



Visualization of groundwater pollution using computer graphics
by James S Palakovich

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in
Computer Science

Montana State University

© Copyright by James S Palakovich (1990)

Abstract:

Water from subsurface systems comprises the largest majority of all the earth's drinking water, yet common waste disposal practices are continuously degrading the quality of our water resources. The contamination problem of subsurface water systems has been obvious for some time, although recently the problem has become progressively worse. In order to maintain the quality of our drinking water, it is necessary to develop techniques for extracting contaminating substances from the groundwater. When developing these extraction techniques, the predominant problem is visualizing the groundwater system. In this paper we describe a system that has been developed, using three dimensional computer graphics, to display models of groundwater systems including the watertable surface, the ground surface, and the pollution plume. We will show some examples of output using superficial data for a groundwater pollution system.

**VISUALIZATION OF GROUNDWATER POLLUTION
USING COMPUTER GRAPHICS**

by

James S. Palakovich

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

of

Master of Science

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

November 1990

11378
P1225

-ii-

APPROVAL

of this thesis submitted by

James S. Palakovich

This thesis has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the College of Graduate Studies.

Nov 27th 90
Date

J. Dunbar Stanley
Chairperson, Graduate Committee

Approved for the Major Department

Nov 27th 90
Date

J. Dunbar Stanley
Head, Major Department

Approved for the College of Graduate Studies

November 28, 1990
Date

Henry L. Parsons
Graduate Dean

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a master's degree at Montana State University, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgements of source are made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major Professor, or in his absence, by the Dean of Libraries when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature

James L. Paluszch

Date

November 27, 1990

TABLE OF CONTENTS

	Page
LIST OF FIGURES	v
ABSTRACT	vi
CHAPTER 1	
INTRODUCTION	1
CHAPTER 2	
SIMULATION MODEL	5
Modeling the Ground and Watertable Surfaces	5
Transparency	7
Modeling the Pollution Plume	10
Calculating the Distance Between Surfaces	13
Drawing Contour Lines	16
CHAPTER 3	
IMPLEMENTATION	18
Major Data Structures Used	18
Ground and Watertable Surfaces	18
The Pollution Plume	19
Determining Distance Between Surfaces	20
Drawing Contour Lines	21
Updating the Scene	22
CHAPTER 4	
CONCLUSIONS AND NEW DIRECTIONS	26
REFERENCES CITED	28

LIST OF FIGURES

Figure	Page
1. B-Spline Example	8
2. Transparent Surfaces	9
3. Pollution Plume Colors	11
4. Triangle Subdivision	12
5. Pollution Plume	13
6. Viewer Interaction	14
7. Contour Lines	17
8. Wire Frame Mesh	23
9. Spherical Coordinate System	25
10. Final Groundwater System	26

ABSTRACT

Water from subsurface systems comprises the largest majority of all the earth's drinking water, yet common waste disposal practices are continuously degrading the quality of our water resources. The contamination problem of subsurface water systems has been obvious for some time, although recently the problem has become progressively worse. In order to maintain the quality of our drinking water, it is necessary to develop techniques for extracting contaminating substances from the groundwater. When developing these extraction techniques, the predominant problem is visualizing the groundwater system. In this paper we describe a system that has been developed, using three dimensional computer graphics, to display models of groundwater systems including the watertable surface, the ground surface, and the pollution plume. We will show some examples of output using superficial data for a groundwater pollution system.

CHAPTER 1

INTRODUCTION

Contamination of the earth's water supplies poses a serious threat to the environment. Petroleum, chemicals, and toxic waste are polluting the oceans, lakes, streams, and groundwater supplies of the earth on a daily basis. While there has been a growing concern over surface water pollution, this concern has not been shared with the subsurface environment. In fact, many solutions to the surface water problems, such as deep-well injection of liquid wastes and sanitary landfill for solid wastes, can lead to increased groundwater pollution [2] and [10]. Another major problem with groundwater contamination is the length of time it takes for a water table to cleanse itself (residence time). River water has a residence time of approximately two weeks whereas groundwater moves slowly and residence times in the 10's, 100's, and even 1000's of years are not uncommon. To prevent the deterioration of groundwater quality, it has become necessary to develop a methodology for monitoring, analyzing, and predicting the movement of contaminants through groundwater. The hope of formulating such a methodology has motivated the development of many predictive tools in the form of mathematical models designed to simulate the transport of contaminants through groundwater systems [1] and [9]. This paper describes a visualization project that has been developed using computer graphics to monitor contaminant plumes which may be derived from these predictive methods.

Of all the earth's water reserves, groundwater is the most difficult to monitor. Because of the heterogeneities inherent in subsurface systems, zones of degraded groundwater can be very difficult to detect. The U.S. Environmental Protection

Agency has reported that almost every known instance of aquifer contamination has been discovered only after a water-supply well has been affected [5]. In order to detect and remove groundwater contamination, one must first understand the behavior of groundwater. Difficulty in understanding groundwater behavior is primarily due to the visualization obstacle. To truly understand the situation, it is necessary to have an accurate visual model of the contaminant system. Our project uses three dimensional computer graphics techniques to display the ground and water surface features needed for visualizing a groundwater pollution scenario.

The minimum visual information needed to understand a groundwater pollution scenario is: the ground surface features, the watertable surface features, the pollution plume, and the different concentration levels of the pollution plume. In addition to these basic features, two enhancements have been implemented to further increase the usefulness of our system: the ability to draw contour lines on both the ground and watertable surfaces and the ability to approximate the distances between the two surfaces. A brief discussion of how all of these features are used follows. A more detailed discussion is contained in Chapter 2.

In our visualization system, ground and water surface features are simulated using spline surfaces, or a collection of small patches which together form a smooth surface. To generate the spline surfaces used for the ground and watertable surfaces, a variable set of (x,y,z) positional data control points, usually in rectangular form, are needed for each surface. These control points are not actual points on the surfaces themselves, but instead act as weights to control the shape of the surfaces local to them. Because of this, the surfaces which are generated are not totally accurate, but they are a reasonably close approximation.

Generating the pollution plume requires a collection of data values which consist of an (x,y,z) position, and a contaminant concentration value from 1 to 10 (1 being the lowest concentration). The data points are used to create a collection of uniformly spaced points contained within a sphere, herein called a starburst. The reason for using these starbursts is to achieve a reasonable amount of anti-aliasing, or blending when all the starbursts are displayed at once. In other words, when all the starbursts have been generated and displayed, the result is a smooth, realistic looking pollution plume. Another advantage to this type of plume, as opposed to using a solid, opaque plume, is the ability to see the multiple concentration levels through one another.

Concentration levels of the pollution plume are shown using different color levels. Ten unique colors are used to display concentration levels from high, though moderate, to low. For instance, high concentration is shown using bright red as the color. Concentration levels between high and moderate are shown using colors which fade from red to yellow. Concentration values from moderate to low are shown by fading from yellow to cyan. These three colors (red, yellow, and cyan) provide good color separation so varying concentration levels can easily be distinguished.

The technique used for drawing contour lines on the ground and watertable surfaces is very similar to the technique used for generating the surfaces themselves. Equations which are used for generating the spline surfaces are also used for drawing contour lines. Minimum and maximum elevations and interval size are required for the contour drawing routine, along with the same set of control points used for generating the surface.

Calculating the distance between the ground and watertable surfaces is accomplished using an iterative method based on a two dimensional Newton's method [4]. An (x,y) coordinate pair, input by the viewer, is used to approximate the parameters $(s$ and $t)$ used to generate the surface. Given an initial guess for parameters s and t , each cycle of the iteration process generates a closer approximation of the actual s and t values. Iteration continues until the s and t fall within a given epsilon of their actual values. The iterative method is used on both surfaces with the same set of input.

In Chapter 3, we cover the data structures used in each of the above mentioned modules and discuss the screen updating process. Finally in Chapter 4, we cover conclusions and new directions and give some additional examples of program output.

CHAPTER 2

SIMULATION MODEL

This chapter contains a detailed description of each major component of the visualization system. Although some components are related to each other in one way or another, they will each be discussed separately under their corresponding sub-headings.

Modeling the Ground and Watertable Surfaces

The ground and watertable surfaces are probably the most important components of our system, both as visualization and discussion tools. As visualization tools, the surfaces direct the viewer's attention to the appropriate locations of the scene. As discussion tools, they set up the basis for virtually every other component of our system. Therefore, a great deal of this chapter will be concentrated on the discussion of these surfaces.

Ground and water surfaces are developed using a database of information about the surfaces. Each entity in the database contains an (x, y) coordinate pair, a watertable elevation z_1 , and a ground elevation z_2 . The coordinate (x, y, z_2) defines a point on the ground surface, while (x, y, z_1) defines a point on the watertable surface directly under (x, y, z_2) .

Constructing surfaces is based on a mathematical model for parametric cubic curves. A parametric cubic curve is a smooth curve which is defined using three parametric equations (one each for x , y , and z). Using parametric equations, instead of equations based on the x , y , and z variables themselves, eliminates the problem of handling infinite slopes. Parametric equations are functions of a single parameter

such as t , such that $0 \leq t \leq 1$. Instead of using slopes, parametric equations use tangent vectors which need never be infinite. Parametric cubic curve equations can easily be generalized to parametric bicubic surface equations by defining cubic equations of two parameters, s and t . Varying both parameters from 0 to 1 defines all points on a surface patch. If one parameter is assigned a constant value and the other parameter is varied from 0 to 1, the result is a cubic curve.

What follows is not intended to be an exhaustive discussion to the mathematics of parametric bicubic surfaces. For a more detailed discussion, see [6] and [3]. We will merely cover the mathematics sufficient for understanding of this document.

The evaluation of a point on a bicubic surface patch may be expressed as a matrix multiplication of the form:

$$x = X(s,t) = SMP_x M^T T^T, \quad (2.1)$$

$$y = Y(s,t) = SMP_y M^T T^T, \quad (2.2)$$

$$z = Z(s,t) = SMP_z M^T T^T. \quad (2.3)$$

where s and t are assumed to be between 0 and 1. A "T" superscript indicates the transpose of a matrix. Henceforth, only the equations for x will be stated explicitly; y and z are exactly analogous.

In the equations above, the vectors S and T are:

$$S = [s^3 \ s^2 \ s \ 1]$$

$$T = [t^3 \ t^2 \ t \ 1]$$

and matrix M in the B-Spline formulation is:

$$M = \frac{1}{6} * \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}$$

P contains the coordinates of the 16 control points defining the patch. In a B-Spline patch, P_x takes the following form:

$$P_x = \begin{bmatrix} x_{i-1,j-1} & x_{i,j-1} & x_{i+1,j-1} & x_{i+2,j-1} \\ x_{i-1,j} & x_{i,j} & x_{i+1,j} & x_{i+2,j} \\ x_{i-1,j+1} & x_{i,j+1} & x_{i+1,j+1} & x_{i+2,j+1} \\ x_{i-1,j+2} & x_{i,j+2} & x_{i+1,j+2} & x_{i+2,j+2} \end{bmatrix}$$

In the above matrix, $x_{i,j}$, $x_{i+1,j}$, $x_{i,j+1}$, and $x_{i+1,j+1}$ approximate the four corner points of the x patch. All other values are points which surround the patch.

The parametric surface defined above is called a B-Spline surface. B-Spline surfaces do not in general interpolate (pass through) any control points, but they are continuous and also have continuity of tangent vector and of curvature (that is, their first and second derivatives are continuous at the endpoints). Other types of surfaces which do interpolate the control points exist, but have continuity only for the first order derivatives, not for the second. A B-Spline surface is generally a smoother surface, but because the surface does not interpolate the control points, it is usually not a completely accurate approximation of the actual surface. Even so, B-Splines are still well suited for modeling the ground and water surfaces of our system. A sample watertable surface using B-Splines can be seen in Figure 1.

Transparency

Transparent surfaces greatly enhance the visual understanding of the system. For instance, it is necessary to be able to see the pollution plume under the water surface, or see the water surface under the ground surface. The transparency technique used in our system is called screen-door transparency [8].

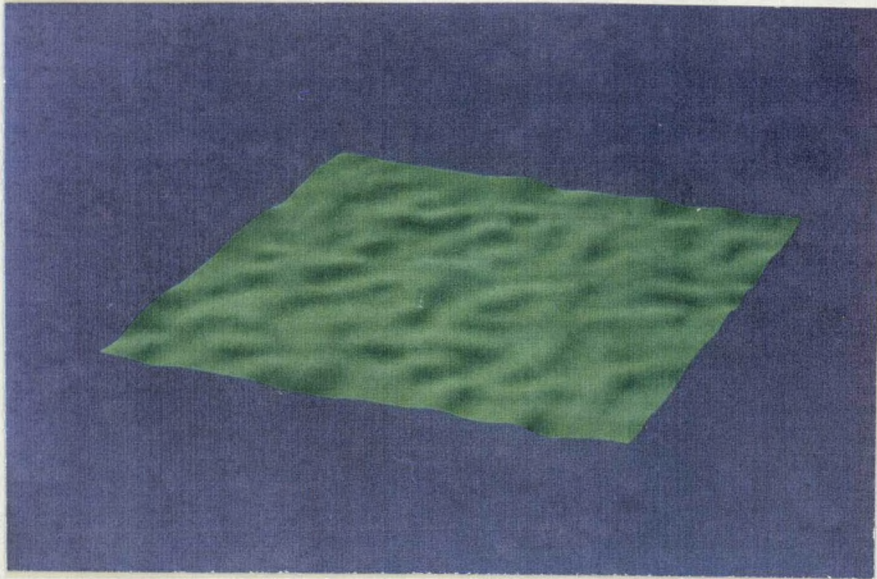


FIGURE 1
B-Spline Example

Screen-door transparency is implemented by imposing a bit mask which defines which pixels can be affected by subsequent graphics primitives. A 4×4 cell, which is duplicated over the entire screen, contains 16 bits which can be set to 0 or 1 (pixel-off and pixel-on respectively). Thus, there are 17 levels of transparency: 0 (opaque) through 16 (completely transparent), based on the total number of bits set to 1. The bits in the mask correspond to the bits in the 4×4 transparency pattern in the following manner:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

maps into:

0	1	2	3
4	5	6	7
8	9	10	11

12	13	14	15
----	----	----	----

To prevent the transparency's "texture" from dominating the scene, the bits in the transparency mask must be distributed as evenly as possible when viewed as a 4x4 square. The following pattern is used:

1	13	4	16
9	5	12	8
3	15	2	14
11	7	10	6

That is, place the first write-protected pixel in location 1, the second in location 2, and so forth. In this way, the transparency pattern is as smooth as possible. Figure 2 shows a ground surface with transparency level 14 over the water surface with transparency 7.

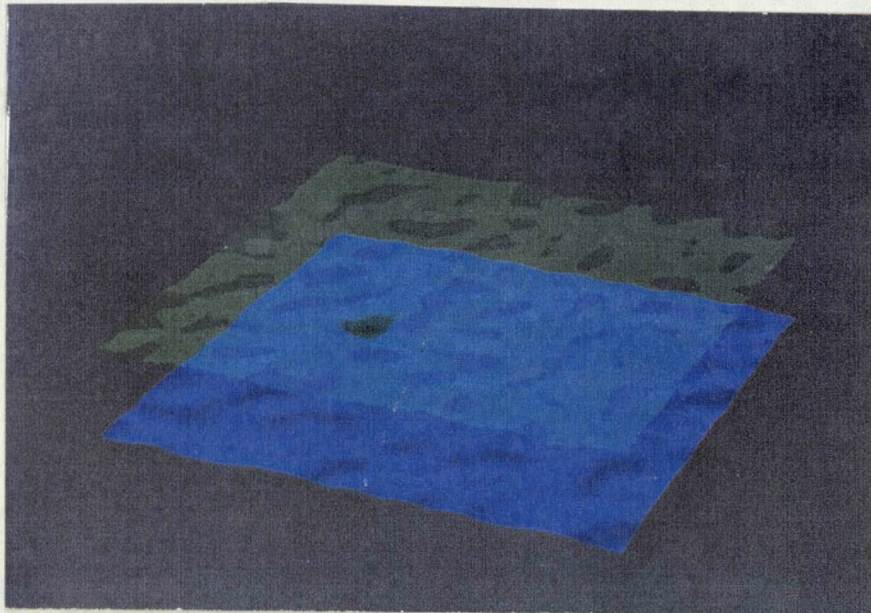


FIGURE 2
Transparent Surfaces

Modeling the Pollution Plume

The pollution plume in our system is a graphical display of contaminant which has been dissolved in the water. Pollutants such as petroleum, which float on the water surface instead of dissolving, are not considered, but our system could be generalized to accommodate this situation (see Chapter 4). Instead, our system will display ten levels of dissolved pollutant concentration using ten distinct colors to separate the levels.

The U.S. Environmental Protection Agency has established water quality standards for drinking water that vary to a large degree depending on the constituent. For example, the recommended concentration limit for chlorine (Cl) in drinking water is 250 mg/l, while the limit for hydrogen sulfide (H_2S) is 0.05 mg/l. Because of the radical differences in the EPA limits, it is up to the user to decide the range for each of the ten concentration levels in the system. Concentration levels are represented using a scale from 1 to 10. A concentration level of 1 would be a very low percentage of pollution to water and a level 10 would be very high. The ten colors used by the system are shown in Figure 3.

A database is used for storing information about the plume. Each entity of the database contains an (x,y,z) location and a concentration level. The number of separate entities in the database is unbound, it is dependent only on the application. Each entity is used by the system to generate a single unit of the pollution plume on the screen (a starburst).

A starburst is a set of three concentric spherical shells. On the surface of each shell are a set of points (of each pixel are equidistant from that pixel. Each shell of the starburst contains progressively more points as the shells progress outward.

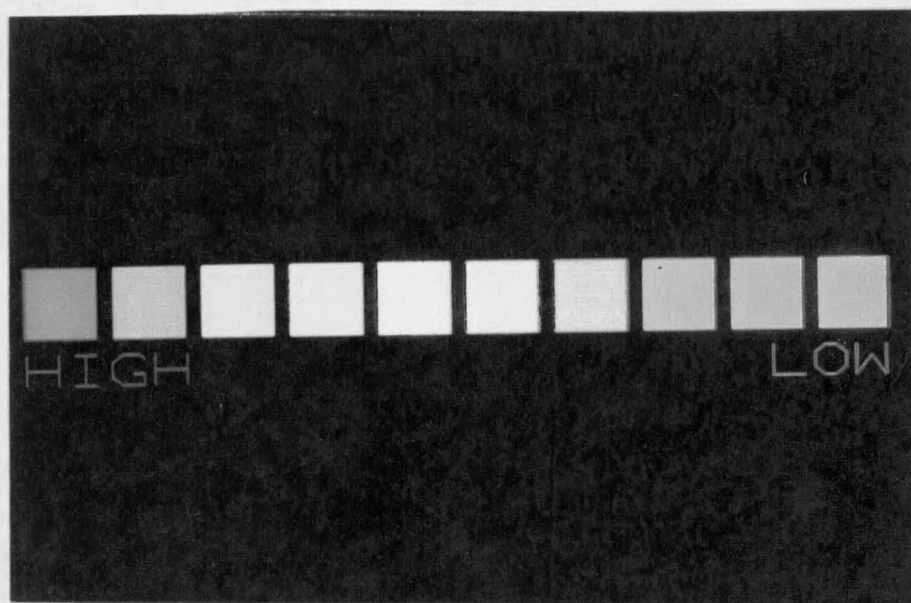


FIGURE 3
Pollution Plume Colors

A triangular approximation of each shell of the starburst is used to obtain equidistant points. A triangle is an excellent approximation because the three vertices of an octant of a sphere form an equilateral triangle. We first start with an octahedron as our approximation of the sphere and use recursive subdivision to divide each triangular side of the octahedron into four equal triangles. Each input triangle with vertices labeled $[0, 1, 2]$ as shown in Figure 4 will be transformed into four new triangles $[0, b, a]$, $[b, 1, c]$, $[a, b, c]$, and $[a, c, 2]$. The subdivision process makes three new points, namely a , b , and c , by calculating the midpoint of each of the sides of the triangle. Points a , b , and c must then be normalized before constructing the four new triangles. Normalizing the points is accomplished using the following equation:

$$\frac{1}{\sqrt{a^2+b^2+c^2}}$$

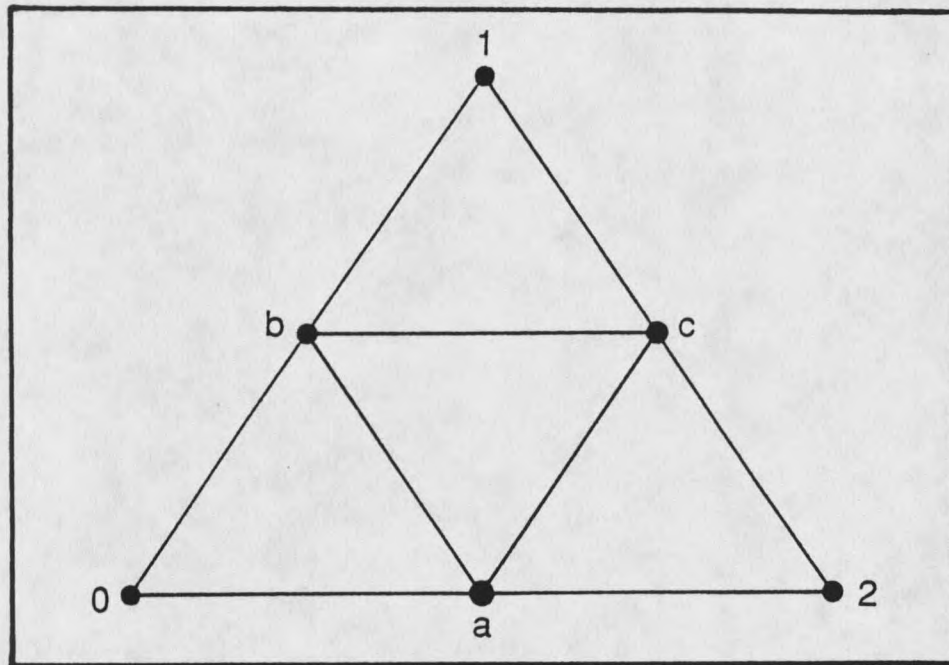


FIGURE 4
Triangle Subdivision

Glassner [7] describes another method of generating equidistant points on the surface of a cylinder and mapping them onto the surface of a sphere.

In our system, a starburst is used, as opposed to a solid sphere, to add depth to the plume. A solid sphere reduces the field of view of the plume to that of a two-dimensional solid, whereas a starburst allows one to view multiple concentration levels through one another. A starburst also adds an inherent color blending effect to the plume. A fully generated pollution plume is shown in Figure 5.

Notice that only the surface of the water is shown in Figure 5. In actuality, the water should fully surround the pollution plume. Our system displays only the water surface to reduce visual confusion between the water and the plume. It is much easier to examine the pollution plume without the obstruction of the water.

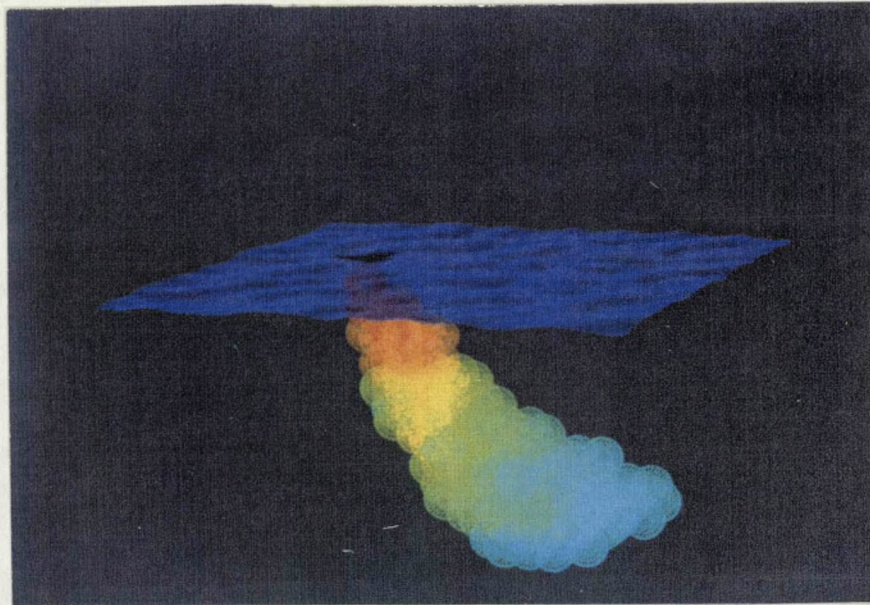


FIGURE 5
Pollution Plume

Calculating the Distance Between Surfaces

Knowing the distance between the earth's surface and the watertable surface can be very useful information, especially when determining the cost of extracting pollution from groundwater. The most significant portion of the cost of a pollutant extraction process is drilling a well (or series of wells).

Our system has the capability of approximating the distance between the ground and water surfaces by calculating the elevations of both surfaces and subtracting the watertable surface from the ground surface. Approximating the elevations of the surfaces is accomplished using equations 2.1, 2.2, and 2.3:

$$x = X(s,t) = SMP_x M^T T^T, \quad (2.1)$$

$$y = Y(s,t) = SMP_y M^T T^T, \quad (2.2)$$

$$z = Z(s,t) = SMP_z M^T T^T. \quad (2.3)$$

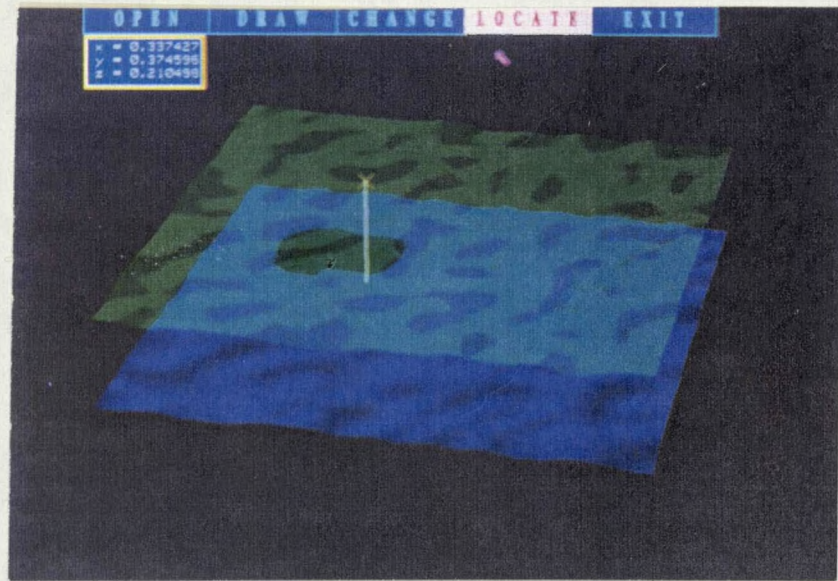


FIGURE 6
Viewer Interaction

Equation 2.3 is the equation used for determining the elevation, based on an (x,y) location on the ground surface. Here, x and y are supplied by the user via an interactive routine which allows the user to select a point on the ground surface (see Figure 6). The x and y values, in conjunction with equations 2.1 and 2.2 respectively, can be used to approximate the parameters s and t . Once s and t have been found, the approximate elevations of the two surfaces can be obtained directly from equation 2.3.

The reason the term approximate is used when referring to calculating elevations is because the method which can be used to calculate the parameters s and t is an iterative approximation technique based on Newton's method for approximating zeroes in functions. In the case of a polynomial equation in one variable, Newton's method has a simple geometric interpretation. We must have an initial guess for the solution to the equation $F(x) = 0$. If x_0 is our initial guess, we calculate the tangent

$F'(x_0)$. The next approximation, x_1 , is given by the point where the tangent crosses the x -axis. For a more detailed explanation of Newton's method, see [4].

In this document, we will limit ourselves to the application of Newton's method to the solution of a system of two polynomial equations (namely s and t). We want to determine the values of s and t where $(X(s,t) - x) = (Y(s,t) - y) = 0$, and we have two initial guesses s_0 and t_0 , which in our case are both 0.5 (the center of the surface patch). The next, or better, approximations are given by:

$$S_{i+1} = S_i + \Delta S, \quad (2.4)$$

$$T_{i+1} = T_i + \Delta T, \quad (2.5)$$

where ΔS and ΔT are the Newton steps:

$$\Delta S = \frac{G \frac{\partial F}{\partial t} - F \frac{\partial G}{\partial t}}{\frac{\partial F}{\partial s} \frac{\partial G}{\partial t} - \frac{\partial F}{\partial t} \frac{\partial G}{\partial s}}$$

and

$$\Delta T = \frac{F \frac{\partial G}{\partial s} - G \frac{\partial F}{\partial s}}{\frac{\partial F}{\partial s} \frac{\partial G}{\partial t} - \frac{\partial F}{\partial t} \frac{\partial G}{\partial s}}$$

After the Newton steps have been calculated and s_0 and t_0 iterated, the new values of s_1 and t_1 are plugged into the same equations to get successively closer approximations to the solution. Iteration continues until $X(s_i, t_i) - x$ and $Y(s_i, t_i) - y$ are both sufficiently close to zero.

Once s and t have been found, the elevations of both the ground and watertable surfaces can be calculated by direct substitution into equation 2.3. Because the

internal representation of both surfaces is identical, there is no need to repeat this process for each surface.

Drawing Contour Lines

The final step in our visualization model is superimposing contour lines on the ground and water surfaces. Contour lines greatly enhance the viewer's ability to interpret the surfaces. Figure 7 shows two pictures of a watertable surface, displayed with and without contour lines. Even with the shading effects due to the light source, the addition of contour lines greatly enhances the viewer's interpretation of the surface. This enhancement is most noticeable at the well in the upper left-hand corner of the surface.

Because the spline surfaces used in this system were calculated in hardware, the positional data used for displaying these surfaces is unavailable. Therefore, the B-Spline surface equations used for generating the surfaces must be re-used to draw the contour lines:

$$x = X(s,t) = SMP_x M^T T^T, \quad (2.1)$$

$$y = Y(s,t) = SMP_y M^T T^T, \quad (2.2)$$

$$z = Z(s,t) = SMP_z M^T T^T. \quad (2.3)$$

The basic idea of the contouring algorithm is to acquire the elevations of the surface at each point on the surface. If a point happens to be at the same elevation as a desired contour line, the contour line is drawn through that point. Contour lines are calculated, one patch at a time, for each patch on the surface. A full discussion of this algorithm can be found in a document pending publication, by Starkey [11].

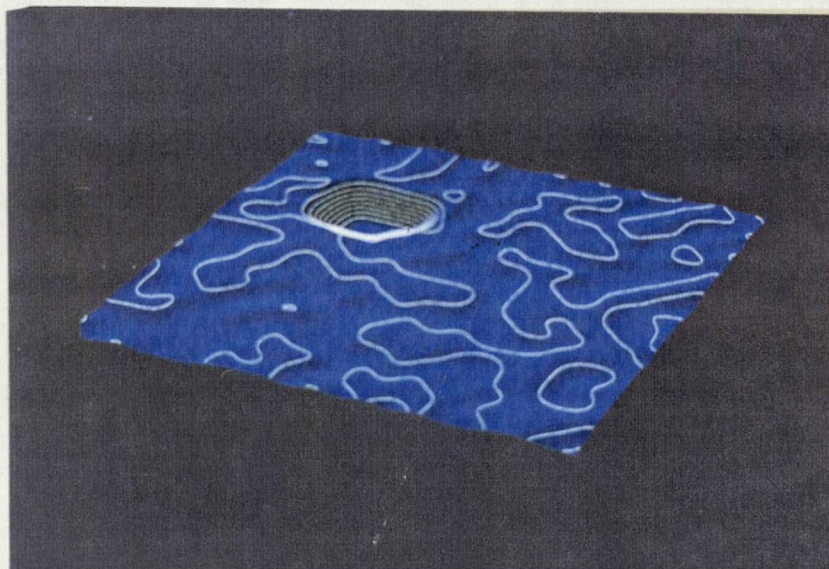
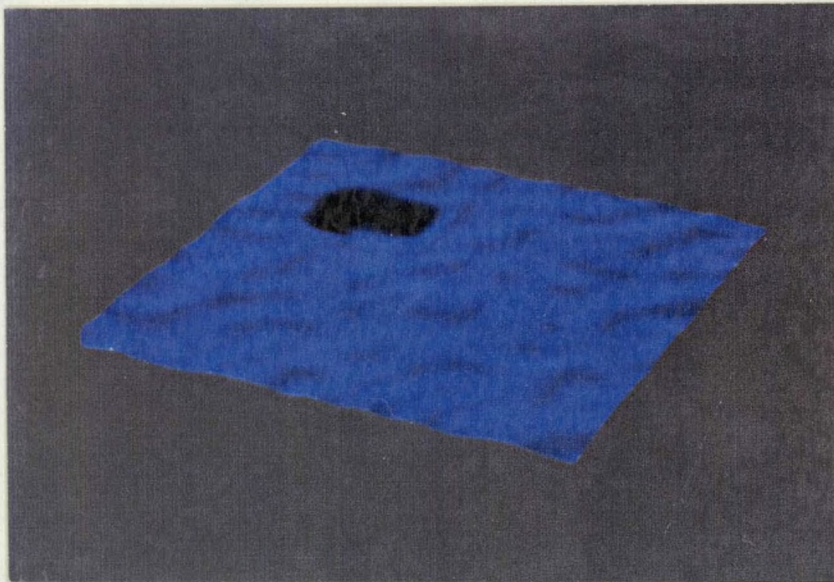


FIGURE 7
Contour Lines

CHAPTER 3

IMPLEMENTATION

We now move our discussion of the implementation of our visualization system which was developed on a Hewlett Packard 350SRX graphics workstation using HP-UX version 6.0 and the Starbase graphics package. The 350SRX has a high-speed graphics accelerator which performs many of its graphics routines in hardware to gain the extra speed needed for compute-intensive routines.

In this chapter, we will discuss the data structures and scene updating of our system.

Major Data Structures Used

As with any computer program, every module in the program has its own data structures to control the organization of the data within the module. Even though its data structures are unique to each module, they must also be used to bind the modules together to form the final working system. In this section, we will discuss the data structures used by each module of our program.

Ground and Watertable Surfaces

Both the ground and watertable surfaces use identical data structures because the two surfaces, in reality, are similar with the exception of their elevations. The surfaces each use a three-dimensional array of (x,y,z) control points (*ground_pts_3d* and *water_pts_3d*) which are passed to the surface generating routine. The size of the arrays are determined dynamically from the rectangular size of the array (the number

of rows and columns) which is stored as the first line in the surface control point database.

Calculating the surfaces themselves is done in hardware using the graphics accelerator of the HP 350SRX. Because of this, the data structures of the B-Spline surface algorithm will not be discussed. For a thorough explanation of the process, consult the Hewlett-Packard Starbase Graphics Reference [8].

The Pollution Plume

Because the plume is made up of many simple entities (starbursts) which are independent of each other, the only data structures used by the module are the data structures used for the starburst. The pixel values which make up a starburst are calculated once and stored using the following C structures:

```
typedef struct {  
    float      x, y, z;  
} point;
```

```
typedef struct {  
    point      pt[3];  
    double     area;  
} triangle;
```

```
typedef struct {  
    int        npoly;  
    triangle   *poly;  
} object;
```

```
object  star_points1, star_points2, star_points3;
```

where *star_points#* are the three shells of the starburst. The number of pixel values stored in each of the structures is proportional to the density of the starburst at each shell. For example, all starbursts used for this paper have 128, 512, and 2084 points

on each shell respectively. This corresponds to 3, 4, and 5 levels of recursion of the starburst generation algorithm for each shell respectively.

Each starburst is drawn independently, that is, the position, color, etc, of one starburst does not affect any other starburst. After the plume is completely drawn, it appears to the viewer that the plume has a real structure, but actually, this is only an illusion created by clustering many independent starbursts together.

Determining Distance Between Surfaces

Determining the distance between the ground and watertable surfaces requires calculating the approximate elevations of both surfaces. This process is identical for both surfaces, therefore the same data structures are used in both cases. The algorithm requires a 4x4x3 array of (x,y,z) control points called *patch* which defines the patch on which the locator is positioned, along with its nearest neighbors. The algorithm breaks the input matrix into three 4x4 matrices *x_patch*, *y_patch*, and *z_patch*, which are used to build *x_mat*, *y_mat*, and *z_mat* in the following way:

```
MatMatMatMult(spline_mat, x_patch, spline_mat_trans, x_mat);  
MatMatMatMult(spline_mat, y_patch, spline_mat_trans, y_mat);  
MatMatMatMult(spline_mat, z_patch, spline_mat_trans, z_mat);
```

where *spline_mat* is a 4x4 matrix for storing the B-Spline matrix and *spline_mat_trans* is the transpose of *spline_mat*. The three input matrices: *spline_mat*, *x_patch*, and *spline_mat_trans*, are multiplied together using matrix multiplication giving *x_mat* (*y_mat* and *z_mat* are exactly analogous). Once *x_mat* and *y_mat* have been calculated, they are used by Newton's method for finding the *s* and *t* parameters.

Newton's method uses two vectors s_vec and t_vec which are the vectors S and T defined in equations (2.1, 2.2, and 2.3). These two vectors are used in the iteration process for finding parameters s and t . Once s and t have been calculated, the elevation of the surfaces is obtained by:

$$z_val = \text{VecMatVecMult}(s_vec, z_mat, t_vec);$$

which takes a vector, a matrix, and a vector, and outputs a scalar. The elevations of both surfaces are found by using a z_mat from each surface.

Drawing Contour Lines

The contouring algorithm takes as input either *ground_pts_3d* or *water_pts_3d* (described earlier in this chapter), depending on which surface contour lines are desired, and stores it in a structure called *in_surface*. The algorithm breaks *in_surface* into single $4 \times 4 \times 3$ arrays called *patch* and draws contours on each patch one at a time. *Patch* contains the (x,y,z) values of each of the four corners of the patch and their nearest neighbors. When calculating *patch* for patches which lie on the edge of a surface, it is impossible to get the nearest neighbors for the side of the patch on the surface edge. To solve this problem, we use another structure called *big_surface*. If *in_surface* is $N \times M \times 3$, then *big_surface* is $(N+2) \times (M+2) \times 3$. In other words, *big_surface* is simply *in_surface* with extra control points around each of the outer four edges. The outside edges of *big_surface* are determined in the following way:

```
for (i = 1; i <= rows; i++)
    for (j = 0; j < 3; j++) {
        big_surface[i][0][j] = 2 * big_surface[i][1][j]
        - big_surface[i][2][j];
        big_surface[i][cols+1][j] = 2 * big_surface[i][cols][j]
        - big_surface[i][cols-1][j];
    } /* end for */
```

The above routine defines the outside rows of *big_surface*, the outside columns are exactly analogous.

As the contouring algorithm operates on a single patch, it keeps track of the intersection points of the square grid using an $N \times 2$ array of s and t values called *outlist*, where N is $Resn * 8$. *Resn* is a user defined constant used for determining the step size of s and t . For example, if *Resn* is 5 then the step size for s and t would be 0.2.

When all intersection points have been found the points defined in *outlist* are connected. If there are only two points in *outlist* a straight line is used to connect the points, otherwise a spline curve is used.

Updating the Scene

Having the ability to view the system from any angle is very important to fully understanding the given groundwater pollution scenario. Every time the viewer wishes to change the reference angle, the entire system must be reconstructed piece by piece. Because of the computational time complexity involved in displaying the system, each component of the system is displayed separately. That is, the ground surface, the watertable surface, the pollution plume, etc, are all displayed as separate entities. This way, the viewer can find the exact desired reference angle before generating the entire scene.

To speed up the angle locating process, the water surface is initially drawn as a wire frame mesh as shown in Figure 8. The wireframe is drawn almost instantly using the graphics accelerator of the 350SRX. By using this wireframe representation of the water surface, the desired viewing angle can be obtained easily and quickly

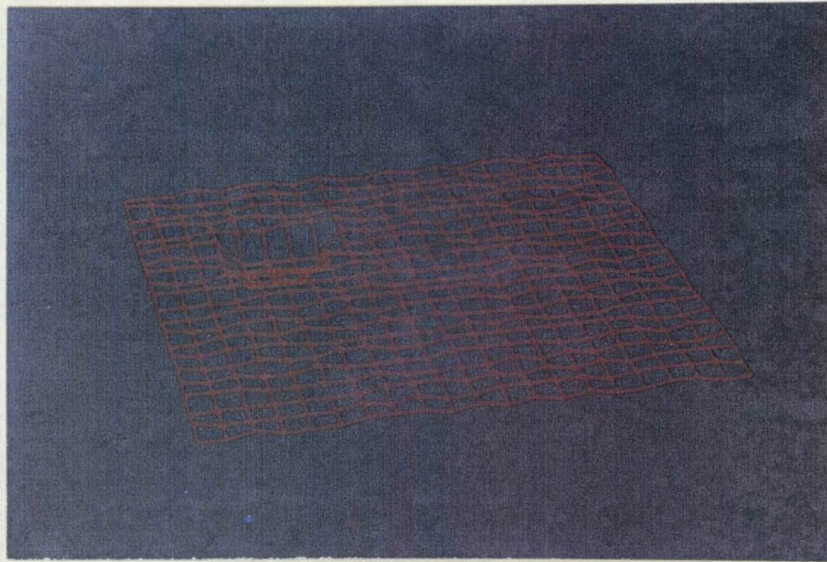


FIGURE 8
Wire Frame Mesh

before beginning the B-Spline surface generation.

Interaction with the computer is controlled through a device called a knob box. A knob box is an input device with nine dials (or knobs) which can be moved left or right. Each knob can be defined to vary input over a specified range. When a knob is turned all the way to the left, the input from the knob would be at the minimum value in the range. As the knob is turned to the right, the input value becomes progressively larger until the maximum range value is reached. When the viewer is interacting with the knob box, the knob positions are constantly being monitored. As soon as the computer senses that a knob has been moved, the picture is updated.

For our system, only six of the nine knobs are used. The first three knobs control the position of the camera relative to the water surface and the second three knobs control the camera reference point on the water surface, which is initially

defined to be the exact center of the surface. Moving the camera is done using spherical coordinates (coordinates in *rho* (ρ), *theta* (θ), and *phi* (ϕ)).

Figure 9 shows the spherical coordinate system. Angle θ defines the camera angle relative to the positive y -axis, angle ϕ defines the camera angle relative to the positive x -axis, and length ρ defines the distance from the camera to the origin of the coordinate system.

Spherical coordinates are used, as opposed to Cartesian coordinates, because moving an object using spherical coordinates is easier to mentally visualize than moving an object in cartesian coordinates (i.e. the object can be moved left or right, up or down, or in or out, and the object always stays a distance ρ from the reference point on the water surface).

Converting from spherical coordinates to cartesian coordinates can be done using the following equations:

$$x = \rho \sin\phi \cos\theta \quad (3.1)$$

$$y = \rho \sin\phi \sin\theta \quad (3.2)$$

$$z = \rho \cos\phi \quad (3.3)$$

After finding the desired reference angle, the water surface can be drawn using B-Spline surfaces as described earlier. The time required to display any of the B-Spline surfaces used in our system is approximately between 10 and 15 seconds. To display the entire system including the ground surface, the water surface, the pollution plume, and contour lines on either or both surfaces can take anywhere from one to two minutes. Therefore, the wireframe representation greatly reduces the time and complexity of finding the desired reference angle.

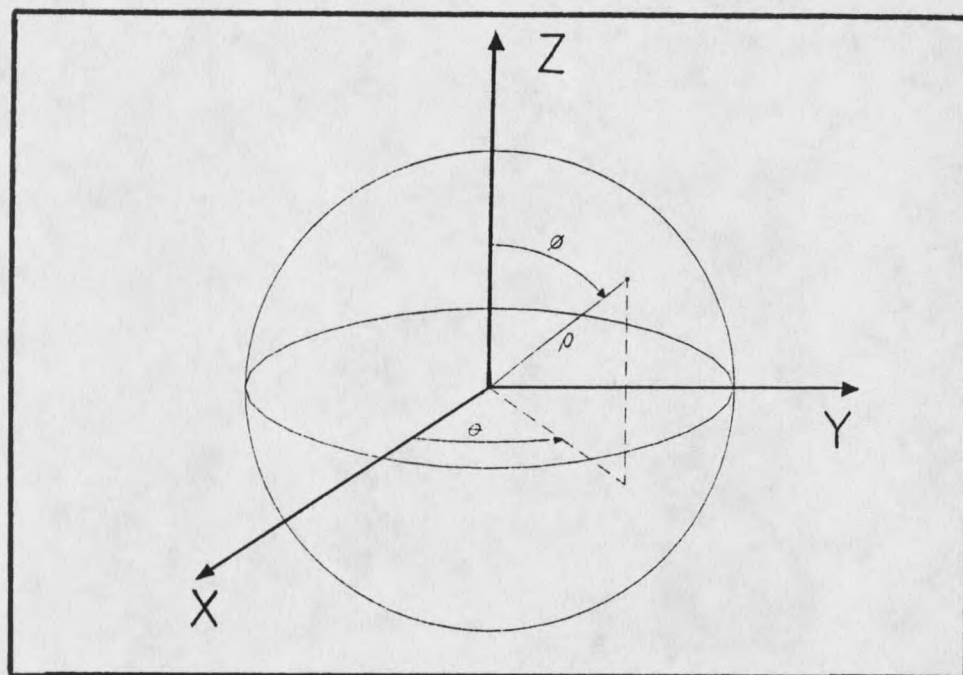


FIGURE 9
Spherical Coordinate System

CHAPTER 4

CONCLUSIONS AND NEW DIRECTIONS

The system described in this document is a computerized visualization tool used to study the effects of pollution in subsurface watertables. The system includes the features necessary to view the ground and watertable surfaces and a plume of dissolved contaminant. Figure 10 shows a final example of our visualization system. The ground surface is shown with contour lines and a transparency index of 11 while the water surface has a transparency index of 7.

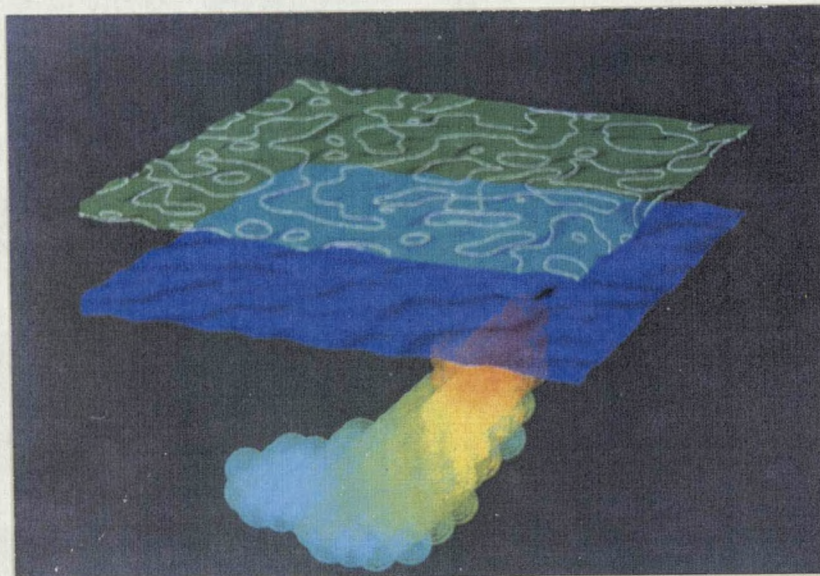


FIGURE 10
Final Groundwater System

Although our system is an excellent tool for visualizing a groundwater scenario for the dissolved pollutant case, it does not deal with the floating pollutants such as petroleum. To generalize our system for the non-dissolved case, a new pollutant display scheme must be developed. One such scheme is to display the pollutant as another B-Spline surface, superimposed over the water surface. Pollutant surfaces

would not necessarily be the same shape, or color, as the water surfaces so the two could be easily distinguished. The multiple color feature could be used, in this case, to signify the depth of the pollutant at any particular location.

Another important addition to the system would be the ability to incorporate into the system, any of the mathematical predictive methods currently available for estimating pollution flow over an extended time interval [1] and [9]. With such capabilities, the system could, when given the initial conditions, predict the shape of the pollution plume after any length of time.

A research grant is currently being sought to further develop this system to include the features mentioned above. Much remains to be done to reach the ultimate goal of a fully interactive visual and predictive tool for studying groundwater pollution.

REFERENCES CITED

1. Anderson, M. P., Using Models to Simulate the Movement of Contaminants Through Ground Water Flow Systems, CRC Critical Reviews in Environmental Control, v.9, 1979.
2. Ballentine, R.K., Reznick, S. R., Hall, C. W., Subsurface Pollution Problems in the United States, EPA Technical Studies Report, TS-00-72-02, 1972.
3. Bartels, R. H, et al, An Introduction to the Use of Splines in Computer Graphics. Notes for SIGGRAPH 85 Tutorial: The Use of B-Splines and Beta-Splines to Model Freeform Curves and Surfaces, 1985.
4. Faux, I. D., Pratt, M. J., Computational Geometry for Design and Manufacture, John Wiley, 1979.
5. Freeze, R. A., Cherry, J. A., Groundwater, Prentice-Hall, 1979.
6. Foley, J. D., van Dam, A., Feiner S. K., Hughes J. F, Computer Graphics: Principles and Practice, Second Edition, Addison-Wesley, 1990.
7. Glassner, A. S., Graphics Gems, Academic Press, 1990.
8. Hewlett-Packard Company, HP-UX Concepts and Tutorials: Starbase Graphics Techniques, v.2, HP 9000 Series 300 Computers.
9. Mendoza, C. A., McAlary, T. A., Modeling of Ground-Water Contamination Caused by Organic Solvent Vapors, Journal of Association of Ground Water Scientists and Engineers, v28-2, 1990.
10. Rima, D. R., Chase, E. B., Myers, B. M., Subsurface Waste Disposal by Means of Wells - A Selective Annotated Bibliography, U.S. Geological Survey Water Supply Paper 2020, 1971.
11. Starkey, J. D., Contours on Spline Surfaces, Pending Publication, 1990.

MONTANA STATE UNIVERSITY LIBRARIES



3 1762 10075097 3

