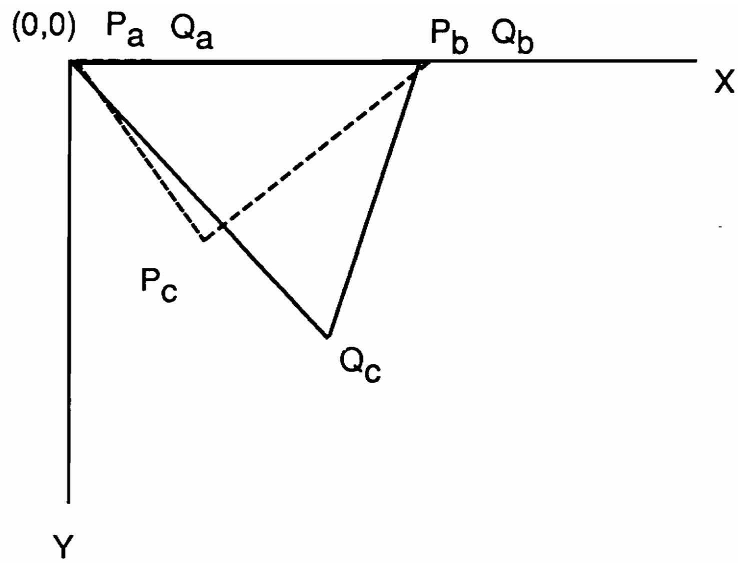

Figure 3.8: Effect of Translation and Rotation

This transformation matrix M is applied to point P_c. The constants needed for TM_k at this stage are,

$$tcx = tsx \times \cos\theta - tsy \times \sin\theta - Q_{ax} \times \cos\theta + Q_{ay} \times \sin\theta$$

$$tcy = tsy \times \sin\theta - tsy \times \cos\theta - Q_{ax} \times \sin\theta - Q_{ay} \times \cos\theta$$

Figure 3.8 shows the effect of this transformation.

Now point P_c is stretched in the vertical direction such that the y coordinate of P_c is the same as the y coordinate of Q_c. The x direction has a scale factor of 1, but the y direction has a scale factor $ys = Q_{ay}/P_{ay}$.

The transformation matrix M for this is,

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & ys & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This new M is applied to P_c.

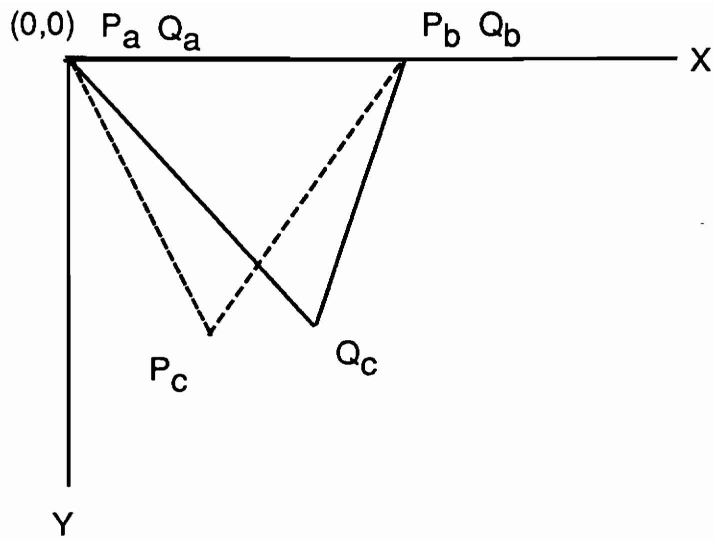

Figure 3.9: Effect of Stretching

Figure 3.9 shows the effect of this transformation.

Shearing:

Now, vertex P_c of T_k is sheared in the X direction to coincide with the corresponding vertex Q_c of ST_k. The shear factor shx is $(Q_{cx} - P_{cx})/P_{cy}$. The transformation matrix M is

$$M = \begin{bmatrix} 1 & 0 & 0 \\ shx & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

M is applied to P_c.

Figure 3.10 shows the effect of this transformation. The constants needed for TM_k at this stage are,

$$thx = \cos^2\theta \times sx - \sin^2\theta \times sy + shx(sx + sy)\sin\theta \times \cos\theta \times ys$$

$$thy = (sx + sy)\sin\theta \times \cos\theta \times ys$$

$$trx = \sin\theta \times \cos\theta(-sx - sy) + shx(\cos^2\theta \times sy - \sin^2\theta \times sx)ys$$

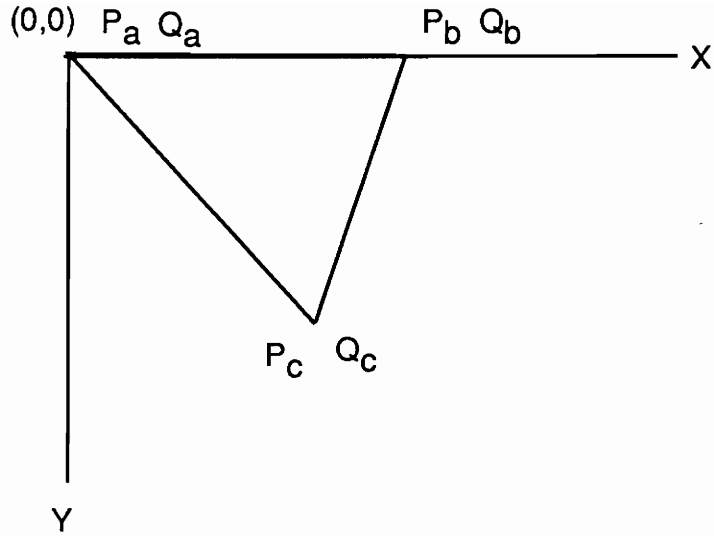

Figure 3.10: Effect of Shearing

$$try = (\cos^2\theta \times sy - \sin^2\theta \times sx)ys$$

Next, we have to restore the triangles back to their original locations. Thus we first unrotate the edges (P_a, P_b) and (Q_a, Q_b) from the X axis and then retranslate the triangles back to the original Q_a (the reference point).

The transformation matrix M is,

$$M = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ Q_{ax} & Q_{ay} & 1 \end{bmatrix}$$

M is applied to P_c.

After multiplying all of the pieces together, we come up with the final transformation matrix TM_k as follows:

$$TM_k = \begin{bmatrix} thx \times \cos\theta + thy \times \sin\theta & -thx \times \sin\theta + thy \times \cos\theta & 0 \\ trx \times \cos\theta + try \times \sin\theta & -trx \times \sin\theta + try \times \cos\theta & 0 \\ Q_{ax}(tcx + shx \times tcy \times ys) & Q_{ay} \times tcy \times ys & 1 \end{bmatrix}$$

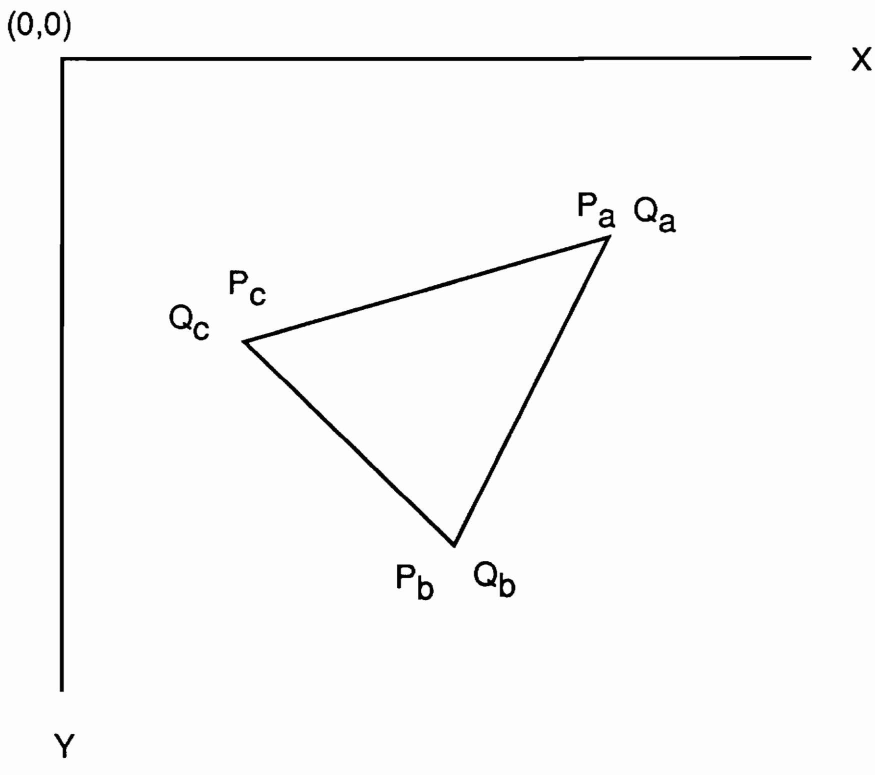

Figure 3.11: Final Transformation

Figure 3.11 shows the final transformation.

We now have T_k exactly matching the corresponding ST_k and the transformation matrix TM_k associated with the transformations. Every point M_i of the outline map lying within triangle T_k is multiplied by TM_k. After all the points M_i on the outline map have been multiplied by their respective TM_k, they form the coordinates of the warped outline map.

The main advantage of piecewise linear transformations is that the errors are localized to the region in which they occur. For example, typical U.S. students from the East Coast get every city on the East Coast within tolerance, but not those cities on the West Coast or in the Mid-West. The warpage obtained by this algorithm shows distortions only on the western outline of the map and not so much on the eastern outline. This correctly reflects the student's "mental map".

3.3.2 Limitations of Piecewise Linear Transformations

Results show that even though greedy triangulation gives small, almost equilateral type of triangles, it cannot guarantee this for all cases. Sometimes, there is a possibility of one or more thin scalene triangle(s). This type of triangle(s) can lead to a change in topology from the original triangles, and thus cause flip-overs (Figure 3.12) in the outline map, which is aesthetically undesirable.

The change in topology is due to the fact that one of the vertices of the original triangle is so close to the opposite edge of that triangle that a small distance error in the location of the city associated with that edge can cause the vertex to be on the other side of the edge in the new triangulation. Also, if the student has grossly erred in the location of a city, no matter how good the triangulation is, chances are that this kind of distance error can distort the original topology causing flip-overs in the outline map. In practice, this does not occur very often unless the subject student has absolutely no idea about the geography of the country or region on which the quiz is based, and thus, ends up locating the cities at arbitrary positions on the map.

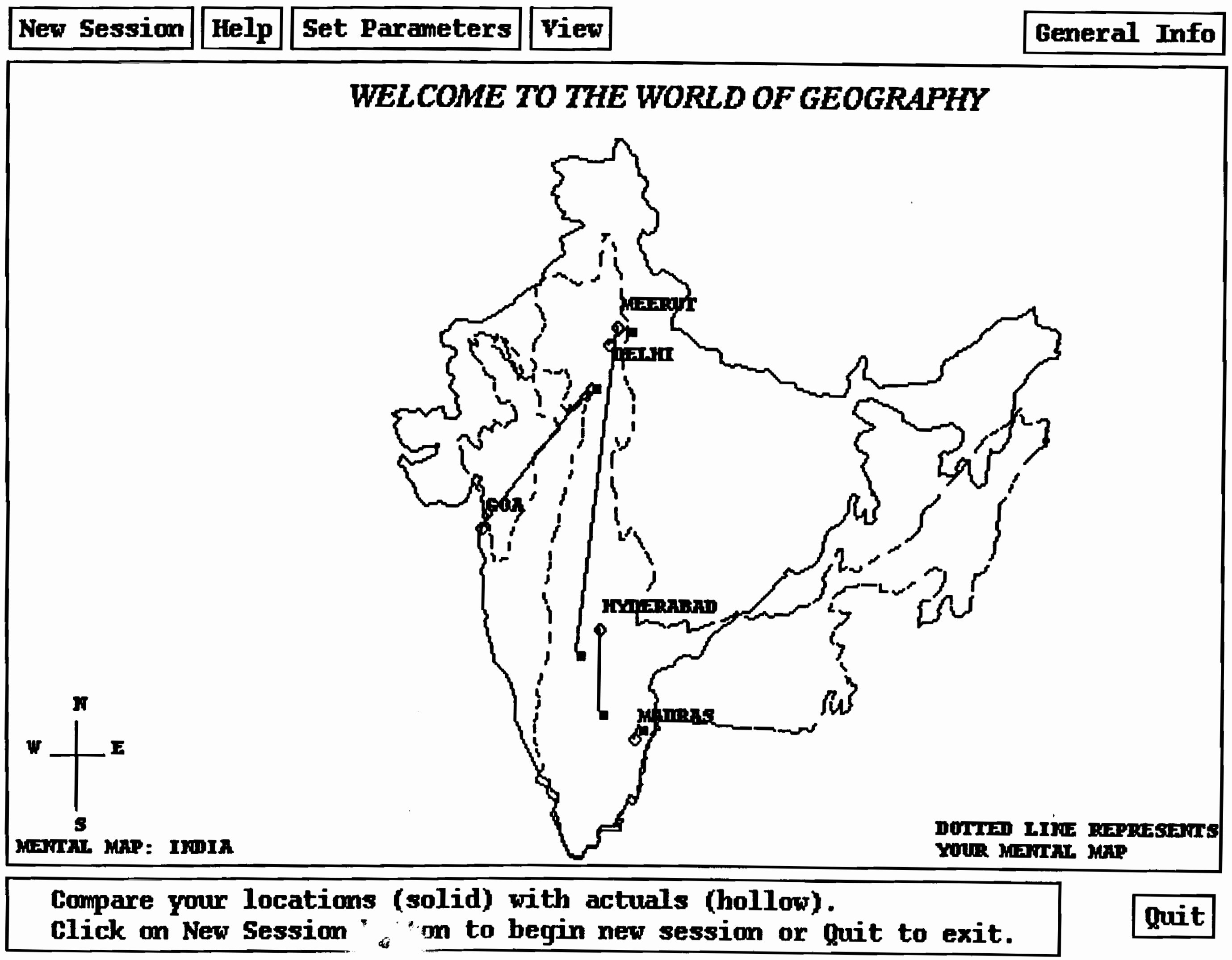

Figure 3.12: Flip-Over in Mental Map

3.3.3 Method of Least Squares - Best Fit

To overcome the limitations of the above method, another method for rubber-sheeting the outline map was explored. This method involves the following steps:

1. The original set of points (cities) SP is stored in a suitable data structure. Each point has a unique ID P_i, $0 \leq i < n$, n = total number of cities.

2. The student's set of points SQ is recorded. There is one to one correspondence between the original set of points and the student's set of points, i.e., associated with each $P_i \in SP$, there is one and only one $Q_i \in SQ$.

3. The student's set of points form set PO.

4. PO is linearly transformed so that the sum of the squares of the euclidean distances between each pair (P_i, Q_i) is minimum. A convergence test has to be carried out to determine the minimum.

5. These transformations are stored in a 3×3 matrix TM which has the coefficients for the overall transformation. There will be only one transformation matrix TM for the entire outline.

6. Every point M_i on the outline map is multiplied by TM to get the warped outline.

3.3.4 Limitations of Method of Least Squares

Distance errors local to a certain region of the map can have a global effect on the outline map since there is just one transformation matrix TM for the entire map. Even though the student may perform correctly on all but one city, this distance error will be propogated throughout the outline. This may not be a good feedback to the student since the distortion is not localized to the area in which it occurs.

3.4 Choice of Rubber Sheeting Algorithm

The method of piecewise linear transformations is preferred over the method of least squares because it represents the student's correct mental map. It takes under 10 seconds to run for a set of 20 cities, which is a satisfactory response time. The method of least squares can take longer for the same number of cities due to the convergence test involved. Moreover, the warped map outline obtained as a result of this algorithm is not the true mental map because location errors are propagated throughout the outline instead of localizing them to only those regions in which they occur. From the point of view of implementation, both require about the same amount of programming effort. However, the method of piecewise linear transformations uses traditional computer graphics algorithms and thus simplifies most of the programming effort.

The piecewise linear transformation algorithm is a combination of greedy triangulation algorithm and homogeneous 2-D linear transformations. This combination is specific to our application, but the individual algorithms are well known in computational geometry and computer graphics. The method of least squares algorithm is a combination of homogeneous 2-D linear transformations and linear programming algorithms. This algorithm is used extensively in Civil Engineering applications.

Chapter 4

INTERFACE DESIGN

4.1 Design Criteria for the Mental Mapping Module Prototype

The design of tutoring modules for geography presents some significant challenges to the developers of computer-aided learning tools. The key problems identified in this prototype were the human-computer interface (HCI) for naive computer users, the need for interactive graphics, and overall flexibility in the design structure to cater to future modifications.

Some of the important requirements of a good geography tutor have been identified to be the following [Magu89]:

- The tutor should be robust.
- It should be capable of use by novices with little or no computer experience.
- It should require minimum student input to make it less cumbersome to use.
- It should run quickly and be entertaining in order to capture the student's attention.
- It should illustrate something of both the principles and applications of geography.

Recent research has demonstrated the value of using highly interactive systems in the learning process [Rape89]. Systems using window managers, icons, mice, and pop-up menus (WIMP) have become popular following their development in the Smalltalk-80 project at XeroxPARC, and their successful implementation on the Apple Macintosh. These techniques now represent a new methodology in HCI for micro-computers and workstations. Recent research by the Gartner group [Rape89] showed how computer users were able to learn more quickly and become fully familiar with more applications when using a consistent WIMP interface, rather than a standard command line system. This can be traced to the use of the visual cues on the display window in the form of messages, the ability to select options from menus, and the execution of commands using the mouse. The learning overheads on the students can be reduced by making the system easy to use: this leaves more time and attention to be devoted to the subject matter.

Another key problem identified concerned the predominance of graphical subject matter required to teach geography concepts [Rape89]. To illustrate these concepts requires sophisticated graphical capabilities to display maps and the operations carried out on them. However, to exploit fully the advantages of the graphical display, the student should also be able to interact with the maps (for example, locating cities). Accordingly, a WIMP interface was seen as necessary to simulate real graphical operations and to allow question/answer sessions.

The GeoSim system aims to work with several WIMP systems, including Microsoft Windows, the Macintosh Toolbox, and the X Window System. We are currently in the "proof-of-concept" phase of our project, and since (i) the International Population module is being implemented using the Macintosh Toolbox, and (ii) the X Windows environment was the one best known by the present author, the X Window System was selected as the prototype implementation environment.

The X Window System offers a convenient opportunity to address these interface problems since it supplies a smooth and flexible user interface at a time when users are growing increasingly accustomed to window-style interfaces. It provides a complete graphics system

— complete enough to create windowing interfaces *à la* Macintosh and complete enough to handle graphics-intensive applications. The X Window System shows great promise toward solving a seemingly intractable problem: how to provide a common interface accross many different computers that are running a number of operating systems with a number of different displays [John89]. It provides a graphical interface that runs on every platform from IBM PCs to large mainframes and supercomputers.

The X Window System, by being operating system independent, encourages the portability of software. The standard X Window C library routines called Xlib [Nye90], are the same on every machine running X, which means the graphics code ports directly from one machine to another because all the X calls are the same. Also, the Athena widget set provides a standard set of C library routines called Xaw [Nye90] for the interface primitives. Since the user interface typically takes up 30-60 % of the code in this kind of application, the applications become more portable.

X takes user input from a pointer. The pointer is usually a mouse. The pointer allows the user to control a program without the keyboard, by pointing at objects displayed on screen such as menus and command buttons. This method of using applications is often easier to learn than the traditional keyboard control.

4.2 User Profile

The most enigmatic problem in designing any interface is deciding what kind of user would be using the application. Depending on the application, a wide spectrum of users could be using it, making it difficult to design a most general interface. It therefore becomes necessary to assume a certain minimum background (in terms of familiarity with computers, etc.) that a user should possess before the interface can be designed. Since the Mental Mapping module is targeted towards introductory geography students, the end user is assumed to be at least a senior high school student. Context-sensitive operating instructions have been provided to guide the student through every phase of the learning session. About 10 out of 15 students who tried using the module found them to be very helpful.

4.3 Construction and Layout of the Mental Mapping Module Prototype

The objectives the interface attempts to achieve are:

1. High level of interaction with the student with quick feedback, thus ensuring student's interest throughout the session.

2. No keyboard input thus eliminating typing of text. Being consistently mouse-driven, this module is less cumbersome to use than a combination of keyboard and mouse driven interfaces.

3. Context-sensitive operating instructions, to guide the student throughout the session.

4. Adequate help facility to make the student familiar with the operation of the module. Most of the students who used this prototype were able to guide themselves with the available help.

5. Support for color display to grab the students attention.

6. A portable software unit compatible with any UNIX/X Windows based environment.

The components of the interface are as follows:

1. A 570 pixels by 365 pixels graphics window, in the center of the display, which displays the outline map to the student and accepts mouse button input. (Figure 4.1).

2. At the bottom of the graphics window, a text area which dynamically presents the operating instructions to the student. For example, "Locate Cancun on the Map".

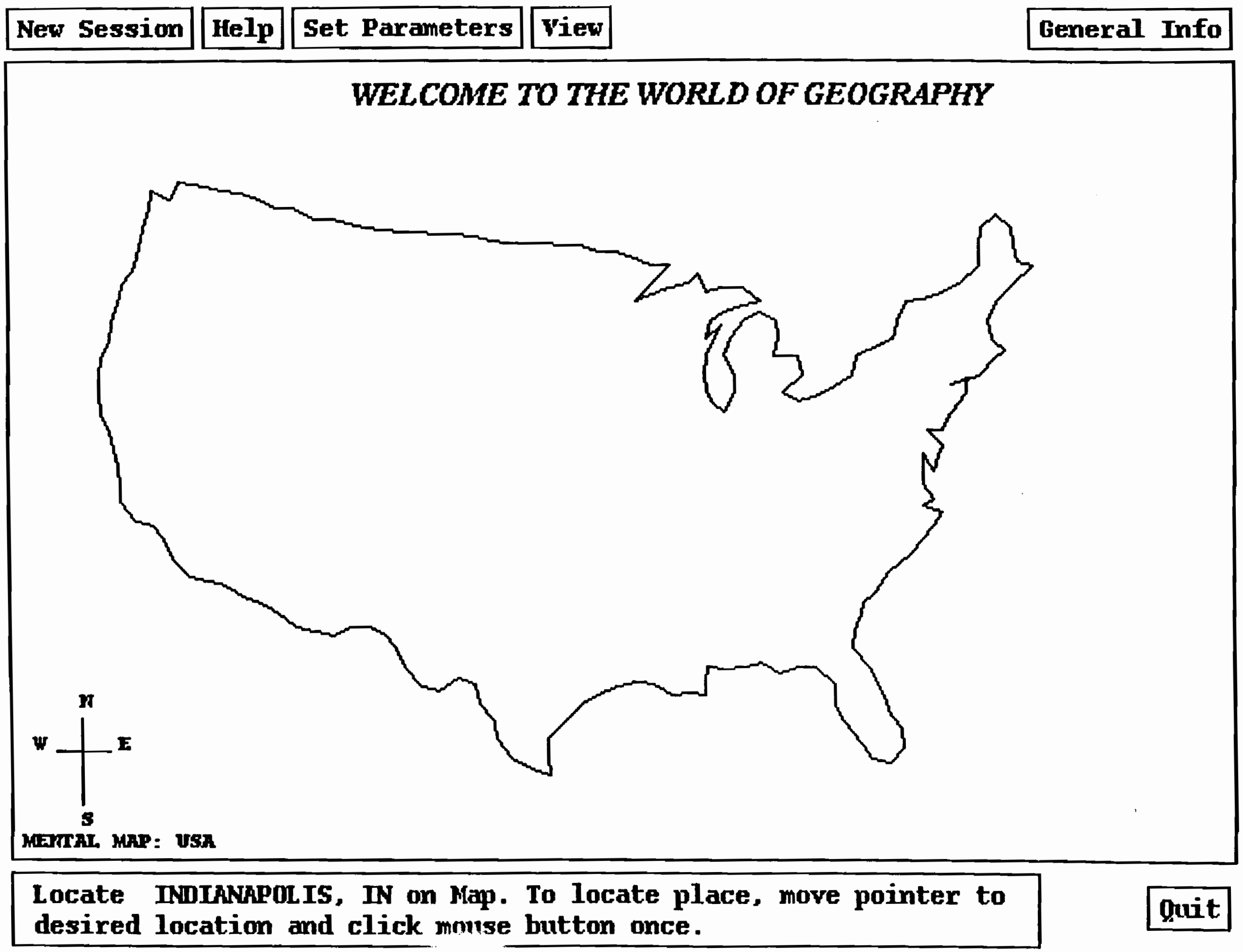

Figure 4.1: The Basic Interface

3. At the top of the graphics window, the menu/button header which contains pull down menus and buttons for various functions.

The "General Info" menu is a pull down menu that contains several options. The items on this menu are: i) The "About Mental Map" pops up a window on screen which displays a scrollable ASCII text file containing the name of the module, the version number, the authors and the date of creation; ii) The "About GeoSim" displays a scrollable ASCII text file which is shared by all modules and contains information about project GeoSim; iii) The "Technical Info" displays a scrollable ASCII text file which contains technical information about the module viz. algorithms used, database formats, etc.

The "Parameters" pulldown menu allows the instructor to set the input parameters for a given session. The "Cities" option under this menu allows the instructor to set the number of cities to be queried on for the given session. A pop-up window appears on screen with the options 5, 10, 15 or 20, which can be selected by a mouse click. The "Population" allows the instructor to turn the query about city population on or off. The "Climate" allows the instructor to turn the query about city climate on or off. The "Housing" allows the instructor to turn the query about city cost of housing on or off. The "Help" contains information on how to set these parameters.

The "View" pulldown menu allows the student to view population, climate and cost of housing statistics along with the student's values. The "Place" option under this menu allows the student to view all the statistics viz. population, climate and cost of housing for a given place. A list of places appears on screen and the student selects the place of his choice. A "Help" button is provided that will display a scrollable ASCII text file containing detailed help on how to get started on the module, its functionality, operating instructions, etc. (Figure 4.2).

The "New Session" button allows the student to abort the current session and/or start a new one.

The "Quit" button allows the student to quit the module.

4.3.1 Student's Interaction with the Module

The help area comment prompts the student to locate a city on the map by clicking the mouse button at some location on the graphics window (Figure 4.1). A small circle appears on the graphics window at that location to acknowledge the input. If additional questions have been selected, a horizontal slider bar appears on screen, prompting the student to set the values for population, climate and/or cost of housing in that city. The slider bar contains the range within which the actual values lie and has to be moved by "grabbing" with the cursor to set the values accordingly. (Figure 4.3).

As the student moves the position of the slider bar with the mouse, the values change accordingly. These are displayed to the student simultaneously as an acknowledgement of the action. After the student is satisfied with the value, he or she must click on the OK button to indicate that the value now can be accepted. After the preselected number of cities have been queried on by the student, the module provides feedback to the student in the following manner.

1. Original outline map is displayed. Distance errors in the locations of cites are displayed as straight lines between the actual locations and the student's corresponding locations. (Figure 4.4).

2. A warped outline map is also displayed on the same screen to show the student's mental map of the country.

3. New screens for each question (such as population size, climate and cost of housing) are

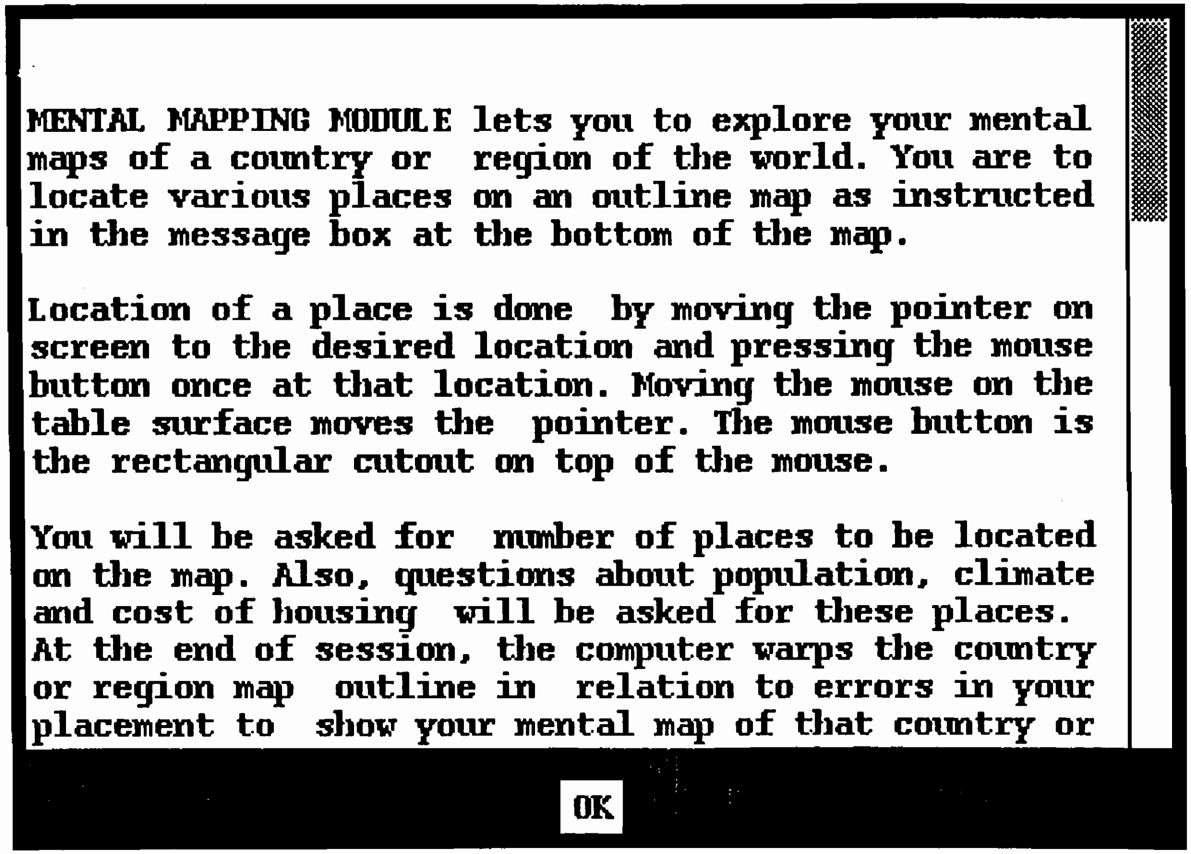

Figure 4.2: The Help Window with Scrolling Facility

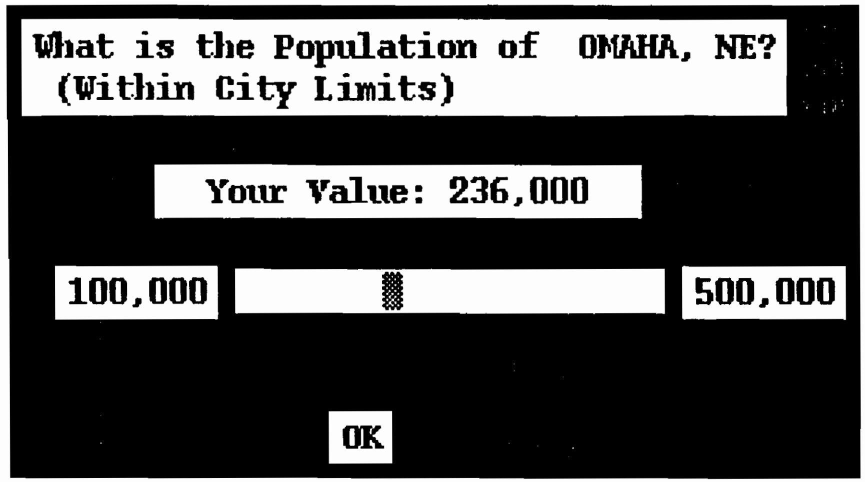

Figure 4.3: Scrollbar to Set the Student's Responses

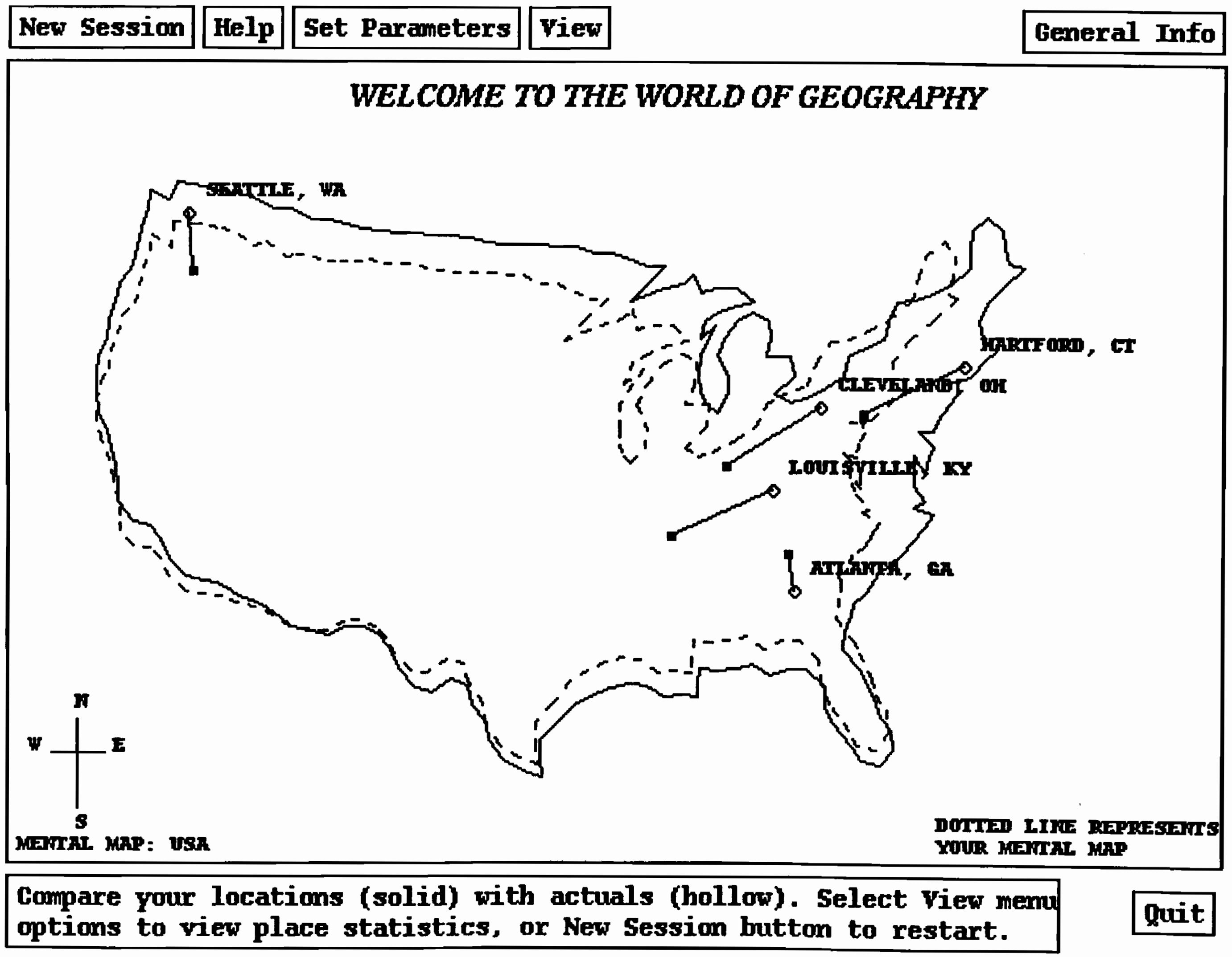

Figure 4.4: Display of Locational Errors and Mental Map

displayed along with the original outline map. The actual values along with student's values are displayed at the actual locations of the cities on the map. (Figure 4.5).

4. If a student desires to view the statistics for a place of his/her choice, she can do so by choosing the "Place" option under the "View" menu. (Figure 4.6).

The online context-sensitive help, coupled with a no-keyboard, totally-mouse-driven interface should make this module easy to use for a novice with little or no learning time.

4.4 Integration

The Mental Mapping module is part of project GeoSim, which is a comprehensive multidisciplinary effort involving members of the Department of Geography and Computer Science at Virginia Tech to develop computer-aided instruction(CAI) software for the teaching of introductory geography. During its first stage, the GeoSim project has focused on providing a series of GIS-based computer modules to be used as supporting material in college level introductory geography courses. Mental Mapping is only one of these modules and its design and implementation must reflect the fact that it is part of a larger whole.

A wide variety of personal computers and workstations will eventually be able to run these modules since implementation will be provided for several popular interface systems. Currently, the International Population module prototype is being developed in the Macintosh Toolbox environment. The Reduction of Commuting Time module prototype is being developed in the X Window environment. The interfaces for all modules will eventually have a common look and feel.

All modules will share the same menu-driven interface, and will obey the following format. First, a multi-media presentation of background material will be presented. In some cases (such as Mental Mappping), it may simply provide operating instructions, in other cases, its primary purpose is to set the stage for what is to come. Next, the primary interactive exercise will allow the student to explore the material at his or her own pace. The student may also repeat the exercise as many times as he or she desires. During this phase,

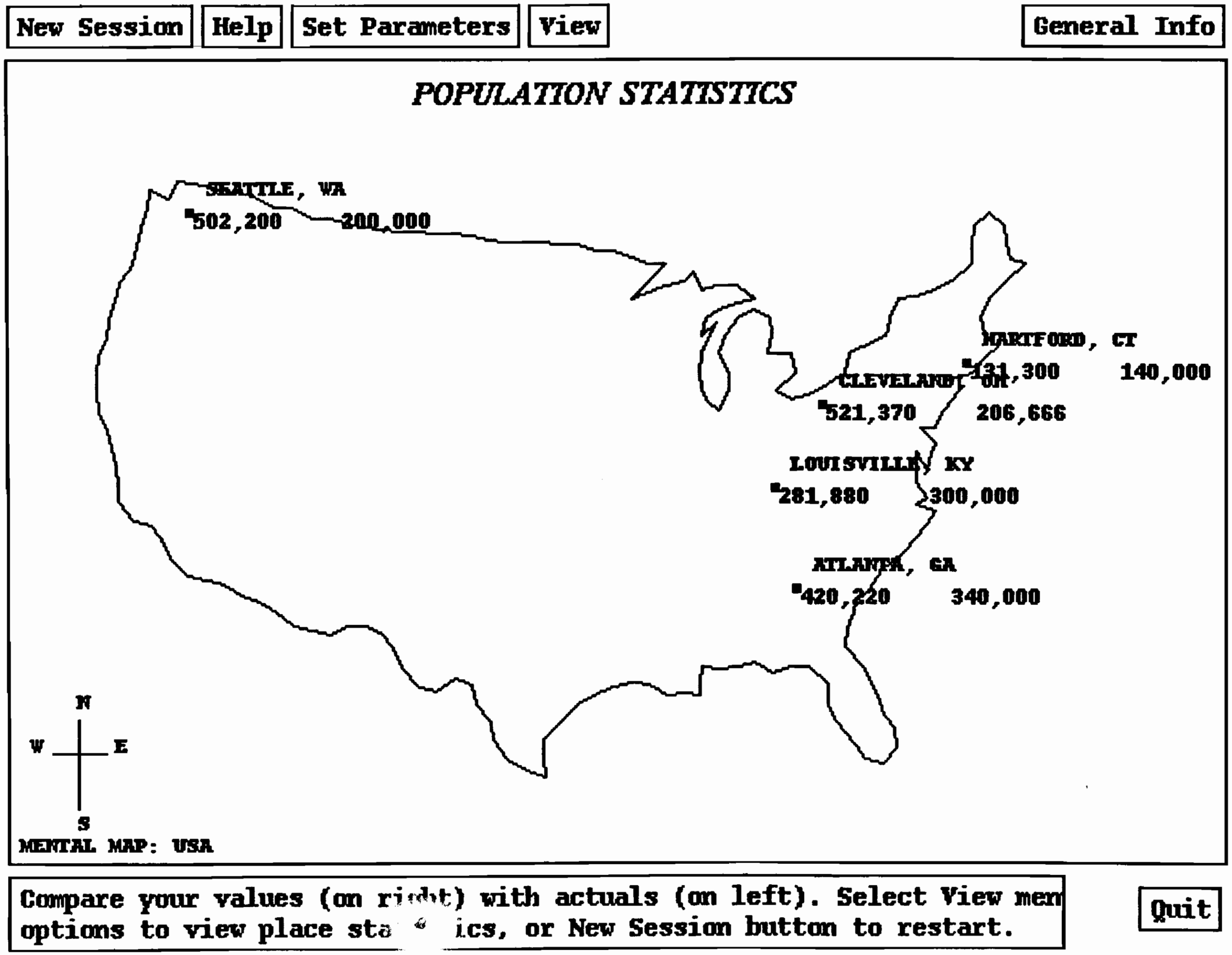

Figure 4.5: Display of Population Attributes

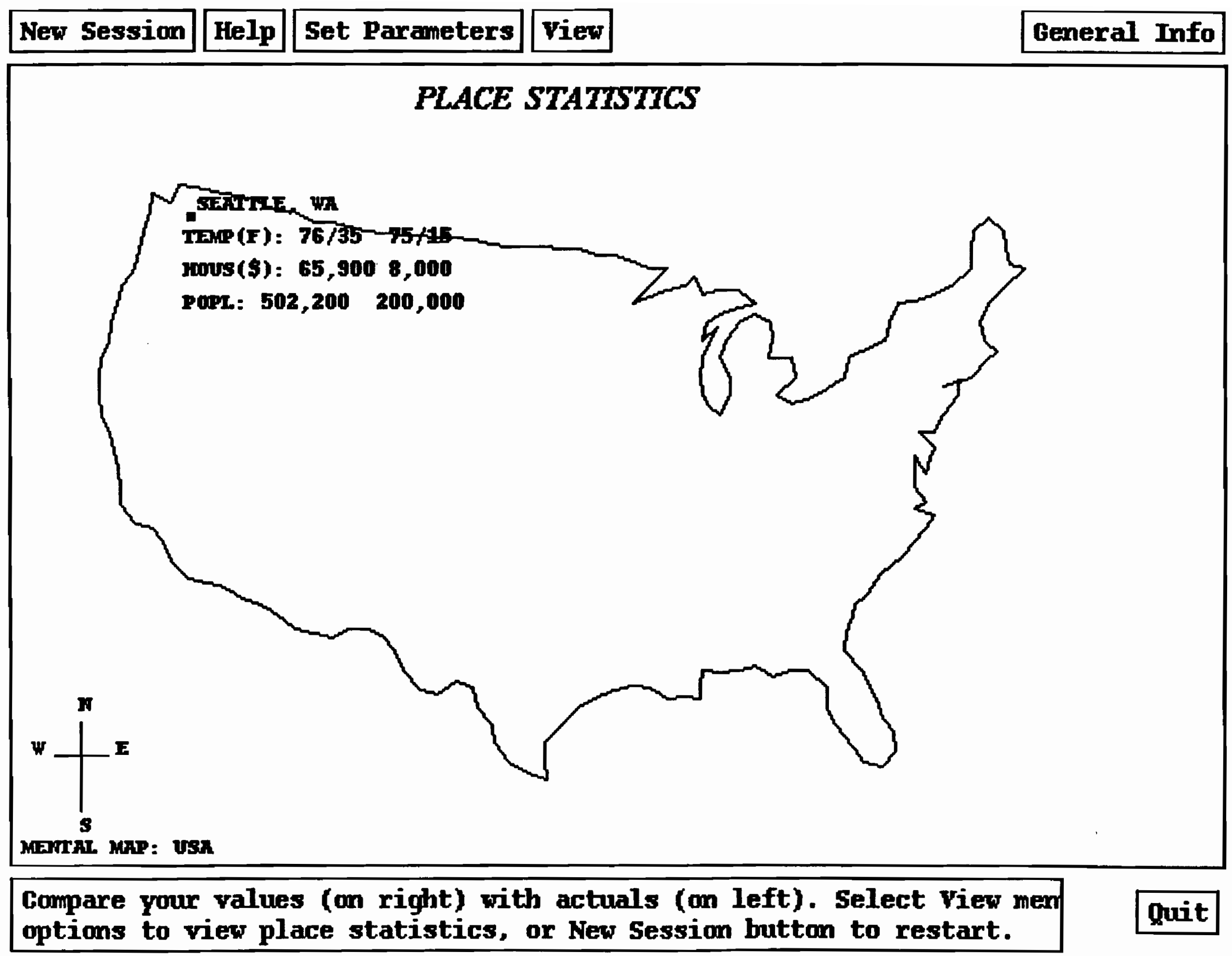

Figure 4.6: Display of Attributes for a Single Place

performance information will be collected for later use by the instructor. The third phase will be evaluation by the module software of the student's performance in order to provide some form of immediate feedback. All modules will contain an optional section describing the simulations and/or geometric algorithms used, as appropriate. These presentations will be geared for the novice computer user (as will all aspects of the modules). Some modules will provide post-exercise tutorials. These will be multi-media presentations directed in part by the student's performance on the exercise. Finally, the student will be given a chance to provide comments on the learning experience.

4.5 Databases

4.5.1 Input

The input files used to initialize the Mental Mapping module will be plain ASCII text files. Currently, a general reader is being developed to convert the general data file format to module specific input file format. The general reader will be common to all modules of project GeoSim. This general reader will scan the data file for key words such as population, climate, etc., and extract the data associated with these key words for use in the module. All irrelevant data will be ignored. The advantages of the general reader are: (i) It allows the tabulated, documented, human readable, general file format to be converted to machine readable format without any extra effort on the part of the user; (ii) It is common to all modules.

At present, the Mental Mapping module prototype requires two input files. One of them contains the x and y coordinates for the outline map and its format is as follows:

```
/* Top of File */
NAME OF COUNTRY/REGION
XCOORD, YCOORD
XCOORD, YCOORD
XCOORD, YCOORD
.
.
.
.
.
/* End of File */
```

XCOORD and *YCOORD* are reals. These have to be converted and/or scaled into actual pixel values so that the outline can fit into the 570 × 365 pixels graphics window so as to occupy most of it. The map outline is plotted as a series of straight lines joining each point to its adjoining point.

An example map outline input file would be as follows:

```
/* Top of File */
USA
8.0, 7.0
9.0, 7.5
9.5, 6.5
11.0, 6.8
12.0, 7.0
13.0, 7.1
14.0, 7.5
15.0, 7.9
16.0, 8.0
17.0, 8.5
...., ...
...., ...
...., ...
...., ...
...., ...
...., ...
...., ...
...., ...
...., ...
/* End of File */
```

The second input file contains details about cities and its format as follows:

/ TOP OF FILE */*

X Y POPULATION MAXTMP MINTMP HOUSING NAME STATE

.

.

.

.

.

.

/ END OF FILE */*

with:

X : Real value for the x coordinate of the location of city on map.

Y : Real value for the y coordinate of the location of city on map.

POPULATION : Integer value for the population of the city.

MAXTMP : Integer value for the maximum temperature in degrees Farenheit.

MINTMP : Integer value for the minimum temperature in degrees Farenheit.

HOUSING : Integer value for the average housing cost in dollars.

NAME : An ASCII string representing the name of the city.

STATE : A two character(ASCII) abbreviation for the state name.

The NAME and the STATE are read in as an array of characters.

An example city details file would be as follows:

/ Top of File */*

X	Y	POPULAT	MAX	MIN	HOUSE	NAME	STATE
52.5	17.5	7352700	85	26	54000	NEW YORK	NY
8.5	25.7	3352710	83	47	96100	LOS ANGELES	CA
39.1	19.0	2977570	83	15	47200	CHICAGO	IL
32.4	34.3	1698090	94	42	49100	HOUSTON	TX
....	....	....	..	..	.....	....	..
....	....	....	..	..	.....	....	..
....	....	....	..	..	.....	....	..
....	....	....	..	..	.....	....	..
....	....	....	..	..	.....	....	..
....	....	....	..	..	.....	....	..

/ End of File */*

One of the most useful results of this project to many users is the database. Despite the wealth of material commonly made available on computer networks, rarely does one find a geographic database that combines useful data, an accessible format, and clear documentation of that format. Even less often has the data format been explicitly designed for human readability. By storing our databases primarily in ASCII text files, it will be easy for others to use these data for their own purposes. Clear documentation is provided for every database format (including self documentation built into the format through fields tagged by names) along with separate programs for reading and interpreting data files.

4.5.2 Output

The output generated by the prototype can be used to determine the progress of the student through different sessions. Aggregate results for the class can illustrate both significant similarities and dissimilarities among a large group of students. The output file has the following format:

/ TOP OF FILE */*

<SESSION 1>

<HEADER for each of the fields below >

NAME STATE X Y POPULATION MAXTMP MINTMP HOUSING END_OF_LINE

.

.

.

.

.

<SESSION 2>

.

.

.

.

<SESSION N>

.

.

.

.

/ END OF FILE */*

The above values represent the student's input for the different sessions.

An example student's response file would be as follows:

/* *Top of File* */

NAME_OF_CITY	STATE	XCOORD	YCOORD	POPULATION	HOUSING
NEW YORK	NY	43.5	14.5	7352700	84000
LOS ANGELES	CA	9.5	20.7	4352710	66100
CHICAGO	IL	33.1	16.0	3977570	40200
HOUSTON	TX	30.4	30.3	2000000	60000
MEMPHIS	TN	37.4	16.3	1000000	30000

NAME_OF_CITY	STATE	XCOORD	YCOORD
INDIANAPOLIS	IN	53.2	13.5
ATLANTA	NM	20.5	19.4
NORFOLK	VA	43.1	26.0
OMAHA	NE	34.4	20.3
WICHITA	KS	27.4	21.3

NAME_OF_CITY	STATE	XCOORD	YCOORD	POPULATION
....	..	....	....	....
....	..	....	....	....
....	..	....	....	....
....	..	....	....	....
....	..	....	....	....

/* *End of File* */

Every session begins with a new header line NAME_OF_CITY ... etc.. The number of items in the header will vary depending on the number of parameters set by the instructor for each session. Also, the number of cities may vary from 5 to 20 depending on the number set by the instructor.

Since the actual values for the above cities are already available from the input files, the student's responses stored in the above format could serve as input to a program designed in the future to perform analysis of variances between actual values and student's responses.

Chapter 5

Summary, Conclusions and Future Work

The Mental Mapping module prototype provides a valuable introduction (or hopefully, re-introduction of topics studied in primary and secondary education settings) to several of the key themes of geographic education: location (position), place (physical or human characteristics), and regions (formation of regions of similar characteristics and distribution of those characteristics). This exercise allows the students to obtain feedback on the quality and accuracy of their "mental maps" and also to comment on several characteristics of places that should be at least marginally familiar.

The advantages of this module are its flexibility since it is portable across any platform supporting the X Window System, its WIMP interface which promotes active involvement rather than passive participation, its immediate feedback on the quality and success of decision making, and its window-based menu-driven interface with context-sensitive help making it easy to use. Also, this module forms part of a bigger system which should combine a powerful research tool (GIS) with a teaching/learning technique that is equally effective to create a new learning environment for hundreds of students per year.

5.1 Future Developments

Currently, the Mental Mapping module prototype is in the "proof of concept" phase. Next, the prototype will be integrated with other existing prototypes[1] to ensure that a common user interface is maintained. Since the X Window System supports networking protocols, expansion to support data from any part of the world can be allowed by accessing a remote database common to all modules. In order to be useful in teaching geography, the output from the module could be used as input by a report generation program which will allow the instructor to check on the progress of the student through the system. This report could also be used to track the interests and aptitudes of the students in order to indicate where to develop or improve coverage.

To educate the novice student in the use of the modules, a tutorial can be presented at the beginning of the actual learning session. Later, due to the advances in DVI technology, a multimedia tutorial on the cultural, climatic and demographic attributes can be presented to the student with a combination of text, still graphics, animation, sound and a wide range of special effects. This could make learning an interesting and enjoyable experience. Also, there is promise that DVI technology will be made available at a relatively low cost to future students.

The author along with Drs. Clifford Shaffer and Lawrence Carstensen has designed, implemented, and tested the prototype. Code documentation, and instructions for setup and operation have been provided in the resident directories.

GeoSim attempts to be more than a collection of computer modules. A system for creation of new modules, tools to help prepare digital multimedia presentations, a common user interface, common GIS and simulation functionality, and common database formats are significant aspects of its software efforts. These aspects are over and above the capabilities provided by the modules themselves.

1 Prototypes for the International Population, Reduction of Commuting Time, and Orienteering Modules are currently under development, with databases for the other two modules being researched.

REFERENCES

[Boye85] R. Boyer, and D. Savageau, *Places Rated Almanac: Your Guide to Finding the Best Places to Live in America*, Rand McNally and Company, Chicago IL, 1985.

[Cars90] L.W. Carstensen, C.A. Shaffer, R.W. Morrill, and E.A. Fox, GeoSim: A GIS Based Simulation Laboratory for Introductory Geography, *Proposal Submitted to the Undergraduate Curriculum and Course Development in Engineering, Mathematics and the Sciences Program of NSF*, Virginia Polytechnic Institute and State University, Blacksburg VA, 1990.

[Espe88] E.B. Espenshade, Jr.(Editor), J.L. Morrison (Consultant), *Goode's World Atlas*, Rand McNally and Company, Chicago IL, 1988.

[Fole90] J. Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics: Principles and Practice*, Addison Wesley Publishing Company, Reading MA, 1990.

[Goul74] P. Gould, and R. White, *Mental Maps*, Penguin Books, Baltimore MD, 1974.

[Hear86] D. Hearn, and M.P. Baker, *Computer Graphics*, Prentice-Hall Inc., Englewood Cliffs NJ, 1986.

[John89] E.F. Johnson, and K. Reichard, *X Windows Applications Programming*, MIS Press, Portland OR, 1989.

[Keat82] J. S. Keates, *Understanding Maps*, John Wiley and Sons, New York NY, 1982.

[Kern88] B.W. Kernighan, and D.M. Ritchie, *The C Programming Language*, Prentice-Hall Inc., Englewood Cliffs NJ, 1988.

[Luth89] A.C. Luther, *Digital Video in the PC Environment*, McGraw-Hill, New York NY, 1989.

[Magu89] D.J. Maguire, Demogis Mark 1: An Erdas-Based GIS Tutor, *Proceedings of the Ninth International Symposium on Computer-Assisted Cartography*, Baltimore MD, April 1989, 620-630.

[Murm89] E.M. Murman, The Athena Project: Goals, Philosophy, Status and Experiences, *Proceedings of the Nordic Conference on Computer Aided Learning in University and Higher Education*, Vol. I, Trondheim, Norway, June 1989, 1-8.

[Nye90] A. Nye, and T. O'Reilly,*The Definitive Guides to the X Window System - Volumes 1-5*, O'Reilly and Associates Inc., Sebastopol CA, 1990.

[Prep85] F.P. Preparata, and M.I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag Inc., New York NY, 1985.

[Rape89] J.F. Raper, and N.P.A. Green, Gist: An Object-Oriented Approach to a Geographical Information System Tutor, *Proceedings of the Ninth International Symposium on Computer-Assisted Cartography*, Baltimore MD, April 1989, 610-619.

[Shne87] B. Shneiderman, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison Wesley Publishing Company, Reading MA,1987.

[Shaf90] C.A. Shaffer, V. Miranda, J.C. Wang, S. Kriss, L.W. Carstensen, R.W. Morril, and E.A. Fox, Computerized Simulations for Introductory Geography Instruction, *Proceedings of Applications of Geographic Information Systems, Simulation Models and Knowledge-Based Systems for Landuse Management*, Blacksburg VA, November 1990, 479-480.

[Youn90] D.A. Young, *The X Window System Programming and Applications with Xt*, Prentice-Hall Inc., Englewood Cliffs NJ, 1990.