A Visualization System for Tidal Basins

Kenrick Mock
CS 470 – Sample Project Write-up

April 5, 2010

Table of Contents

A Visualization System for Tidal Basins

Kenrick Mock

Abstract

A tidal basin is composed of a body of water adjacent to a coastline and includes the waterway to the open ocean.   Developers and scientists are often interested in such basins. For example, developers may be interested in the effects of erosion and how the basin might be shaped to minimize erosion. Scientists may be interested in how an oil spill will propagate throughout the basin. This project, BasinViz, was developed to read and parse data files containing properties of a tidal basin. The properties are then shown graphically and animated so that an analyst can easily visualize what is happening in the basin. A user interface was also developed so that the analyst can focus on a particular area or property of interest.

1. Introduction

This project was developed for a small Anchorage business named Coastal Engineering. Coastal Engineering studies the processes involved at the shoreline within the coastal zone using practices derived from civil engineering. The direction of the work is often aimed at combating the erosion of coasts, studying the effects of pollution, or providing navigational access through waterways.   For example, projects that have been studied by the firm include an analysis of coastal erosion in Kotzebue and a simulation of dispersion for a potential oil spill in Cook Inlet.

Coastal Engineering is composed of only two individuals, both with Ph.D.'s in engineering. Although one of the employees possesses self-taught knowledge of computer programming, I was hired on a contractual basis to perform the bulk of the software development. Since the company is very small, our interactions were quite informal.

2. Project Overview

The goal of this project is to develop an automated visualization system for tidal basins. Tidal basins are of particular interest due to the potential for development that may occur around the basin.   This system will help study the effects of contaminants that may drain into the basin from land or contaminants that may be produced as the result of a spill or accident. However, the scope of this report is limited to the technical details of the visualization system itself, not how an engineering analysis may be developed as a result of the visualization.

2.1 Data Files

A basin is represented by as a grid of two-dimensional cells. Each cell represents a small square region of land or water. A set of five text files contains properties regarding each cell. Each property is described below:

- Depth. This file contains the depth to the basin floor for each cell. Land is represented by a value of -99.
- Elevation. This file contains the height of the water in each cell for each time period. The values vary as the tide ebbs and flows.
- U velocity. This represents the x velocity of the water in a particular cell for each time period.
- V velocity. This represents the y velocity of the water in a particular cell for each time period. When paired with the U velocity, we get a vector indicating the magnitude and direction of flow for a particular cell.
- Concentration. This represents the percentage of contaminant, silt or other material in a particular cell for each time period.

The data values are generated from one of two methods:

1. Measurement from a physical model. In this approach, a small-scale replica is built of the actual basin. For example, the replica might be 10 by 10 feet and built of clay. Dye is placed in the model and taped with a video camera to measure the velocities.

2. Generation from a computer model. The bulk of the data for BasinViz was generated by a computer model written in FORTRAN.

The data files are all in text format and values are delimited by white space. Here is a short sample of a 4x3 Depth file. In practice, most files are of dimensions at least 50x50:

```
-99.00 -99.00   3.96   3.96
-99.00   3.96   3.96   3.96
-99.00   3.96   3.96  -99.0
```

This represents a basin shown in figure 1, where green is land and blue is water (in this case, to a depth of 3.96 units).

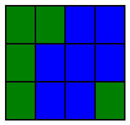


Figure 1. Basin Depth

All other text files are organized by periods of time. For each time period, the value of every 2D cell is specified. The following is an example of a 4x3 concentration file with three time periods. Most files contain at least 50x50 cells and over 40 time periods:

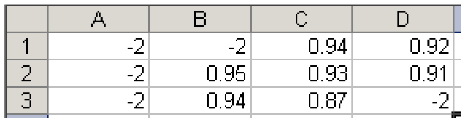
```
VERTICALLY AVERAGED BOD DAYS=  2.0833334E-02  HOURS=  0.5000000
-2.00 -2.00   1.00   1.00
-2.00  1.00   1.00   1.00
-2.00  1.00   1.00  -2.00
 VERTICALLY AVERAGED BOD DAYS=  4.1666668E-02  HOURS=   1.000000
-2.00 -2.00   0.98   0.94
-2.00  0.98   0.95   0.93
-2.00  0.97   0.94  -2.00
 VERTICALLY AVERAGED BOD DAYS=  4.1666668E-02  HOURS=   1.500000
-2.00 -2.00   0.94   0.92
-2.00  0.95   0.93   0.91
-2.00  0.94   0.87  -2.00
```

This file contains concentration values for the basin for each time period. In this case, the time period is 30 minutes. The value of -2 corresponds to cells representing land and should be ignored. In this case, the second cell on the bottom has a concentration value of 1.0 in the initial time period, 0.97 in the second time period, and 0.94 in the third time period.

The elevation and velocity files are organized in a manner that is similar to the concentration file, except each value represents elevation, u velocity, or v velocity. For large basins, each file could be up to 5 Mb in length.

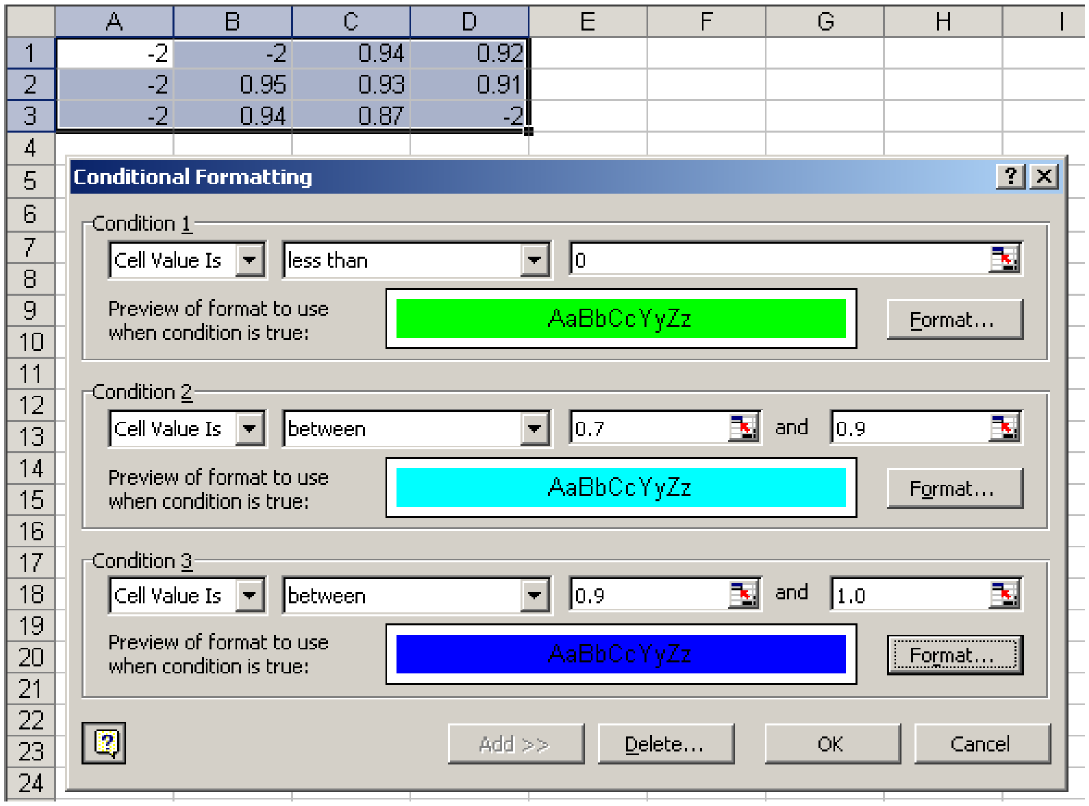
2.2 Prior Visualization Process

Prior to the development of this system, the firm used Microsoft Excel to visualize the basin properties. The section of the file of interest was imported into Excel and the background color of each cell was formatted depending upon its value. For example, to visualize the data contained in hour 1.5 of the concentration example in section 2.1, the data was first imported into Excel:



	A	B	C	D
1	-2	-2	0.94	0.92
2	-2	0.95	0.93	0.91
3	-2	0.94	0.87	-2

Next, conditional formatting was applied to assign colors to different data ranges:



	A	B	C	D
1	-2	-2	0.94	0.92
2	-2	0.95	0.93	0.91
3	-2	0.94	0.87	-2

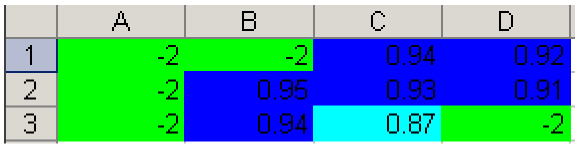
Conditional Formatting

Condition 1: Cell Value Is | less than | 0 — Preview of format to use when condition is true: AaBbCcYyZz

Condition 2: Cell Value Is | between | 0.7 and 0.9 — Preview of format to use when condition is true: AaBbCcYyZz

Condition 3: Cell Value Is | between | 0.9 and 1.0 — Preview of format to use when condition is true: AaBbCcYyZz

The result is a color display of the data:



	A	B	C	D
1	-2	-2	0.94	0.92
2	-2	0.95	0.93	0.91
3	-2	0.94	0.87	-2

This process was repeated for all data of interest. Needless to say, this manual process was very time-consuming if a project contained a large amount of data. The goal of this system was to entirely automate the visualization process.

3. Project Requirements

The requirements set forth by Coastal Engineering for BasinViz were fairly general. The firm did not have an exact product in mind, but had a general idea of what they wanted the product to do. Due to this grey area, I employed the prototyping methodology and expressed the requirements in English rather than using a more formal method.   A danger of using English is the higher likelihood for requirements to change; consequently, I allocated additional time for refinement later in the development process. As described in section 5, several of the requirements listed here were actually added late in the development process after initial prototypes had been created.

3.1 Functional Specifications

1. The system must display a dialog box to allow the user to select which data set to read from disk and read all data from disk into memory.

2. A graphical display should be generated that corresponds to the tidal basin in the data set. The display should have the proper aspect ratio but be maximized to fit the dimensions of the video screen.

3. A "play" button should be placed on the application that begins animating the graphical display for each time step at a user-specified speed.

4. Concentration and Elevation values should be displayed by allowing the user to set cell colors for different data ranges. A grayscale option is mandatory, and colors are optional. There must be at least 3 different data ranges available.

5. In addition to the concentration values, there should also be an option to display the following quantity, denoted E, for each cell:

$$E\_cellVal = 1 - \left(concentration\_cellVal\right)^{\frac{12}{timeValue}}$$

6. A velocity vector must be computed for each cell based upon the u and v data values. The vector should be displayed as an arrow pointing in the specified direction from the center of each cell.

7. There should be a zoom feature to magnify a portion of the tidal basin.

8. The elevation of the tide should be graphically displayed in relation to the current time step.

3.2 System Specifications

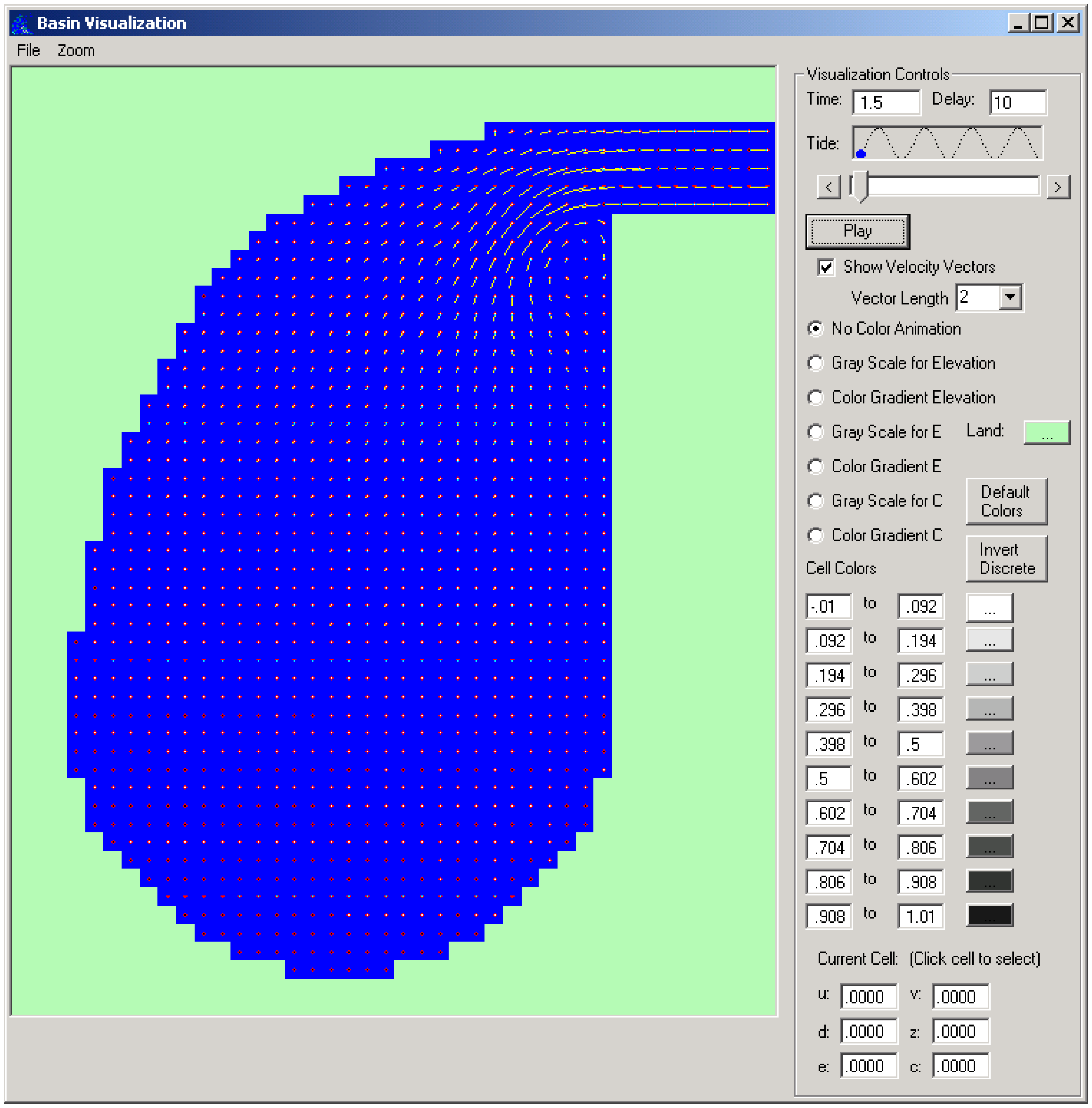
The system must be constructed using Visual Basic 6.0 running Windows 2000 or Windows XP. The resulting executable must also run using Windows 2000 or Windows XP but the end-user system might not have Visual Basic installed. At a minimum, the system should have 256 Mb of memory and run at a speed of 450 MHz or higher. The video adapter must be capable of a resolution of at least 1024 by 768 pixels.

4. System Design

Due to the selection of Visual Basic 6.0 as the programming language, a true object-oriented design is not possible. However, a modular decomposition and event-based design was performed.

4.1 User Interface Design

A majority of the work centers on a single form, frmBasin. This form contains the graphical visualization of the basin along with all of the controls available to the user. A screenshot of the form is shown below in figure 2.



Time: 1.5 Delay: 10

Tide:

[✓] Show Velocity Vectors

Vector Length 2

[● No Color Animation]

○ Gray Scale for Elevation

○ Color Gradient Elevation

○ Gray Scale for E Land:

○ Color Gradient E

○ Gray Scale for C

○ Color Gradient C

Cell Colors

-.01	to	.092	...
.092	to	.194	...
.194	to	.296	...
.296	to	.398	...
.398	to	.5	...
.5	to	.602	...
.602	to	.704	...
.704	to	.806	...
.806	to	.908	...
.908	to	1.01	...

Current Cell: (Click cell to select)

u: .0000 v: .0000

d: .0000 z: .0000

e: .0000 c: .0000

Figure 2. Main form, frmBasin

The main image is contained within a picturebox control. A second, hidden picturebox was used for off-screen drawing as described in section 4.4. The controls on the right of the screen are for:

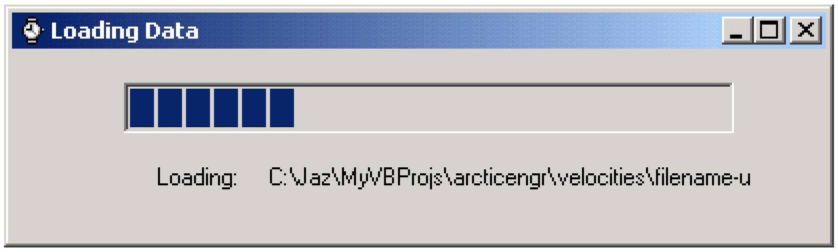
- Time : displays current time slice
- Delay : User-entered parameter to control the speed of the animation

- Tide : A picturebox that shows the elevation (tide) in relation to the current time value, depicted as a blue circle. The slider below the tide is tied to the tide picturebox.
- A "Play" button to begin animating the visualization.
- A checkbox whether or not to display vectors and a pull-down menu to select the length of the vector
- A set of radio buttons to select what property to color code each cell
- Buttons to reset default colors or invert the colors
- Textboxes indicating the data ranges for the selected value, and a button to select the color that should be drawn in each cell for the selected range
- Textboxes that show the u, v, c, e, depth, and elevation values for the selected cell.

Under the menu options are:

- F)ile, O)pen : to browse for data files and open them
- F)ile, E)xit : to exit the application
- Z)oom, Zoom option : To select a zooming level

A second form, shown in figure 3, was used to display a progress bar while the data files are loading.



Loading: C:\Jaz\MyVBProjs\arcticengr\velocities\filename-u

Figure 3. Progress Bar Form.

4.2 Data Structures

The data structures used in BasinViz are relatively simple as the underlying data consists primarily of numeric tables. To facilitate a quick animation, all of the data files were read into memory. The depth file was simply read into a two-dimensional array of type double.

The elevation, velocity, and concentration files are a bit more complicated since they each contain an entire 2D array of doubles for each time value (single precision could have also been used to lower memory usage). A natural data structure to use here is a three-dimensional array, with time as one dimension and the X and Y indices as the other dimensions.

However, to simplify coding I used a dictionary object instead of an array to store the elevation, velocities, and concentrations. The dictionary object is part of the Microsoft Scripting Runtime object and it allows for dynamic content-based access to data. The data stored in the dictionary can be a simple data type or an object. For example, a developer can add items to the dictionary using a primitive value as key and then retrieve the item using the key.

In this application, I used time as the key and an entire 2D array of doubles representing the entries for that time as the value. This allows direct access to an entire 2D array of data given a time value. Consequently, there is no need to search the dictionary for the array that corresponds to a particular time. Alternately, the dictionary also allows us to access the data using a traditional index, as in an array. This is illustrated in figure 4:

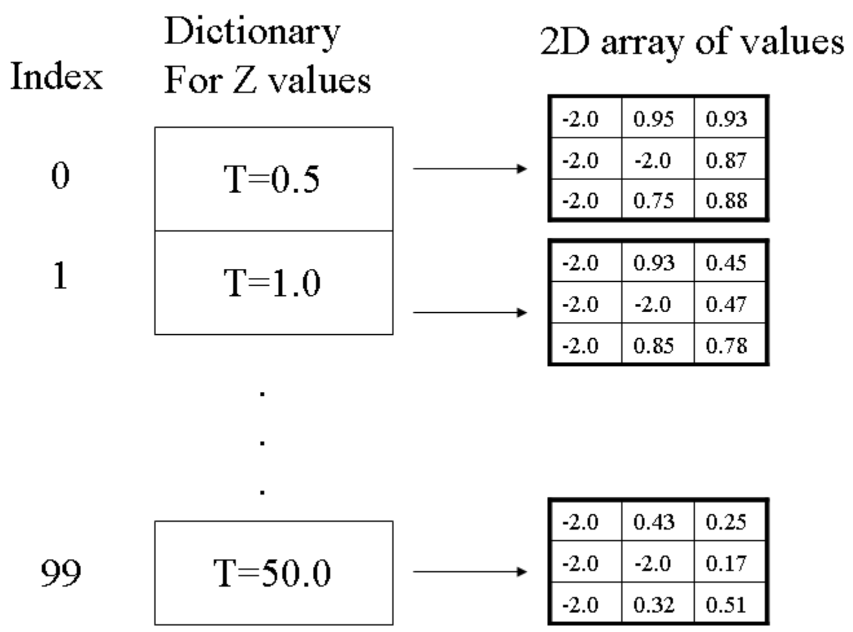


Figure 4. Dictionary object to store entire 2D arrays using time as index

By utilizing the dictionary object to store data, implementation was simplified since the dictionary could be constructed dynamically and accessed by time or by index. All of these access methods were employed in the program. Furthermore, Visual Basic handled memory management details, freeing the developer from explicitly allocating and de-allocating memory.

4.3 System Architecture

The overall system was decomposed into the major modules shown in figure 5.

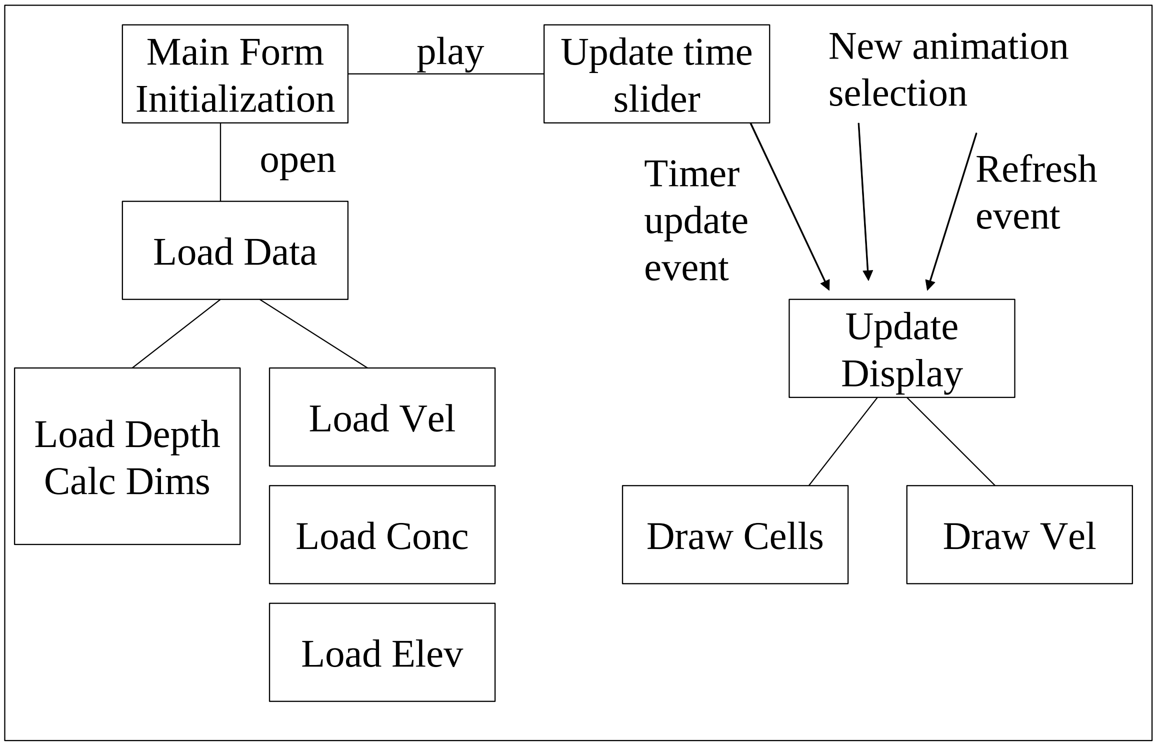


Figure 5. System Architecture

Upon initialization of variables (e.g., the dictionary objects) the user must first select data files to open. This will invoke the modules that parse and load the data into memory. The depth file will be read first, and this is used to calculate the dimensions of the basin and allocate the proper size arrays. Furthermore, cells that correspond to land are stored in the baseDepth() array so that these values can be ignored in the other data files. This helps to speed up data processing and is also necessary since values that correspond to land should not be included in water-based calculations.

After the data has been read, the user may play the animation. This initiates a loop that updates a slider control for each time value.

Tied to the event of updating the slider is code that updates the graphical display. To determine what should be updated, a control path is executed that examines the current selections on the form. If the radio button for elevation is checked, then elevation values will be drawn in all of the cells. Similarly, if the radio button for E is checked then E values are drawn, or if the radio button for concentration is checked then concentration values are drawn.

To draw the cells in the proper colors, the update code retrieves the 2D array from the appropriate dictionaries based on the selected property (concentration, E, or elevation). This 2D array can be directly retrieved from the dictionary using the current time value as a key. Next, the cells are drawn in the colors selected by the user on the form.

After the cells have been drawn, velocity vectors are computed and then drawn over each cell. Drawing must be performed in XOR mode to ensure that the vectors are visible on any background color.

By modularizing the graphical update code we can invoke this routine based upon any event, not just the event of updating the slider. For example, if the user resizes the display, we must redraw the screen. Similarly, if the user decides to animate some other feature, such as the elevation instead of the concentration, then we can simply invoke the update procedure to freshen the display.

4.4 Algorithms

Most of the algorithms in BasinViz are fairly straightforward; it primarily involves bookkeeping to ensure that the proper user selections are reflected in the animated visualization. However, some care must be taken in the algorithm to re-draw the screen. If the colors of each cell are updated sequentially, then the update will be visible on the screen. The result will be an uncomfortable flickering of the display during the animation as portions of the display are displayed in their correct new color while other portions are still in the old color. For example, we may have the proper vectors and colors in the top half of the screen for time=1.0, while the bottom half of the screen is still in the old color and the old vectors for time=0.5.

To rectify this problem, the solution that is typically used in animation is to draw on a "hidden screen". After the hidden screen is completely updated with all new images, the entire hidden screen is copied to the visible screen during the vertical blanking interval. The end result to a user is that the entire screen is updated instantly with no visible flickering.

To implement this algorithm, I used a hidden picturebox to represent the hidden screen. The visible screen is represented by a visible picturebox. The process of updating the screen is to take the following steps:

- Draw new cell colors on the hidden screen
- Draw new vectors for velocities on the hidden screen
- Copy entire hidden screen to the visible screen

The drawing routines were implemented using Visual Basic's DrawRect procedure, which can draw lines and rectangles in different colors and fill patterns. The copy procedure was implemented using the PaintPicture method supplied in Visual Basic.

5. Software Development Process

Due to the somewhat vague requirements of BasinViz, I used the prototyping development methodology. I implemented an initial prototype, showed it to the company, gathered new requirements, designed and implemented them, and repeated the process until the product was deemed worthy of meeting the corporation's needs.

5.1 Testing and Debugging

I spent a majority of my time in testing and debugging. Several of the bugs that took the most time to debug were actually the least significant. For example, I spent several hours trying to determine why a few pixels for the slider were not properly drawn inside their picturebox when the slider had reached a maximum or minimum tide. Instead, they were drawn outside of the visible picturebox. This was hardly critical, as the loss of a few pixels was hardly noticeable. However, I was ultimately able to track down the bug as a miscalculation of data ranges.

Another insidious bug arose from the way that Visual Basic creates arrays. In some cases, Visual Basic prefers arrays to start at index 1. In other cases (such as retrieving a record from a dictionary object) arrays start at index 0. Unfortunately this scenario arises several times in my code. In one case, I assumed that the array began at 1 when it really began at 0. This resulted in an "off by one" error that also took several hours to debug.

A large number of other bugs were related to refresh events. To detect these and other bugs, I periodically ran the following tests after performing a major update:

- Load two different basins with different aspect ratios and ensure that they are loaded and displayed properly
- Play the animation color coded for depth with and without vectors
- Play the animation color coded for E with and without vectors
- Play the animation color coded for C with and without vectors
- Select four cells and ensure that their values are reflected in the text boxes
- Select one cell and cross reference its values displayed in the application with the data values in the raw text file to ensure they are the same
- Resize the screen and ensure that the image aspect ratio is correct
- Occlude and minimize the screen and ensure the display is properly refreshed
- Test all zooming modes and play the animation for E with and without vectors
- Test switching between all zooming modes and no zoom
- Test inverting the colors
- Test changing the color for land and for one of the data range values
- Enter a manual data range value and test that the colors are updated accordingly
- Test manually dragging and clicking the slider

5.2 Prototyping Challenges

The lack of formal requirements and use of prototyping also presented several challenges. The most significant challenge was the addition of new requirements after viewing the prototype. When new requirements arise late in the development cycle they may often be very expensive to implement. Fortunately the new requirements did not require significant changes to the code and data structures, but did require some time.

After viewing the initial prototype, two new requirements were added. The first requirement specified that the animation speed should be selectable by the user. While the animation speed was satisfactory on my development machine, the animation was too fast on the client's much faster machine. This feature was implemented by adding a delay setting. The second requirement specified that a zooming feature should be added. For large basins, it was difficult to see what was happening in certain regions. The implementation of the zoom feature required a significant amount of code to graphically select the zooming region via rubber-banding, compute the new magnification factor, and redraw the display to only the selected data range.

After the first prototype, the client also requested a change in how vectors were drawn. Initially the client request that vectors be drawn as an arrow centered in each cell. I spent a large amount of time writing code that was capable of drawing arrows with different size arrowheads. However, the client determined that the arrowheads were distracting and cluttered the interface. Instead, the company requested that arrows be drawn simply as a vertical line with a red circle in the center. This new arrow can be drawn quite easily in just a few lines of code. I found it somewhat frustrating to discard the rather lengthy code drawing arrowheads that I had spent so much time perfecting, but in the end I do believe it made the interface more readable.

After viewing the second prototype, another requirement was added. The client decided that it would be useful to see the elevation of the water (i.e. the tide) correlated with the current time value. Together we drew up an interface that accomplished this graphically by integrating the tide into the slider. The tide was drawn as a sine wave and a circle was plotted on the wave as it correlated to the current time slice. In this fashion we created a graphical slider that showed tide and time.

Finally, one rather interesting bug arose relating to notation. In plotting the velocity vectors, I used the notation that is standard in computer graphics: the origin is in the upper left hand corner and positive X is to the right and positive Y is to the bottom. However, the engineer that output the data files used the mathematical notation that the origin is in the lower left hand corner and positive X is to the right and positive Y is to the top. As a result, the vectors were drawn incorrectly. Somewhat surprisingly, this bug was not caught until the second prototype, as the vectors appeared correct until inspected more closely and correlated with the direction of the tide in the raw data files.

5.3 Work Breakdown

The Gantt chart in figure 6 shows the anticipated and actual work schedule. The values in green represent the projected amount of time per task determined during the planning stage of the project, and the values in blue represent the actual amount of time taken. The total amount of actual time is less than the projected schedule. This discrepancy is primarily because it was easier to use the picturebox control than I anticipated to draw the elements and perform the animation. There are also some breaks in the actual schedule when I did not work on the project or was waiting to schedule a meeting with the client.

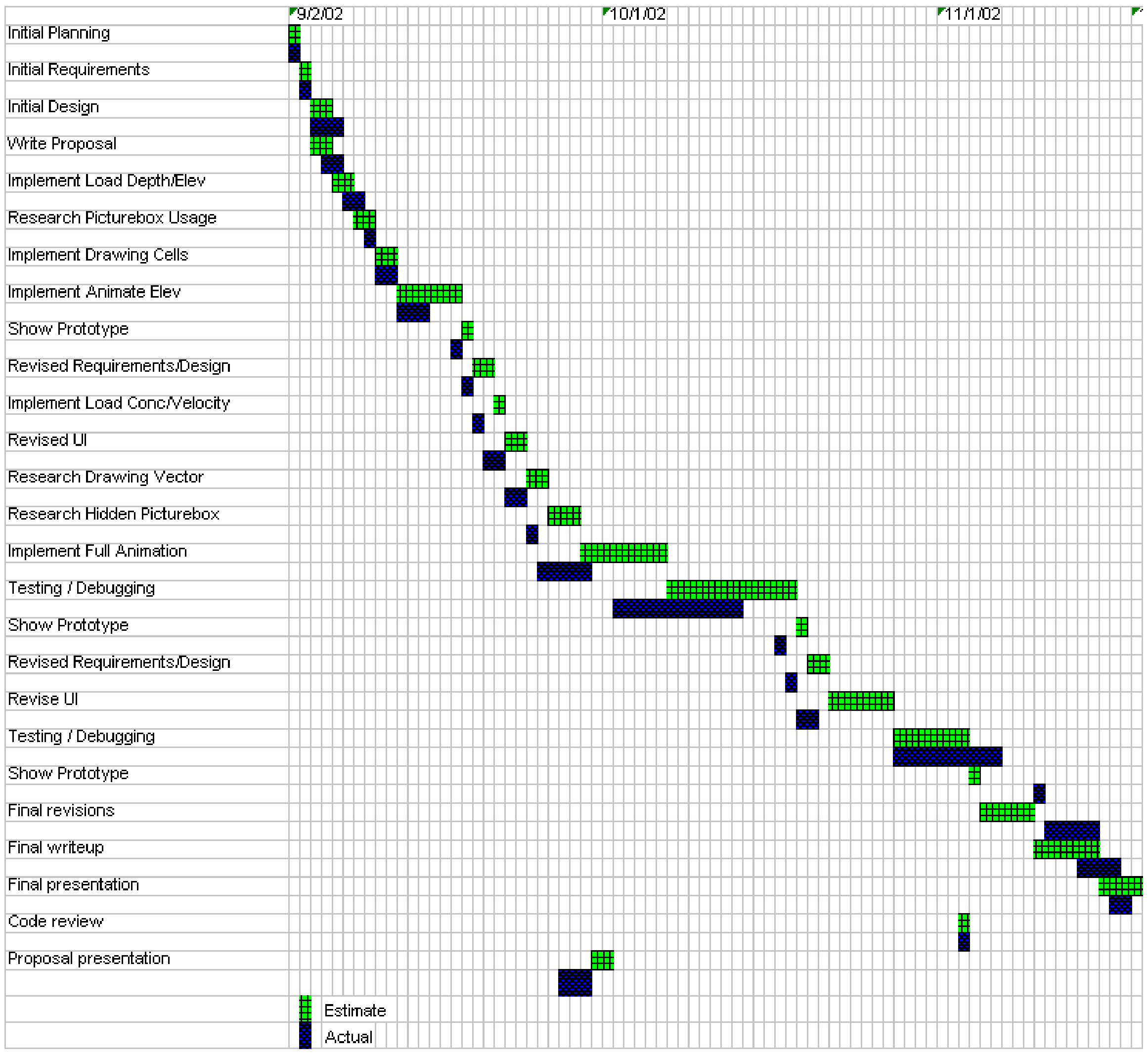


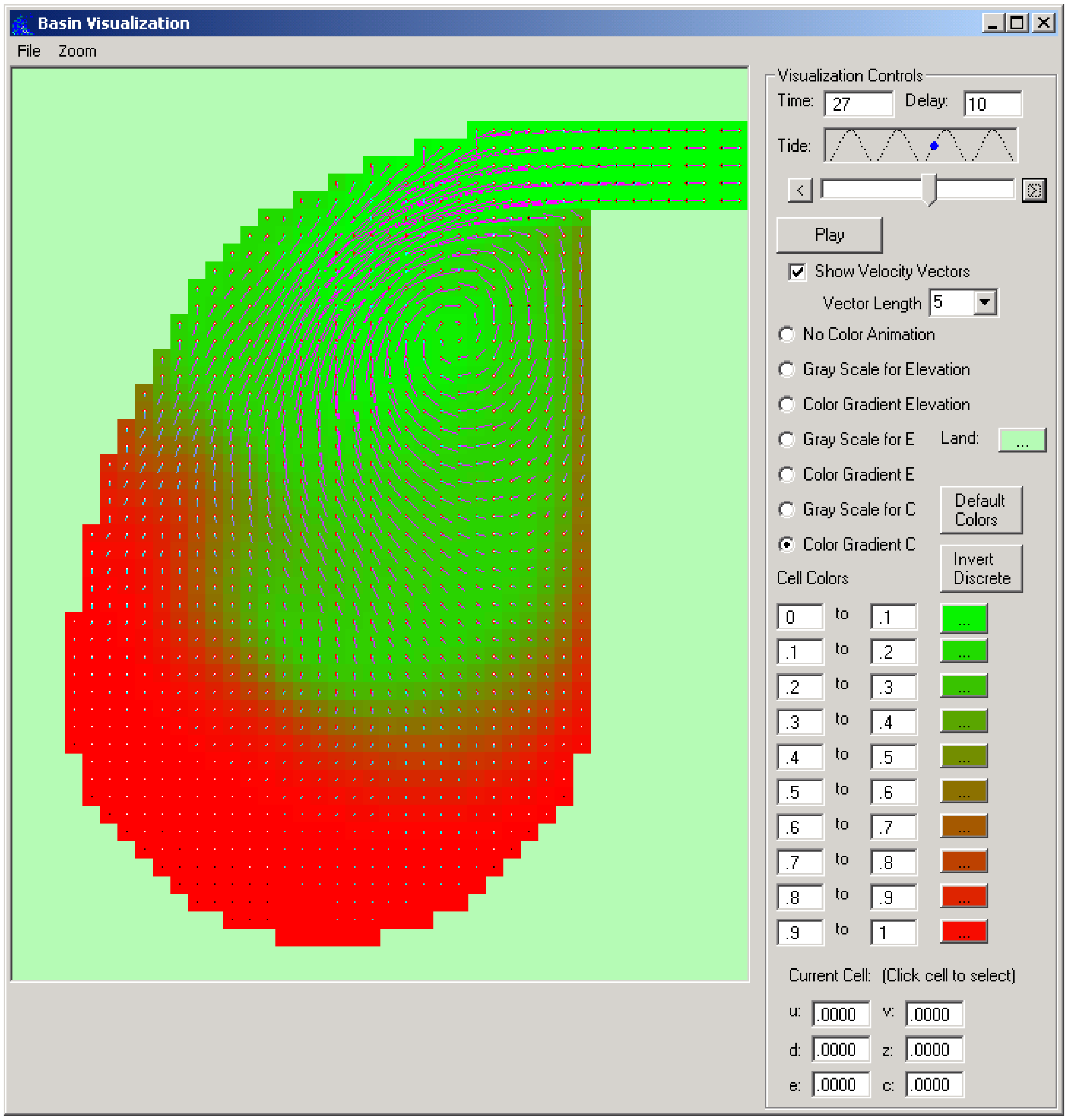
Figure 6. Projected timeline vs. Actual timeline

6. Results

The BasinViz application was completed on time and was used successfully to analyze several different basins. While the program was used initially to visualize basins for analytical purposes, the program has been used more recently as an informational tool in presentations to funding agencies and clients. Overall, the company has been very pleased at the new ability to easily visualize the flow and concentration values of arbitrary basins. All of the requested requirements were met to the satisfaction of the company in the final deliverable.

6.1 Final Program

Several screenshots of the final program are shown below.   For a thorough description of the interface elements, refer to the user manual in Appendix A. Figure 7 depicts the application in color gradient mode for the concentration while also depicting vectors during the midpoint between tides. In gradient mode, a color gradient is set up with green for small values (0) transitioning to red for large values (1.0). If the play button is selected, then all cell values will be updated and animated over time.



Time: 27 Delay: 10

Tide:

[✓] Show Velocity Vectors

Vector Length 5

( ) No Color Animation
( ) Gray Scale for Elevation
( ) Color Gradient Elevation
( ) Gray Scale for E
( ) Color Gradient E
( ) Gray Scale for C
[✓ Color Gradient C]

Land: ...

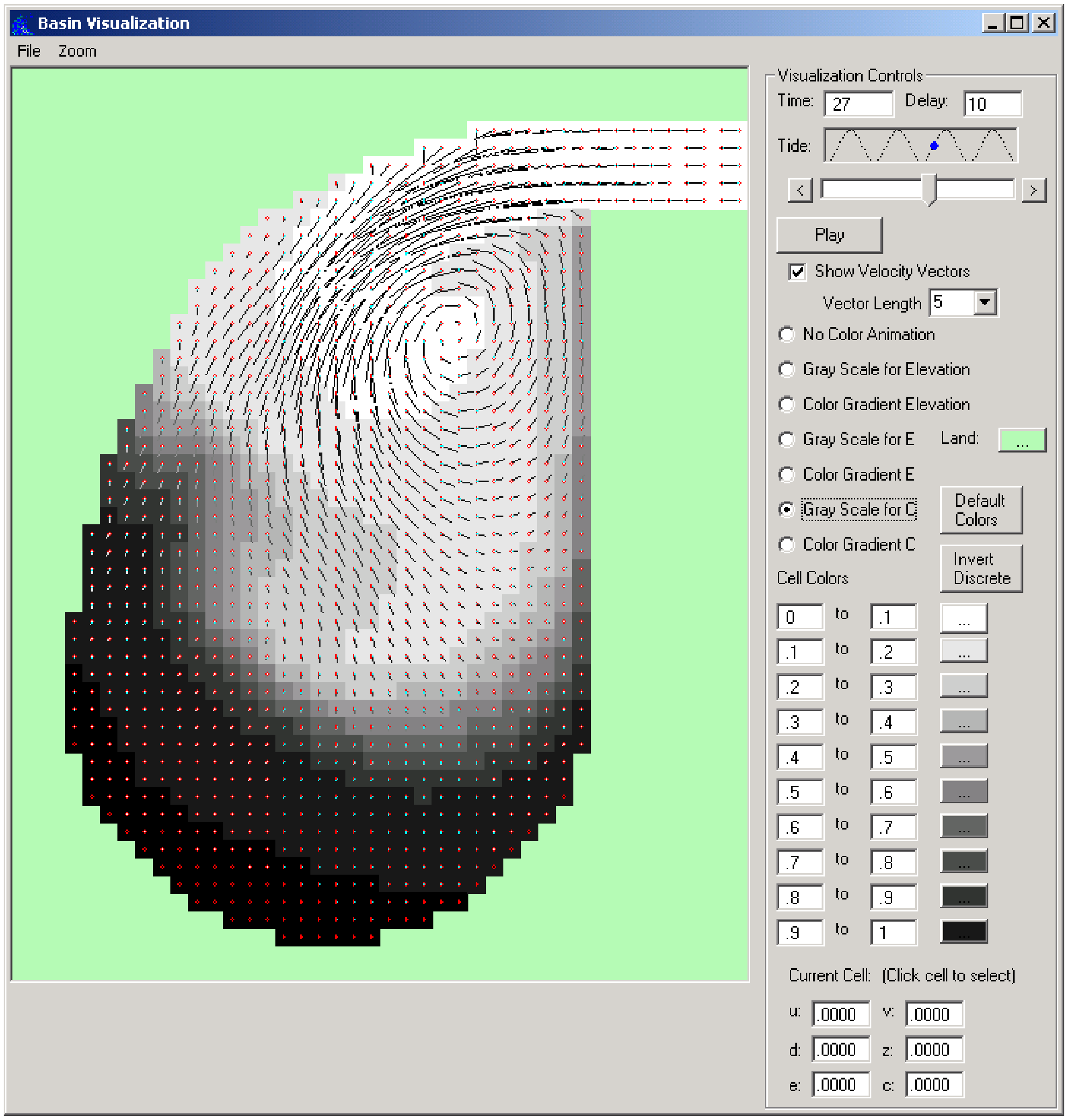
Cell Colors

0	to	.1	🟩
.1	to	.2	🟩
.2	to	.3	🟩
.3	to	.4	🟩
.4	to	.5	🟫
.5	to	.6	🟫
.6	to	.7	🟧
.7	to	.8	🟥
.8	to	.9	🟥
.9	to	1	🟥

u:	.0000	v:	.0000
d:	.0000	z:	.0000
e:	.0000	c:	.0000

Figure 7. Gradient Mode for concentration with vectors

Figure 8 depicts the same data as figure 7, but in gray scale mode instead of color gradient mode. In gray scale mode, ten discrete colors are set up for the data ranges shown in the text boxes. The user has the ability to change the color for each data range, but in practice the company left the gray scale range as-is. This was the most popular mode used by the engineers.



Time: 27 Delay: 10

[✓] Show Velocity Vectors

Vector Length 5

( ) No Color Animation
( ) Gray Scale for Elevation
( ) Color Gradient Elevation
( ) Gray Scale for E
( ) Color Gradient E
(●) Gray Scale for C
( ) Color Gradient C

Cell Colors

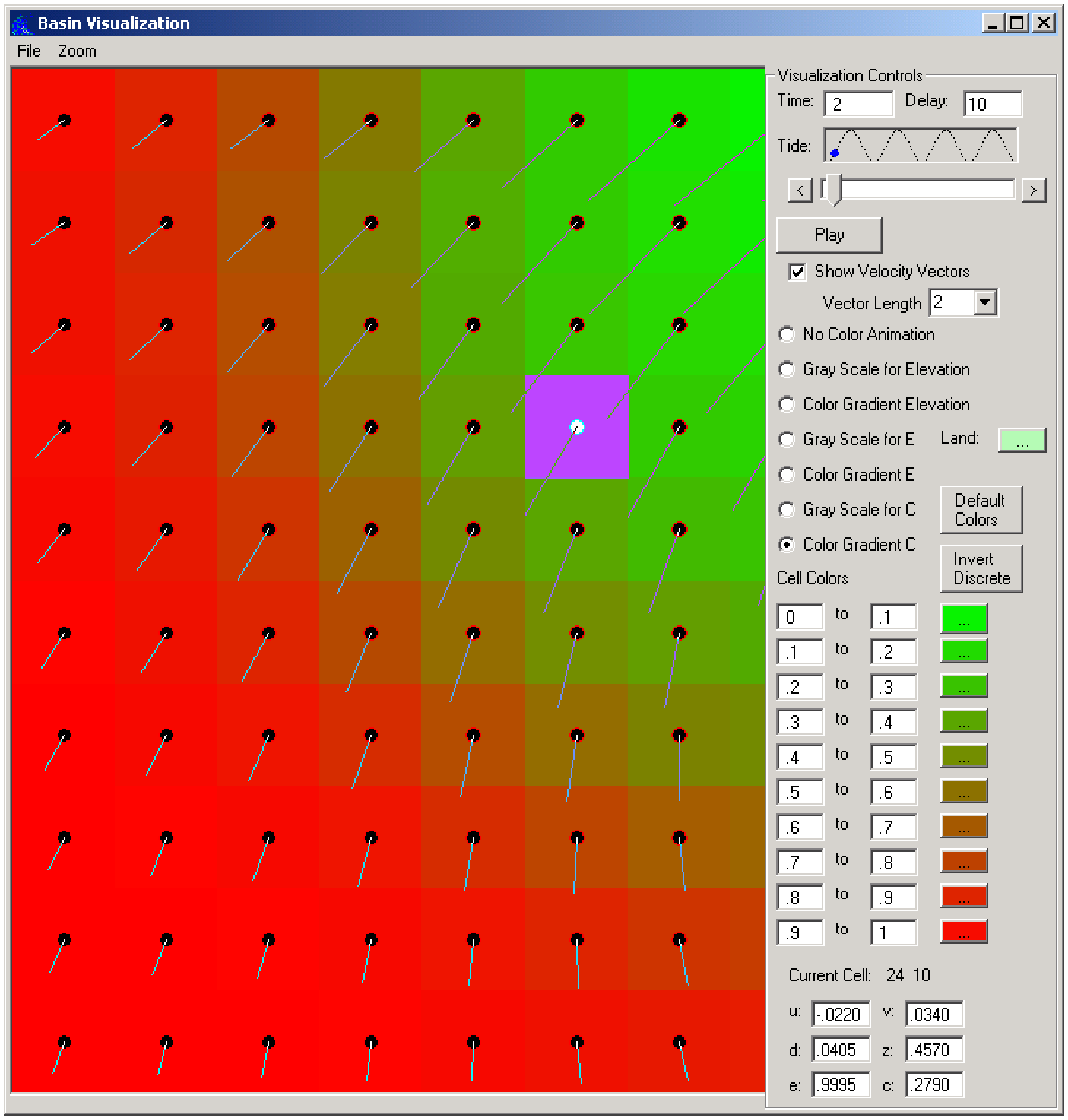
0	to	.1	...
.1	to	.2	...
.2	to	.3	...
.3	to	.4	...
.4	to	.5	...
.5	to	.6	...
.6	to	.7	...
.7	to	.8	...
.8	to	.9	...
.9	to	1	...

Current Cell: (Click cell to select)

u:	.0000	v:	.0000
d:	.0000	z:	.0000
e:	.0000	c:	.0000

Figure 8. Gray Scale Mode for concentration with vectors

Finally, Figure 9 shows the application in zooming mode. In this case, we are in color gradient mode for C and have zoomed in near the upper right corner of the basin. The cell that is purple has been highlighted, so its data values are displayed in the text boxes in the lower right hand corner of the interface. By zooming in, we can get a much clearer indication of the direction of the vectors and the values in each cell.



Time: 2 Delay: 10

Tide:

[✓] Show Velocity Vectors

Vector Length 2

( ) No Color Animation
( ) Gray Scale for Elevation
( ) Color Gradient Elevation
( ) Gray Scale for E Land: ...
( ) Color Gradient E
( ) Gray Scale for C
(•) Color Gradient C

Cell Colors

0	to	.1	🟩
.1	to	.2	🟩
.2	to	.3	🟩
.3	to	.4	🟩
.4	to	.5	🟩
.5	to	.6	🟫
.6	to	.7	🟫
.7	to	.8	🟥
.8	to	.9	🟥
.9	to	1	🟥

Current Cell: 24 10

u:	-.0220	v:	.0340
d:	.0405	z:	.4570
e:	.9995	c:	.2790

Figure 9. Zoom-in 10x10 area, Gradient Mode for concentration

6.2 Future Steps

The success of the application has prompted a request for future development. The first request is a user-friendly graphical front end that selects the appropriate data files for the basin of interest for situations where there are dozens of basins to choose from. The second request is to integrate the model that generates the data directly into the BasinViz program. Currently the data is generated from a separate Fortran program. By integrating the model into BasinViz, the Fortran program can be eliminated and the data file step could also be eliminated. I intend to work on these enhancements sometime early next year.

7. Summary and Conclusions

BasinViz was developed in Visual Basic 6.0 for Coastal Engineering with the goal of visualizing characteristics of tidal basins. The project was completed on time. Through the use of prototyping, the client was able to describe new features and find bugs fairly early in the development process. The final product met the requirements for the company and they have expressed a great deal of satisfaction in the product. Moreover, the company would like to expand the features of the program in the future.

Overall, I found this to be a challenging but exciting project since I had to learn how to use a myriad of controls and techniques in Visual Basic. While I had some prior experience in Visual Basic, I had limited experience drawing graphics and using the picturebox control. Additionally, I gained the experience of developing software for a corporate client. By having a real product that met real needs, scenarios and bugs had to be discovered and addressed that might otherwise be swept under the rug if this software were created purely as an academic exercise.

One item that I would change in the future would be the specification of more formal requirements. A number of bugs and new features could have been avoided if the requirements were specified more carefully in the beginning of the project. For example, the bug regarding the location of the origin in drawing the vectors could have been eliminated with more detailed specifications using tools such as the Unified Modeling Language.

8. References

[1] Balena, Francisco. (1999). Programming Visual Basic 6.0. Microsoft Press, Seattle WA.

[2] Stephens, Rod. (1999). Visual Basic Graphics Programming: Hands-On Applications and Advanced Color Development. 2nd Edition, John Wiley & Sons, New York, NY.

Appendix A: User Manual

Minimum System Requirements

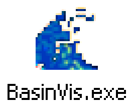
Windows XP, Vista, or 7
512 Mb of memory
1 Ghz CPU
Video resolution of 1024 x 768

Installation

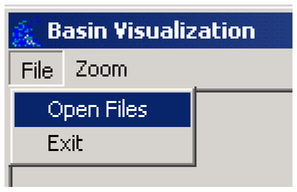
Insert the CD and double-click on “BasinViz-Inst.exe”. This will launch the installer which will prompt you for a location on your hard drive to install the software. Select the desired folder and the software will be copied to your hard drive.

Starting the Program

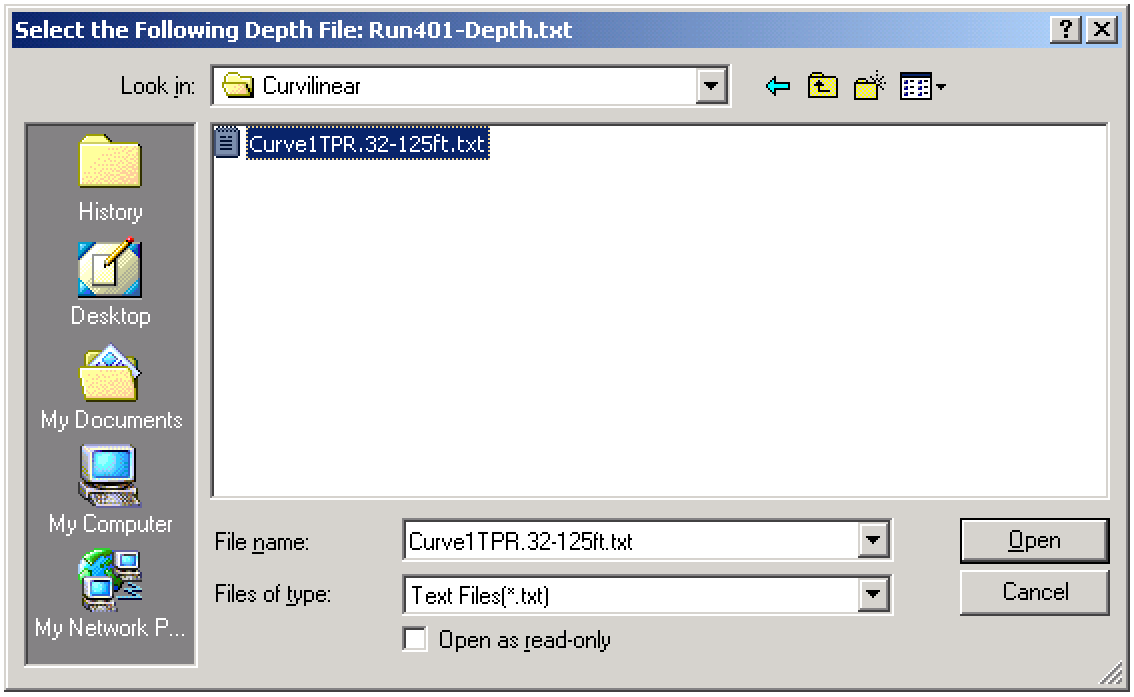
Run the program by selecting “BasinViz” from the start menu or by double-clicking on the BasinViz icon:



The program will launch and bring up an empty screen. Select F)ile and O)pen to select data files to load.



Navigate to the **depth** file of the basin you would like to read. The basin files must be in the POM file format, with a file for depth, U velocity, V velocity, elevation, and concentration. In the example below, the depth file has been selected for the Curvilinear basin:


Select the Following Depth File: Run401-Depth.txt

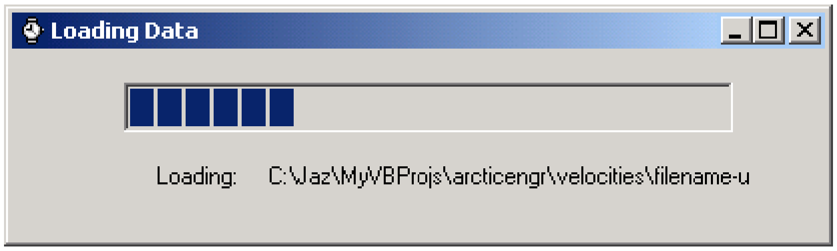
Look in: Curvilinear

Curve1TPR.32-125ft.txt

File name: Curve1TPR.32-125ft.txt

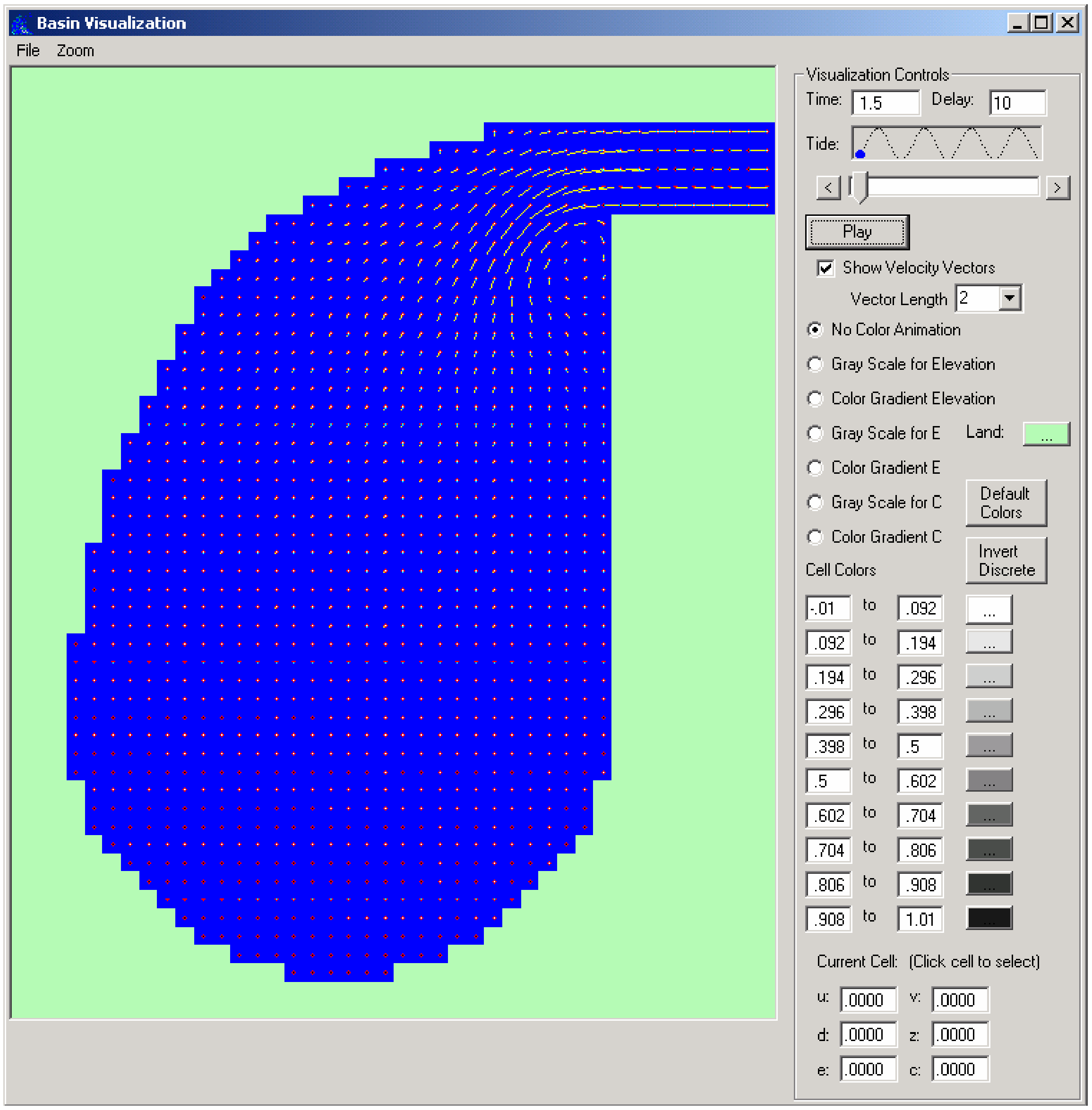
Files of type: Text Files(*.txt)

Select "Open" and the file will be loaded. The program will then automatically load the U, V, elevation, and concentration files while displaying a progress bar:


Loading Data

Loading: C:\Jaz\MyVBProjs\arcticengr\velocities\filename-u

After the files have completed loaded, the basin will be displayed in the main window. The default selections are shown on the right.


Basin Visualization

File Zoom

Visualization Controls

Time: 1.5 Delay: 10

Tide:

Play

[✓] Show Velocity Vectors

Vector Length 2

[● No Color Animation]

○ Gray Scale for Elevation

○ Color Gradient Elevation

○ Gray Scale for E Land:

○ Color Gradient E

○ Gray Scale for C

○ Color Gradient C

Default Colors

Invert Discrete

Cell Colors

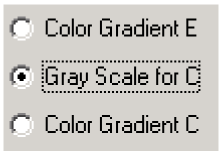
-.01	to	.092	...
.092	to	.194	...
.194	to	.296	...
.296	to	.398	...
.398	to	.5	...
.5	to	.602	...
.602	to	.704	...
.704	to	.806	...
.806	to	.908	...
.908	to	1.01	...

Current Cell: (Click cell to select)

u:	.0000	v:	.0000
d:	.0000	z:	.0000
e:	.0000	c:	.0000

Selecting Visualization Options

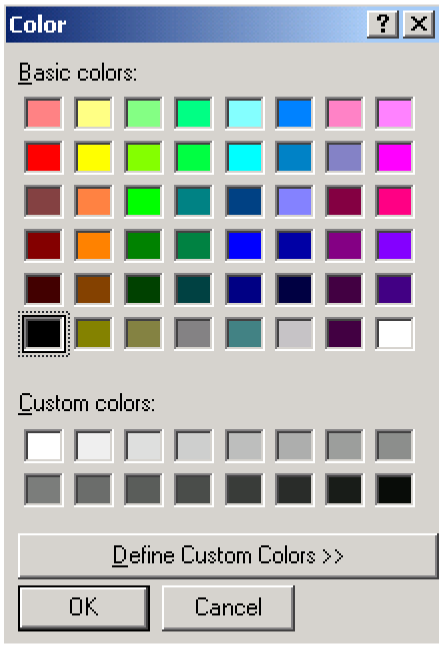
Using the radio buttons on the right side, select the feature you would like to color-coded in the basin. You may animate the Elevation, E, or Concentration. Either feature may be displayed using a color gradient or ten discrete color values. In the picture below, the Gray Scale for Concentration option has been selected.


○ Color Gradient E

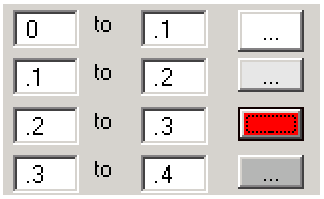
[● Gray Scale for C]

○ Color Gradient C

If one of the gray scale modes is selected then you also have the ability to modify the ten discrete color ranges. This is accomplished by entering the values you wish in the textboxes under the “Cell Colors” section. By clicking on the “…” button you can change the color for that data range. This will bring up the color chooser shown below:



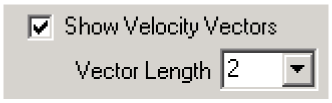
In the example below, we have set the color to red for any concentration values between 0.2 and 0.3.



0	to	.1	...
.1	to	.2	...
.2	to	.3	🟥
.3	to	.4	...

If you have changed several colors and would like to reset them to the default gray scale, click the “Default Colors” button. Optionally you may also click the “Invert Colors” button to flip the color ranges. This will swap the color used for the top range with the color used for the bottom ranges all the way down for each color.

Next, select whether or not you would like to view velocity vectors. By default, the box is checked to show the vectors. Uncheck the box if you do not wish to see the vectors. You may also change the length of the vectors by changing the value in the pull-down menu.

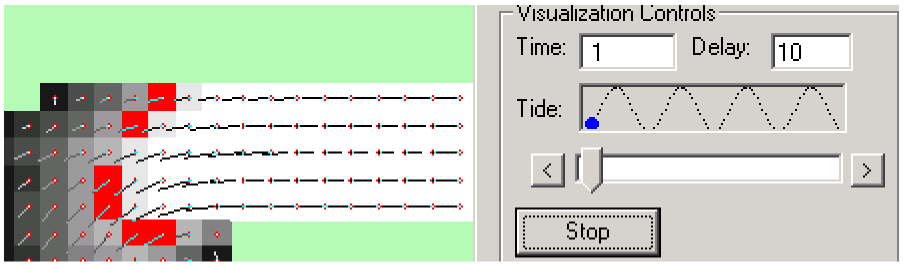


[✓] Show Velocity Vectors
Vector Length 2

Playing the Animation

After the visualization options have been set the program is ready to play the animation. Click the “Play” button and the display will be updated with the data values for each time step. The animation will loop when it reaches the end of the slider Click the “Stop” button to stop the animation.

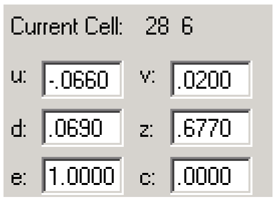
In the screen shot below, we are currently at time period 1. If the animation is running too quickly, increase the value in the “Delay” textbox. The Blue circle is correlated to the slider and the Tide shows the elevation of the water at the current time value. Note that in this example, some of the cells are red since we previously selected the color red for concentration between 0.2 and 0.3.


Visualization Controls

Time: 1 Delay: 10

If the animation is stopped, you may click on the “>” and “<” buttons to move one frame backward in time or one frame forward in time. You may also click and drag the slider to the desired time point. While the animation is running, you may also change any other values you have selected. For example, while running you can click the radio button to display the Concentration using the color gradient, and the display will be updated accordingly.

If you would like to see data values for a specific cell, then click that cell in the display. The data values for that cell at the selected time will be updated in the lower right corner of the window.

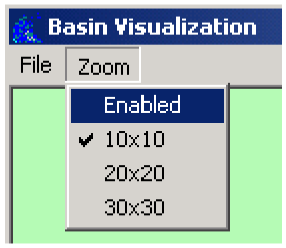

Current Cell: 28 6

u:	-.0660	v:	.0200
d:	.0690	z:	.6770
e:	1.0000	c:	.0000

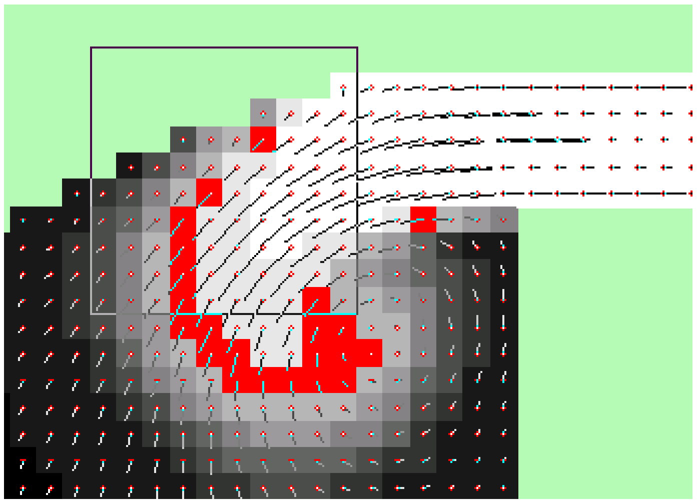
Zooming

To view an area of interest in more detail, the program supports a zoom option. To use it, make sure that the animation has stopped. Then select Z)oom from the menu and pick the size of the zoom area. 10x10 selects a 10 by 10 area of cells and is the largest magnification factor. Other options include 20x20 and 30x30, which is the smallest magnification factor. These options may be disabled if there are fewer than 30 cells.

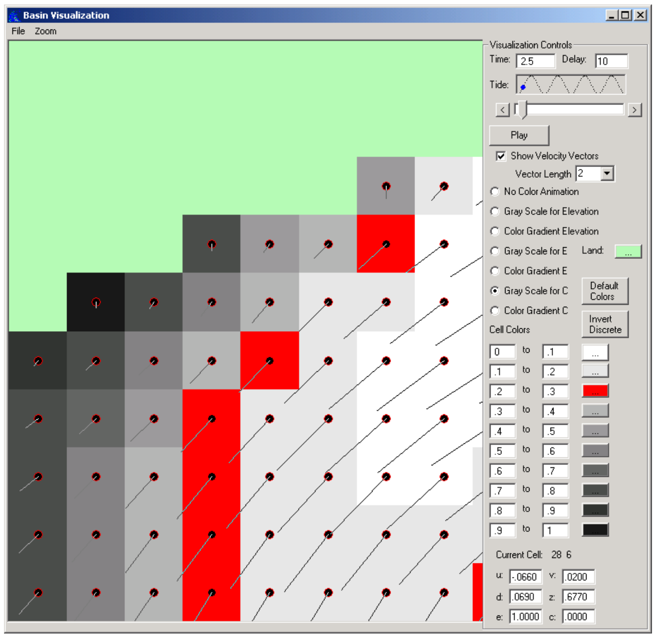
To enable zooming, return to the Zoom menu and select “Enabled”.



This will bring up a box that you can drag over the screen to select the area you would like to zoom into:



Click the mouse to select that region for zooming. The display will be updated to the area you have selected.



Visualization Controls

Time: 2.5 Delay: 10

Tide:

[✓] Show Velocity Vectors

Vector Length 2

No Color Animation

Gray Scale for Elevation

Color Gradient Elevation

Gray Scale for E Land: 🟩

Color Gradient E

[✓ Gray Scale for C]

Color Gradient C

Cell Colors

0	to	.1	⬜
.1	to	.2	⬜
.2	to	.3	🟥
.3	to	.4	⬜
.4	to	.5	⬜
.5	to	.6	⬜
.6	to	.7	⬛
.7	to	.8	⬛
.8	to	.9	⬛
.9	to	1	⬛

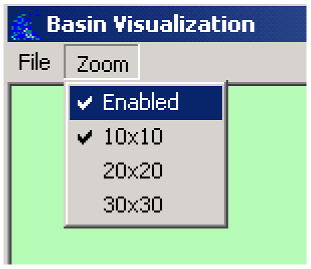
Current Cell: 28 6

u: -.0660 v: .0200

d: .0690 z: .6770

e: 1.0000 c: .0000

To disable zooming, return to the Z)oom menu and click “Enabled” again.



The checkmark will be removed and Zooming will be turned off, restoring the original view of the entire basin.

Screen Shots

To get a screen shot of the program, hold Alt-PrintScreen. This will copy the program window to the clipboard. You may then paste the image into a paint program such as Paint or Adobe Photoshop and edit/print the image.

Exiting

When you would like to exit the application, choose F)ile, E)xit or click the X in the upper right corner.

Appendix B: Code Listing

Include a listing of your code here (or attach as a compressed zip/tar file when submitting).