

University of Montana

ScholarWorks at University of Montana

Graduate Student Theses, Dissertations, &
Professional Papers

Graduate School

1998

Computer visualization of petroleum reservoir characterization data

Jeffrey Alan Braun
The University of Montana

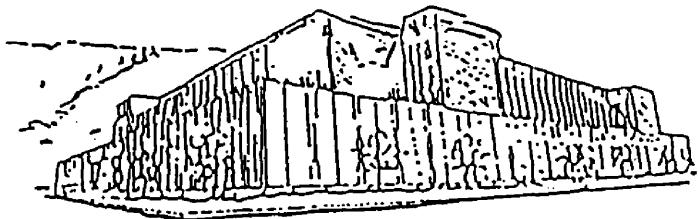
Follow this and additional works at: <https://scholarworks.umt.edu/etd>

Let us know how access to this document benefits you.

Recommended Citation

Braun, Jeffrey Alan, "Computer visualization of petroleum reservoir characterization data" (1998).
Graduate Student Theses, Dissertations, & Professional Papers. 8253.
<https://scholarworks.umt.edu/etd/8253>

This Thesis is brought to you for free and open access by the Graduate School at ScholarWorks at University of Montana. It has been accepted for inclusion in Graduate Student Theses, Dissertations, & Professional Papers by an authorized administrator of ScholarWorks at University of Montana. For more information, please contact scholarworks@mso.umt.edu.



Maureen and Mike
MANSFIELD LIBRARY

The University of **MONTANA**

Permission is granted by the author to reproduce this material in its entirety,
provided that this material is used for scholarly purposes and is properly cited in
published works and reports.

*** Please check "Yes" or "No" and provide signature ***

Yes, I grant permission

No, I do not grant permission

X

Author's Signature

Jeffrey A Braum

Date

3-10-98

Any copying for commercial purposes or financial gain may be undertaken only with
the author's explicit consent.

Computer Visualization of Petroleum Reservoir Characterization Data

by

Jeffrey Alan Braun

Submitted in partial fulfillment
of the requirements for the degree of

Master of Science

in

Computer Science

The University of Montana

May 1998

Approved by:



Chairperson



Dean, Graduate School

3-11-98

Date

UMI Number: EP39054

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI EP39054

Published by ProQuest LLC (2013). Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.

All rights reserved. This work is protected against unauthorized copying under Title 17, United States Code



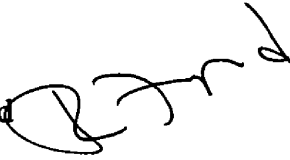
ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

Braun, Jeffrey A., M.S., May, 1998

Computer Science

Computer Visualization of Petroleum Reservoir Characterization Data

Director: Dr. Ray Ford



A multi-step process is presented to transform a variety of types of petroleum data into visual displays. In the visualization process, the scientist must collect the data, identify the visualization goal, import the data into a system with visualization capability (or implement the visualization capability), design the visualization analysis, and determine the final output and its requirements. This project involves the application of the multi-part visualization process to various petroleum data collected for the NE Rabbit Hills oil field in connection with a petroleum reservoir characterization study. Artifacts produced from the visualization process include 3-D renderings of the subsurface, renderings of reservoir model parameters, and various oil production animations. Target audiences for these artifacts range from petroleum scientists and engineers to the general public.

Table of Contents

Abstract	ii
Chapter 1: Introduction	1
1.1 Data Visualization in the Petroleum Industry	2
1.2 NE Rabbit Hills Reservoir Characterization Study	4
Chapter 2: Five Step Visualization Process	6
2.1 Gather, Collect, or Create Data	6
2.2 Formulate a Vision	8
2.3 Prepare and Import Data into Data Explorer	9
2.4 Design the Visualization Analysis	11
2.5 Determine Final Visualization Output Requirements	15
Chapter 3: Visualization of Subsurface Geologic Formations	19
3.1 Geologic Formation Top Data	19
3.2 Vision for the Geologic Subsurface Visualization	20
3.3 Preparing and Importing Formation Top Data into Data Explorer	21
3.4 Visualization Analysis	22
3.5 Final Visualization Output Requirements	27
Chapter 4: Visualization of Oil Production Data	29
4.1 Monthly Oil Production Data	29
4.2 Vision for the Oil Production Animation	29
4.3 Preparing and Importing the Oil Production Data into Data Explorer	31
4.4 Oil Production Visual Analysis	32
4.5 Data Explorer Animation Output	36
Chapter 5: Visualization of Reservoir Model Parameters	38
5.1 Reservoir Simulation Data	38
5.2 Visualization Goals	39
5.3 Preparing and Importing Reservoir Model Parameters	41
5.4 Visualization Analysis	42
5.5 Visualization Output Requirements for Web Site Imagery	46

Chapter 6: Discussion And Conclusions	48
6.1 Visualization Process Revisited	48
6.2 Conclusions and Directions for Future Research	50
Appendix A: Conversion Programs	53
Bibliography	60

List of Figures

2-1 DX Data Model Objects	9
2-2 DX Data Prompter Window	10
2-3 DX Program Network Example	12
2-4 Surface Elevation Display	13
2-5 SurfacePlot Macro	14
3-1 NE Rabbit Hills Geologic Formation Tops Display	20
3-2 Geologic Formation Top Visual Program	23
3-3 Visual Program for Plotting Wells	24
3-4 Flag Module Dialog Box	26
3-5 Subsurface Control Panel	28
4-1 Amoco #1 Monthly Oil Production Rates	30
4-2 Spreadsheet Example of Monthly Production Data	32
4-3 Oil Production Visual Program	34
4-4 1990 Cumulative Oil Production Display	35
4-5 1996 Cumulative Oil Production Display	36
5-1 NE Rabbit Hills Reservoir Thickness	40
5-2 NE Rabbit Hills Reservoir Structure and Thickness	42
5-3 NE Rabbit Hills Reservoir Structure and Thickness	43
5-4 NE Rabbit Hills Reservoir Permeability	43
5-5 Simulated Reservoir Pressure	44
5-6 Simulated Reservoir Oil Saturation	45

Chapter 1: Introduction

Scientific Data Visualization is a process of using a collection of software tools and techniques to generate computer-based displays that allow observers to graphically explore, analyze, and understand data from scientific applications (Earnshaw and Wiseman, 1992; Brodlie et al., 1992). Scientists use visualization to gain insight into data sets that are so large, complex, or diverse that analysis is not possible with conventional means. Visualization is especially useful for displaying and analyzing large volumes of multidimensional and time varying data. Visualization is also very important in computer modeling, where much of the data is generated on-line, since the modeling results can be readily imported into a visualization system.

Generally, a scientist uses data visualization to present and communicate scientific results to a specific target audience. The target audience can vary from the scientist himself/herself, to peer scientists, to the general public. Display techniques designed to communicate to application experts are said to be focused on **modeler support**. In contrast, Keller and Keller (1993) and Earnshaw and Wiseman (1992) distinguish visualizations designed to communicate to a broader, less knowledgeable audience as **presentation graphics**. Visualizations designed for presentations require much more contextual information, annotation, and other special effects to orient audiences unfamiliar with the data and/or processes being depicted.

Visualization experts generally describe the creation of a visualization as a multi-part process, no matter what the target audience. The scientist must collect the data, identify the visualization goal, import the data into a system with visualization capability (or implement the visualization capability), design the visualization analysis, and determine the final output and its requirements. Details of the visualization process are covered in Chapter 2. This project involves the application of the multi-part visualization process to a variety of types of petroleum data collected for the NE Rabbit Hills oil field in connection with a petroleum reservoir characterization study. Artifacts produced from

the visualization process include 3-D renderings of the subsurface, renderings of reservoir model parameters, and various oil production animations. Target audiences for these artifacts range from petroleum scientists and engineers to the general public.

1.1 Data Visualization in the Petroleum Industry

Slatt et al. (1996) discuss the use of visualization technology in the oil and gas industry, focusing primarily on uses for characterizing and understanding surface and subsurface phenomena. Like other scientists, petroleum specialists use visualization tools to view and understand large quantities of data, to help check data consistency, and to integrate data from the different petroleum disciplines. Integration of geological, geophysical, and petroleum engineering data has grown in importance as the petroleum industry shifts towards cross-disciplinary teams for hydrocarbon exploration and reservoir development. Visualization of concepts and information provides a means for overcoming communication problems between scientists with different technical backgrounds.

In this interdisciplinary context, characterizing petroleum reservoirs involves the integration and analysis of geologic, seismic, petrophysical, and production data. A primary goal of reservoir characterization is to produce a 3-D reservoir model that satisfies all these data. Tinker (1996), Ahmed et al. (1997), and Olson et al. (1997) present different methods to integrate diverse data types. Ultimately, a reservoir simulator is used to integrate the data into a computational model that attempts to match various measurements, estimates, predictions, and actual production records. Reservoir simulations thus provide the primary means through which a team of scientists can depict both the structure and the fluid movement within the structure that represents a petroleum reservoir. Simulators typically produce more data as output, including one or more time series of reservoir model parameters. The reservoir simulation output, with simulator input parameters and sequences of state values providing contextual information, are well

suited for the use of visualization techniques that reduce the data load by producing displays, instead of columns of numbers, for the modeler or other audiences.

Existing software designed especially for petroleum production applications supplies some visualization tools. For example, the Eclipse (Schlumberger, 1997) reservoir simulation software provides basic graphical support to allow a modeler to depict model parameters at individual time steps. However, Eclipse does not provide any means of animating the time series data or adding contextual data to the images. The Landmark Seisworks (Landmark, 1997) and GeoQuest IES (GQS, 1989) seismic interpretation software packages are similar in function, in that they display 2-D and 3-D seismic data that geophysicists use to interpret faults and geologic formations, and provide some 3-D visualization tools to display interpreted horizons and faults. However, these visualization facilities are designed primarily to assist the seismic interpreter in understanding the seismic data and their results, and do not allow the easy production of displays that integrate additional contextual information. Thus, although specific tools generally provide good modeler support display capabilities, the tools lack the ability to integrate results across modeling domains.

General visualization software can be used to supplement existing petroleum software, particularly by integrating information from various disciplines. General visualization packages provide graphical and image production support that the specialized petroleum software may not provide. For example, the IBM Data Explorer (DX) visualization package is used in this paper to demonstrate the visualization process. Data Explorer's power and strength lie in its general purpose design. DX provides a flexible means of describing and importing various data types into the visualization system and its data model. The DX data model supports data defined on scattered sample points, regular grids, deformed grids, and irregular grids. Data Explorer can handle data of any type (i.e., real, complex, scalar, vector) defined over spaces of any dimensionality. Data Explorer provides general purpose visualization tools that can be used in multiple ways to transform data into graphic images. The visual artifacts produced by Data Explorer can

be output in a variety of modes, including postscript files, compressed image files, animated image sequences, and VRML objects.

1.2 NE Rabbit Hills Reservoir Characterization Study

Three data sets from the NE Rabbit Hills oil field are used to demonstrate the visualization process. The data were collected as part of a reservoir characterization study being conducted by the Montana University System Petroleum Reservoir Characterization research team. The team consists of geologists, geophysicists, computer scientists, and petroleum engineers located at the University of Montana, Montana Tech, and the Montana Bureau of Mines and Geology. The main focus of the research is to develop a method to integrate the various data sets into a complete 3-D reservoir model. The reservoir model is used to validate various measurements, predictions, and analyses, then predict future reservoir performance in connection with different proposed enhanced oil recovery scenarios.

The NE Rabbit Hills oil field is located in north-central Montana, about eight miles north of Chinook, Montana. The oil field was discovered in 1972 with the Amoco #1 USA Erving Wolf well. The oil reservoir occurs in the upper part of the Bowes Member of the Sawtooth Formation. This well went on production in 1973 and continues to produce oil in 1997. Two additional wells were drilled into the reservoir in the 1975 and 1984. In 1991 and 1992, most of the field's producing wells were drilled. A total of 13 productive wells and 9 dry holes have been drilled in the NE Rabbit Hills study area. In 1994, the reservoir characterization researchers began the NE Rabbit Hills field study by gathering existing data and collecting new data from the field area.

The data sets collected include electric well logs, core samples and routine core analyses, 3-D seismic data, crosswell seismic data, and production data. Details for some of these data are discussed in the subsequent chapters. These raw or field data are interpreted

within their respective specialized disciplines, generally producing additional data sets. Interpreted data sets include geologic formation tops, porosity and permeability distributions, and seismic time horizons. Various visualization tools are used to help the scientists understand these data and produce images to communicate to other team members and the general public. The visualization process used to transform several of the data sets into visual displays is presented next in Chapter 2. Chapters 3, 4, and 5 discuss the visualization process for three specific data sets: geologic formation tops, production data, and reservoir simulation results. Additional visualizations, such as for processed 3-D seismic data and electric well log data, have been produced for the reservoir characterization study and are available on the project Web site (<http://www.cs.umt.edu/DOE>), but are not presented in this paper.

Chapter 2: Five Step Visualization Process

Transforming data into visual images is a multi-step process (Brodie et al., 1992; Keller and Keller, 1993; Haber and McNabb, 1990; IBM, 1997). The visualization process normally starts when the scientist gathers or creates the data to be analyzed. Once the scientist understands how the data are organized, he/she begins to envision or collect ideas on what the visualization should show. Next, the scientist needs to import the data into the Data Explorer visualization system, which typically requires converting the data from its original format to the DX data model. Once the data are imported into the system, the scientist performs the visual analysis using a variety of tools and functions provided by Data Explorer. After the scientist completes one or more computer visualizations, he/she needs to determine what the final output is going to be, such as realtime computer presentations, static images, or imagery to be integrated into videos.

2.1 Gather, Collect or Create Data

The visualization process normally starts with gathering, collecting or creating data, which can come from various sources and in a variety of forms. This is especially true in the petroleum industry where geoscientists and engineers use data in many different forms, from paper copies to digital binary data. These data can be organized as point data scattered around a site (e.g., well data) or gridded data (e.g., 3-D seismic and reservoir model parameters). Data can be time independent, from a single point in time, or in a time series (e.g., 4-D seismic or monthly production data).

Some of these data are physical measurements originally collected in the field. Typical examples include electrical well logs, seismic reflection data, reservoir pressures, and oil production data. The field data often need to be processed before they can be analyzed by the scientists. Both raw and processed data sets can be examined with general visualization software like Data Explorer, but often application-specific software has

been designed to aide in particular types of analysis and interpretation, and which offers some specialized visualization capabilities.

Interpreted results, like geologic formation tops, seismic time or depth horizons, and hydrocarbon reservoir parameters, are more suitable for general computer visualization systems since general visualization packages provide graphical and image production support that the specialized software may not provide. For example, application specific software often provides graphical tools that create only simple 2-D displays to aide the scientist in viewing and understanding the results. These displays can be difficult for people from other technical disciplines or with a non-technical background to understand. Another problem with these displays is the inability to add contextual data (surface topography, property boundaries, etc.) to orient the viewer. Keller and Keller (1993) discuss the importance of incorporating contextual information into an image to better communicate to those unfamiliar with the data. General visualization software allows the scientist to transform results and contextual data into presentations, in the form of displays or videos, that communicate results to a broader audience.

Interpreted results can also be integrated by scientists to create models which are used in computer simulation programs. Multidiscipline teams can use scientific visualization to view, discuss and update initial simulation model parameters, as well as view changes in the model's parameters that occur as the simulation program executes. A good example is an oil reservoir simulation program that matches the production history of oil wells and predicts future oil production. Such programs require input parameters from several different petroleum-related disciplines to create an initial model. The simulation program uses the initial model parameters to generate large quantities of intermediate and final model results that can be better understood with visualization analysis.

2.2 Formulate a Vision

Once the data are collected, the scientist needs to understand how they are organized to begin considering what type of visualization to create. Different data lead to different types of visualization. For instance, time series data are ideally suited for an animation which shows data values varying through time. The dimensionality of data is also important for the type of visualization to be constructed. One dimensional data is best suited for graphs and histograms, while 3-dimensional data is better suited for volume rendering or displaying boundary surfaces.

Ideas for the computer visualization, or the “vision”, can come from various sources like scientific journals, Web sites, professional presentations, visualization sample programs, or popular media such as TV or video. Published papers in journals are commonly used for visualization ideas that involve flat imagery. Often the scientist sees a figure that really communicates a concept and wants to show his data in a similar manner. Or the scientist can browse journals for examples on how similar data are shown. Software advertisements in industry magazines can even yield visualization ideas. Another excellent source for visualization examples are Web sites that promote computer visualization software or display research results for similar fields of study. Sample visualization programs included in the Data Explorer software package are a great source of visualization concepts, and have the added benefit of providing the program that shows how the visualizations were created. Scientific television programs that are broadcast on PBS, the Discovery Channel and other stations are other good sources for visualization and animation ideas.

After the scientist has some visualization ideas, he/she identifies a specific visualization goal. Identifying the visualization goal is the key to constructing effective images (Keller and Keller, 1993). The goal is the meaning the scientist hopes to derive from the visualization and/or communicate to others about the data. A goal is necessary to begin selecting which visualization techniques to use.

Once the “vision” or goal has been established, it is necessary to break the visualization down into separate components or visual objects. Each component must be analyzed and programmed individually. Additional data are typically needed to construct some of the objects, particularly data that add general context to specific scientific results. In the end these different visual objects are combined to form the final visualization.

2.3 Prepare and Import Data into Data Explorer

The third step of the visualization process is to import the data into the Data Explorer system and its data model. The DX data model supports various types of simulation and observational data (Data Explorer User’s Guide, 1995a). Data are stored in the form of **objects**, which are data structures that contain the object’s type, along with additional type-dependent information. The three basic object types are Field, Array, and Group (Figure 2-1). Fields are the fundamental objects of the Data Explorer data model. Fields

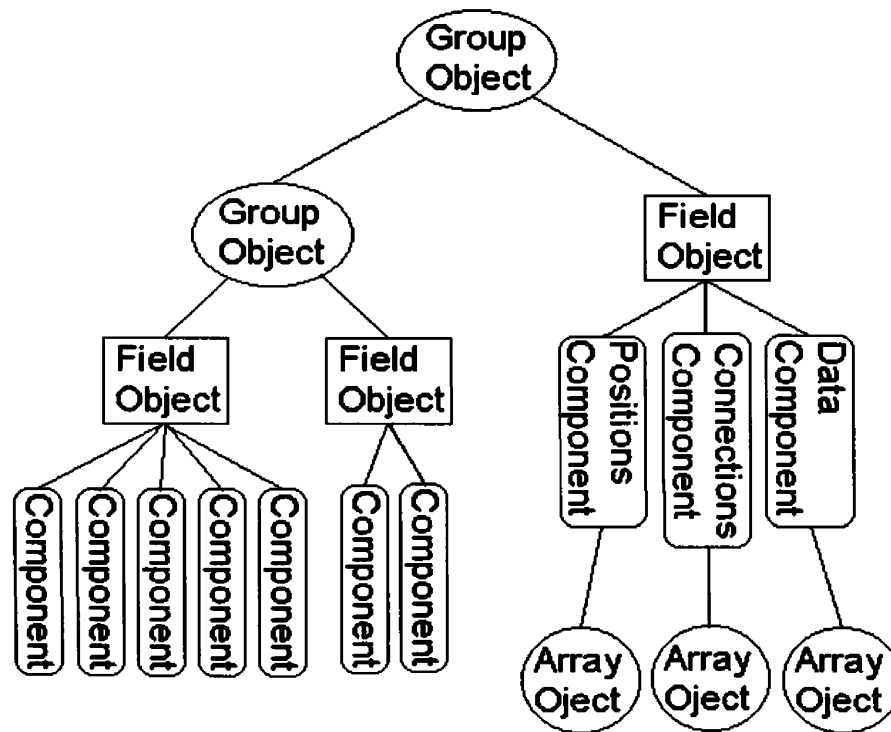


Figure 2-1: DX Data Model Objects.

consist of some number of named components that hold the data values and information about the data. The three standard components are “positions”, “connections”, and “data”. Each component is an object, generally an Array. Arrays are the basic carrying structures that hold the actual information. For example, the “positions” component stores the data positions in an Array object. A Group object is a collection of Fields and other Groups.

The most common way to import data into the DX data model is with the General Array Importer. The general array importer allows the scientist to create a description of the data file, which can be used to guide the actual data import process. The scientist constructs the general array importer description with the DX Data Prompter graphical user interface (Figure 2-2), which allows the user to specify the data filename, data format

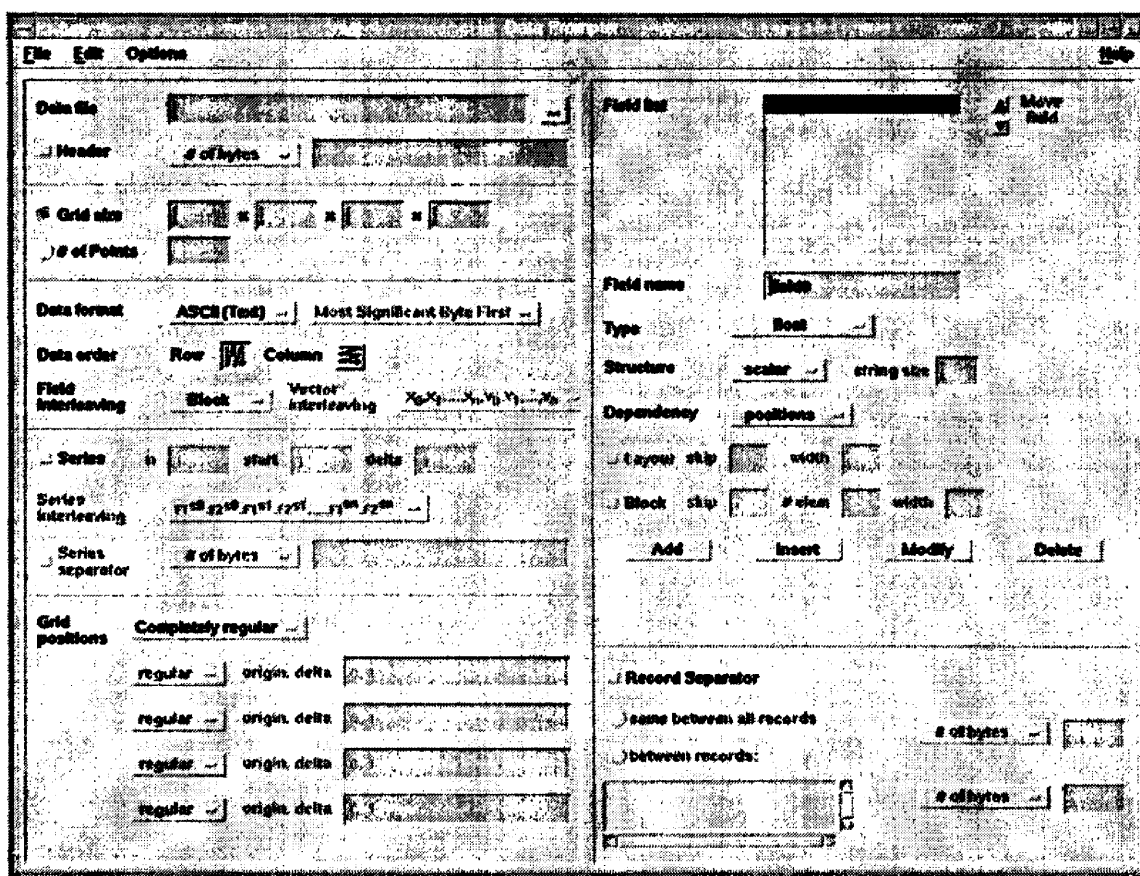


Figure 2-2: Data Prompter user interface used to construct general array importer files.

(grid type, origin, deltas, etc.), time series intervals, data type (scalar, vector, float, integer, etc.) and other miscellaneous information (number of header lines, ASCII or binary, etc.).

Some data file formats are too irregular, complex, or specialized to allow use of the general array importer. For these files a separate conversion program must be written to extract the data of interest and convert it into a format that the scientist can describe with a general array importer file, or convert it directly into a “native” DX data description that can be imported directly. Some conversion programs already exist for commonly used formats, such as Geographical Information Systems (GIS), Digital Elevation Models (DEM), and Digital Line Graphs (DLG) (Moeller, 1997). These data types can provide contextual information in the visualizations, particularly in natural resource applications. These converters allow users to select between a variety of formats, to select data elements from the input files, and output the data in native DX object format, which is an encapsulation of the Data Explorer data model formatted for external storage (Data Explorer User Guide, 1995).

The scientist completes the data import process by using the Import module to read either the General Array Importer file or DX native format file into the DX Visual Programming Editor (VPE) where the visualization analyses are performed. Details on constructing visual programs with the VPE are discussed in the next section.

2.4 Design the Visualization Analysis

In the Data Explorer system, visualization analysis is done by constructing visual programs (Abram and Treinish, 1996). The DX graphical user interface includes a Visual Program Editor (VPE) which allows the user to select, manipulate and transform data into graphic images by interconnecting visualization modules (Figure 2-3). These modules are functions that are represented in the VPE by icons with input and output

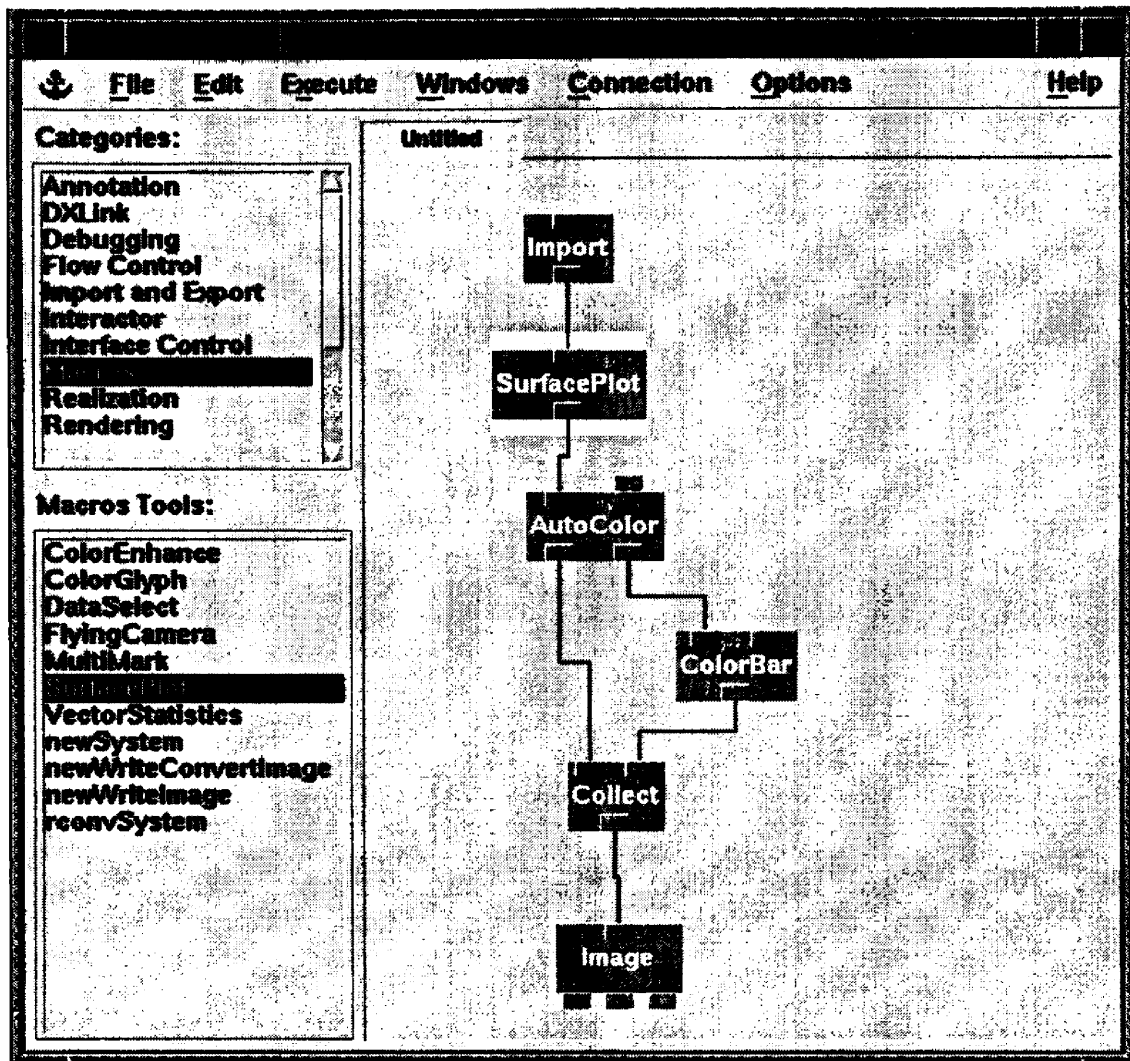


Figure 2-3: A simple Data Explorer program network using DX Modules to convert 2-D elevation data into a colored 3-D surface image.

tabs. The input tabs are analogous to module arguments; the output tabs are module function return values. The user builds a visual program by selecting modules and connecting their inputs and outputs into a network that resembles a flow chart or data flow network. As the data are imported at the top and move through the network, certain modules produce visual artifacts. The flow of execution is traced through modules on the VPE by having them highlighted as they execute. In addition, execution of key modules produces output windows in which the target visualizations are rendered.

In addition to parameter specification via input tab connections, additional module input parameters can be specified in a dialog box specific to each module. Notice in Figure 2-3 that some module icons have unconnected input tabs. If the tab is up, then the module uses a default parameter during the program's execution. If the user double clicks on a module, its dialog box window opens to allow the user to specify the module's input parameters. User specified input parameters have input tabs that are down to indicate that they are defined explicitly in this manner.

Figure 2-3 shows an example visual network program that creates a 3-dimensional colored surface from elevation data with scattered 2-dimensional (x,y) positions. The input into the Import module is a general array importer file that describes the elevation data file. The Import module outputs the elevation data in the form of a Field object with 4-components: data, positions, connections, and box. The Field is input into the SurfacePlot module, which grids the scattered elevation data onto a user specified 2-D grid. The 2-D grid positions are transformed into 3-D positions by incorporating the elevation or z-value at each grid point into its position. The 3-D surface Field object is passed to the AutoColor module, where a color is assigned based on the elevation value at

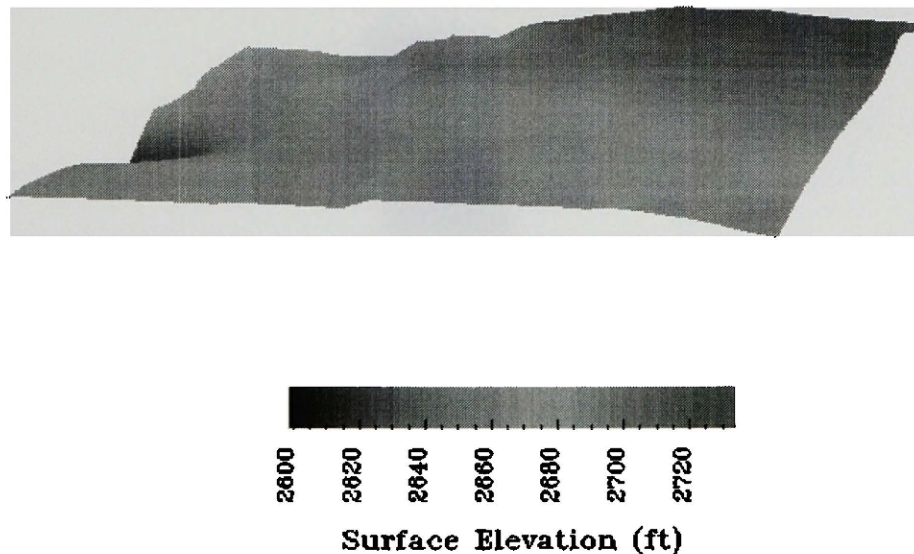


Figure 2-4: Surface elevation display.

each grid point. Values are interpolated between grid positions based on the connections component, in this case a quad connection is used between grid points. The AutoColor module outputs a color surface as well as a color map. The ColorBar module transforms the color map into a color bar with user specified annotation. The colored elevation surface and color bar are combined in the Collect module and then rendered into an image (Figure 2-4) by the Image module.

The SurfacePlot icon shown in the program (Figure 2-3) is actually a DX macro. A macro is a sequence of modules connected together, given a name, then used in the VPE as a single tool. The visualization specialist constructs macros for module sequences frequently used in visual programs. Macros make visual programs simpler by combining several icons into one. Figure 2-5 shows the SurfacePlot macro.

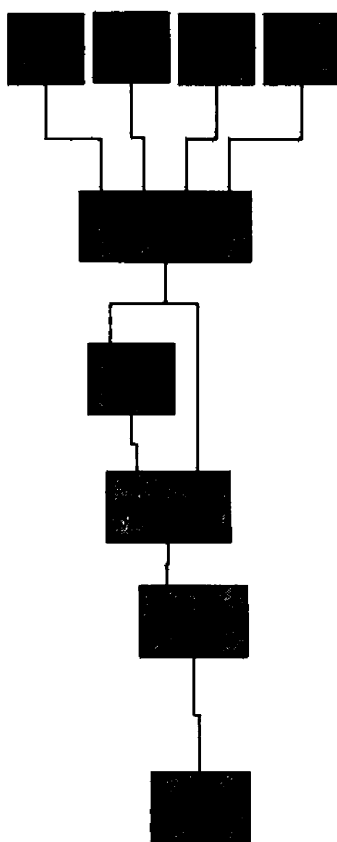


Figure 2-5: SurfacePlot Macro.

Data Explorer supplies a rich and varied set of the modules necessary to write visual programs. However, in some cases the existing DX modules are insufficient for a desired visual analysis. The visualization specialist can also construct, through custom programming, a new DX module that performs an application specific task. The Data Explorer Programmer's Reference (IBM, 1995b) discusses the steps necessary to build a module and incorporate it into the Data Explorer system. The steps are:

- 1) Define the module's function, its inputs and its outputs.
- 2) Create a module description file containing this information.
- 3) Write the module using C-code.
- 4) Compile and link the module.

Much of this work can be completed with the DX Module Builder, a graphical user interface utility that creates the necessary module files from user-supplied information. To complete the module, the visualization specialist must add the application-specific code to a C-code framework file generated by the DX Module Builder.

After the scientist or visualization specialist constructs the visual program with DX modules, macros, and customized modules, he/she needs to consider what additional information should be added to the imagery to better communicate with other scientists. Annotations such as titles, caption labels and color bars are very informative to individuals that are unfamiliar with the particular data set. The scientist can add annotation to the imagery by using the appropriate DX modules within the visual program, or by using additional software (e.g., Photoshop) to add annotation to the imagery after it is produced as DX output.

2.5 Determine Final Visualization Output Requirements

The scientist completes the visualization process by determining what the final visual artifact will be, and what the requirements of this output are. Keller and Keller (1993) discuss some considerations that affect that choice of an output medium. Different final

output types include interactive visualization programs, realtime computer imagery displays, animations saved in video form, printed images on paper, and images or videos formatted for display from Web sites. Each of these outputs have different requirements, discussed below.

Other scientists may want to use the final visualization program to examine their own data. For this situation, the visual programmer needs to design an interactive application with a “standard” display format, and flexible data import to allow the user to easily import and manipulate data in order to change the visual images. Data Explorer provides special interactor modules for such purposes. These modules consist of 2 parts: an interactor “stand-in” that is placed in the visual program and an interactor “dialog box” that is placed in a Control Panel window. Control Panel windows contain one or more dialog boxes that allow the user to choose data files, select data, and enter visualization parameters (e.g., numerical values, colors). The Sequencer, a special interactor contained in its own window with controls similar to a cassette deck or VCR (e.g., play, stop, reverse, etc.), increments an integer value which can be used to animate a sequence of images. These interactor tools allow the user to interact with the visualization application.

Scientists commonly use their visualization programs for real-time computer presentations. The images they display are limited by the size and resolution of the RGB computer monitor. Larger images communicate better with the audience, but take more CPU time to render and use more computer memory. Real time rendering allows the scientist to tailor imagery to the needs of the audience, but rendering delays also can limit the scientist’s ability to interact with the data visualization program during the presentation. To reduce re-rendering time for previously displayed images, the Image and Display modules can be used with the Sequencer module to render and cache a sequence of images in memory. If sufficient memory is available, these cached images can be replayed without re-rendering them, resulting in an almost continuous animation. However, to load an entire image sequence into memory, the number of frames and the

image size must be carefully reduced to match cache size. Thus, real-time computer animations are possible but highly restricted, and animations are best displayed if recorded on video tape.

Thompson (1997) details the process of using images produced by a Data Explorer visual program to create a video animation. To create a video animation, images must be exported from Data Explorer as Tagged Image Format Files (TIFF) and “postprocessed” into video using other multimedia software (e.g., Adobe Premiere). The NTSC video standard restricts the image size to 720x486 or 640x480 pixels. Smooth video requires at least 30 frames per second. Thus, the number and size of frames generated by DX must be set to guarantee that the video animation appears smooth. Alternatively, postprocessing tools can be used to generate additional transition frames to smooth the video flow. Using the Sequencer with the ImageWrite module allow a large sequence of TIFF images to be created and saved to disk. However, most of the postprocessing must be done “by hand”, and can thus be very labor-intensive. It is also important to add either a narrative explaining the animation, background music, or both to the video tape because audiences quickly lose interest watching a silent video.

Data Explorer supports the production of several other image formats that can easily be incorporated into Web sites to make the presentation of scientific results accessible worldwide. Web sites require compressed image format files because of limited Internet bandwidth. For still images, exported TIFF files can be compressed to the Joint Photography Experts Group (JPEG) or directly output compressed Graphics Interchange Format (GIF) formats. For Web animations, Holbrook (1996) discusses the conversion of TIFF images into an MPEG movie format that can be downloaded and displayed through most Web browsers. Data Explorer also supports the MIFF format that contains the entire image sequence in a single file which can be converted to a MPEG or Quicktime movie. A 3-D visualization model can also be exported from DX as a Virtual Reality Modeling Language (VRML) object that users can access over the Web with a VRML

capable browser. Such browsers allow the remote user to interact (zoom, rotate, etc.) with the VRML object to produce different displays of that object.

The last output type is printed hardcopy images. To produce an image for printing requires special preparation of the visualization image before it is exported from the Data Explorer environment. The background color is normally changed to white for printing purposes, whereas annotations need to be changed to black. Laser printers support higher resolution than color monitors, so the images can be redefined with higher resolution. Postscript and color postscript files can be generated in DX and sent directly to laser printers, or postscript files can be exported from DX and subsequently imported to word processing documents. For word processors that do not support postscript (e.g., MS Word), the image can be saved as a TIFF file, then can be converted to Bit Map (BMP) format for inclusion in documents.

Chapter 3: Visualization of Subsurface Geologic Formations

A computer visualization of the subsurface at the NE Rabbit Hills Oil field was constructed from formation top elevations for seven different geologic units. Formation top elevations are the basis for constructing 2-dimensional subsurface structure contour maps. The five step visualization process was used to transform the formation top data into the 3-dimensional subsurface visualization shown in Figure 3-1.

3.1 Geologic Formation Top Data

Geoscientists pick geologic formation tops from either electric wireline logs or driller logs. Electric well log data are representative of the subsurface formations encountered in the borehole. The geologist analyzes these logs to delineate formation boundaries. Geoscientists correlate the formations or patterns recognized in one electric log to surrounding well logs. Tearpock and Bischke (1991) discuss the procedure in which a geologist takes a correlation type log that exhibits a complete stratigraphic section for a region and correlates the geologic formations to surrounding logs. The interpreted formation tops are the basis for a variety of subsurface maps and geologic cross sections.

Dr. Karen Porter of the Montana Bureau of Mines and Geology provided formation top data for 21 wells in the NE Rabbit Hills study area. Dr. Porter performed detailed correlations for four geologic units (Rierdon, Sawtooth, Bowes, and Firemoon) within the zone of interest. The Sawtooth formation contains the productive oil reservoir at the NE Rabbit Hills oil field. Porter (1997) discusses the geology of the NE Rabbit Hills field in detail.

Additional formation top data came from NE Rabbit Hills driller logs. Drilling operators note changes in drill rate and borehole cuttings to determine when different geologic formations are encountered in the borehole. Three formations, the Eagle, Greenhorn, and

Rabbit Hills Subsurface Structure

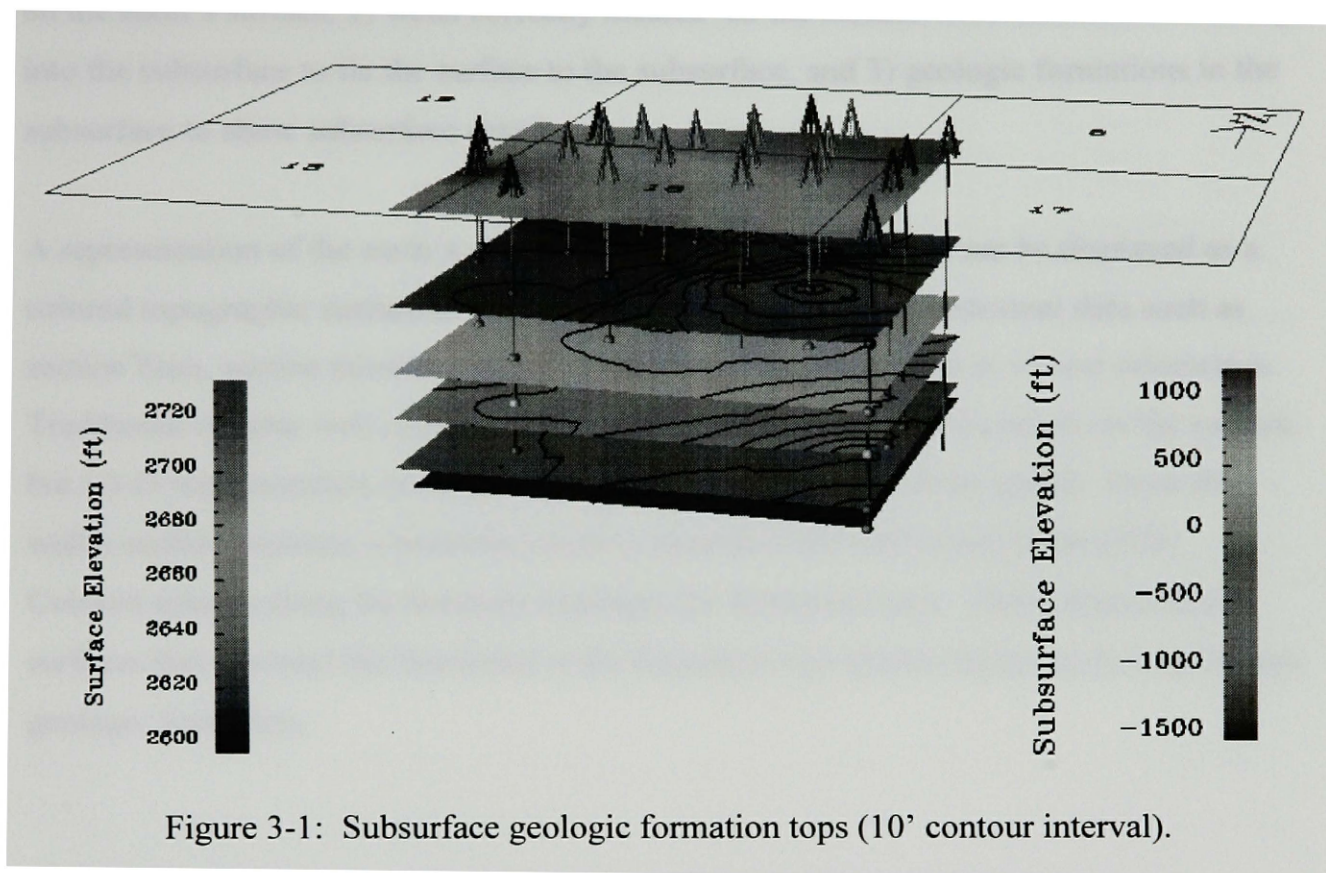


Figure 3-1: Subsurface geologic formation tops (10' contour interval).

Muddy, were consistently noted in the driller logs in the Rabbit Hills area and were also used in developing the data used in the 3-D subsurface computer visualization.

3.2 Vision for the Geologic Subsurface Visualization

Computer visualization work completed by Holbrook (1996) shows a fly-through of a hypothetical subsurface structure at the NE Rabbit Hills oil field. Holbrook's results are the basis for the subsurface visualization created in this chapter from the geologic formation top data. The "goal" is for a scientist to be able to interactively examine the 3-D subsurface geologic model from any point within the field, and with field of view

oriented in any direction. A related goal is to construct a “fly through” of the subsurface along a fixed flight path that represents a sequence of such viewpoints. The desired final subsurface model should have three key features: 1) a ground surface to orient the viewer on the earth’s surface, 2) wells correctly located on the surface with boreholes extending into the subsurface to tie the surface to the subsurface, and 3) geologic formations in the subsurface to show subsurface detail.

A representation of the earth’s surface helps orient the viewer. It can be displayed as a colored topographic surface showing vertical relief, as well as contextual data such as section lines, section numbers, and a “north arrow” to further aide in viewer orientation. Traditional circular well symbols could be used to show the well locations on the surface, but a 3-D representation of a well derrick adds to the realism of the model. From the well’s surface location, a borehole needs to extend to the well’s total depth (TD). Colored spheres along the borehole highlight the formation tops. Three-dimensional surfaces that intersect the boreholes at the formation top spheres represent the top of each geologic formation.

3.3 Preparing and Importing Formation Top Data into Data Explorer

The formation tops were furnished on paper in tabular form. These data were entered into a spreadsheet style file organized with each well’s surface location (x, y, z) followed by formation top elevations (see Table 3.1). Included with each well are the total depth, a well type flag value (oil or dryhole), spud date, and well name.

This data file is readily imported into Data Explorer with the General Array Importer. In this case, the well data are treated as 2-dimensional scattered data with well locations designated by the first two columns. The remaining data columns are separate Field objects in the DX Data Model that share a common positions component designated by the first two (X and Y) columns. There are thirteen Field objects that make up a single

X	Y	Z	Eagle	Green	Muddy	Rierd	datum	Sawt	UBow	Fire	TD	O	M	Y	Well Name
2079519	625952	2688	1231	221	-493	-1221	-1392	-1408	-1428	-9999	4163	1	7	75	Brown 7-1
2082439	626942	2733	1194	209	-421	-1218	-1385	-1397	-1418	-1478	4300	1	2	92	Norfolk 7-9
2081779	627162	2729	1201	205	-435	-1231	-1401	-1414	-1434	-1488	4280	0	10	91	Norfolk 7-10
2080422	625654	2693	1216	225	-420	-1203	-1371	-1385	-1407	-1465	4254	1	5	91	Norfolk 7-14
2082396	625637	2711	1210	218	-422	-1217	-1382	-1396	-1419	-1486	4250	1	10	91	Norfolk 7-16
2078809	626072	2646	-9999	-9999	-9999	-1210	-1377	-1391	-1405	-9999	4109	0	10	76	Brown 7-12
2077489	624522	2655	1226	218	-502	-1231	-1402	-1415	-1435	-9999	4150	0	7	75	Brown 13-12
2077734	623642	2596	1210	210	-432	-1225	-1395	-1409	-1434	-1486	4100	0	11	80	Texas O&G 1-
2083759	623592	2694	-9999	-9999	-9999	-1224	-1392	-1405	-1414	-1482	4210	0	3	92	Brown 17-5
2083759	624592	2701	-9999	-9999	-9999	-1223	-1387	-1403	-1413	-1470	4200	0	2	92	Brown 17-4
2078809	621992	2659	1206	216	-484	-1221	-1395	-1407	-1429	-9999	4150	0	7	75	Brown 18-2
2082602	623642	2683	1119	225	-476	-1205	-1370	-1383	-1401	-1455	4226	1	1	92	Meridian 42-18
2080265	624502	2672	1217	216	-477	-1212	-1380	-1392	-1418	-1480	4175	1	9	91	Meridian 21-18
2084089	625952	2735	-9999	-9999	-9999	-9999	-9999	-9999	-9999	-9999	4710	0	8	70	Inexco 1-
2079031	625622	2675	-9999	-9999	-9999	-9999	-9999	-9999	-9999	-9999	4069	1	8	84	Brown 7-34-20
2081139	626255	2711	1163	210	-415	-1212	-1380	-1392	-1413	-1472	4236	1	12	90	Norfolk 7-15
2081466	624674	2684	1187	-9999	-475	-1217	-1381	-1395	-1418	-1483	4200	1	4	91	Behm 1-
2081665	623448	2658	1143	-9999	-492	-1221	-1399	-1412	-1433	-1515	4200	1	11	91	Behm 2-
2082510	624643	2698	1142	218	-470	-1207	-1374	-1388	-1409	-1482	4236	1	11	91	Meridian 41-18
2079797	623256	2656	1211	226	-490	-1215	-1383	-1395	-1414	-1477	4200	1	11	91	Meridian 22-18
2078837	624485	2655	1220	-9999	-9999	-1220	-1391	-1405	-1423	-1490	4200	1	1	73	Amoco 1-
2084086	620739	2651	1176	184	-420	-1245	-1421	-1436	-1453	-1531	4206	0	12	69	InterNucl 1-18

Table 3-1: NE Rabbit Hills formation top data.

Group object. The well surface elevation is one such Field. The data are not imported with 3-dimensional positions, because that would imply the formation top elevations are located at the well's surface location. The well names are not imported into DX.

3.4 Visualization Analysis

The subsurface visualization analysis can be broken down into three main components: ground surface topography, wells, and subsurface geologic formations. Figure 3-2 shows the visual program broken down into the separate parts or pages. On the main program page the data are imported for use on this and other pages. The ground surface portion of the visualization is completed and combined with the plotted wells and subsurface results on this main page. A fly through of these visual objects is completed on the Flythru page. The visualization analysis of these four parts is discussed next.

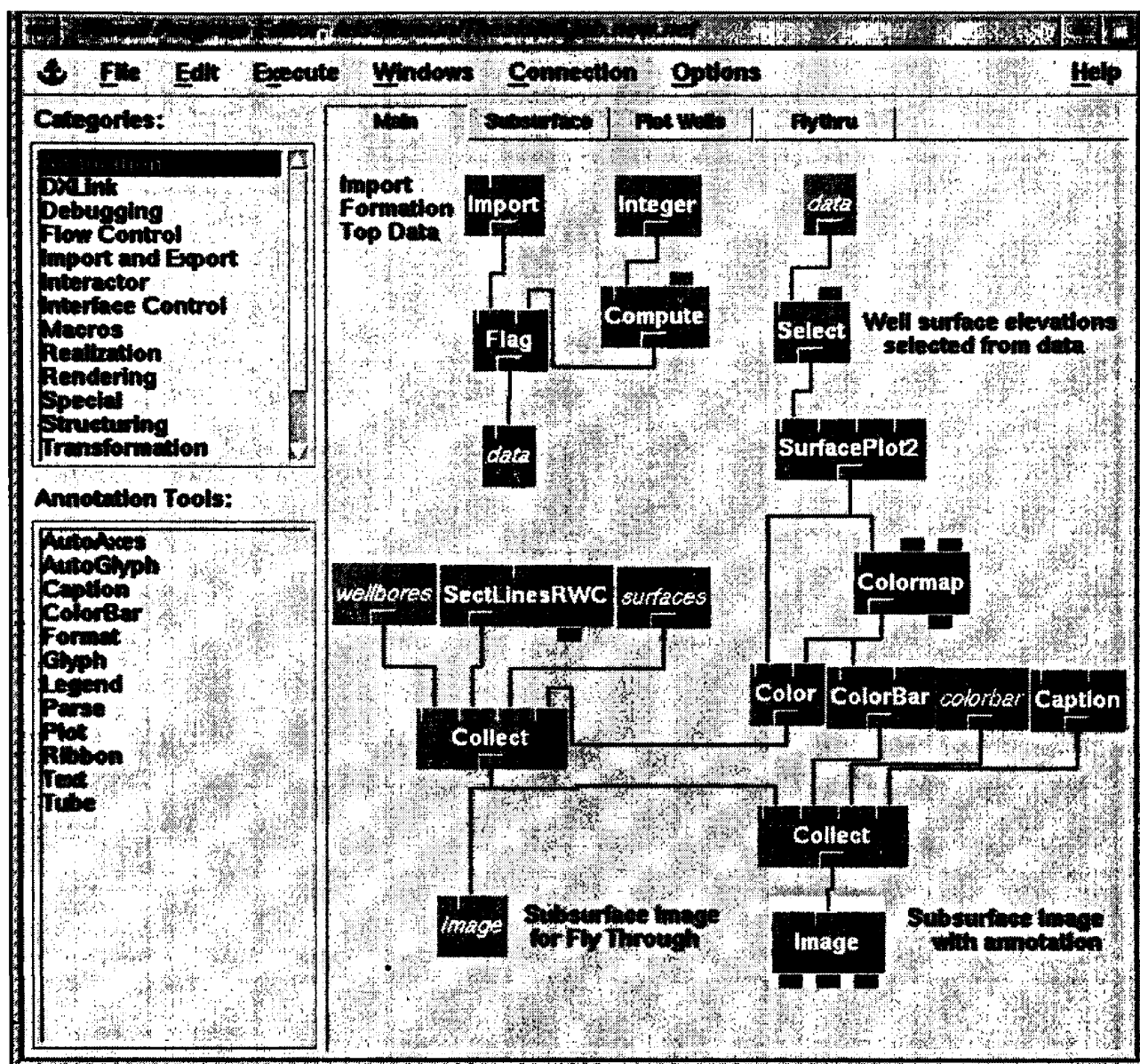


Figure 3-2: Geologic formation top visual program.

The ground surface portion of the visualization provides contextual information to the viewer. It consists of two sub-parts: the topography and section lines. A simple ground surface topography visualization was presented in Chapter 2 to demonstrate how visualization analysis is performed in Data Explorer. The same visual analysis is done in the Main portion of the program in Figure 3-2. The only difference is that the well surface elevation field is first selected from the Group object. Building on this visual program, a separate DX Macro, SectionLines, was created to display section lines in the

NE Rabbit Hills area. This macro constructs and displays a 2-dimensional grid with 1 square mile cells. In the center of each cell is placed a section number. A north arrow is put in the northeastern most section to orient the viewer. This output from SectionLines is placed on top of the ground surface topography to complete the ground surface portion of the image. This macro is used in many other NE Rabbit Hills visualizations that require section lines.

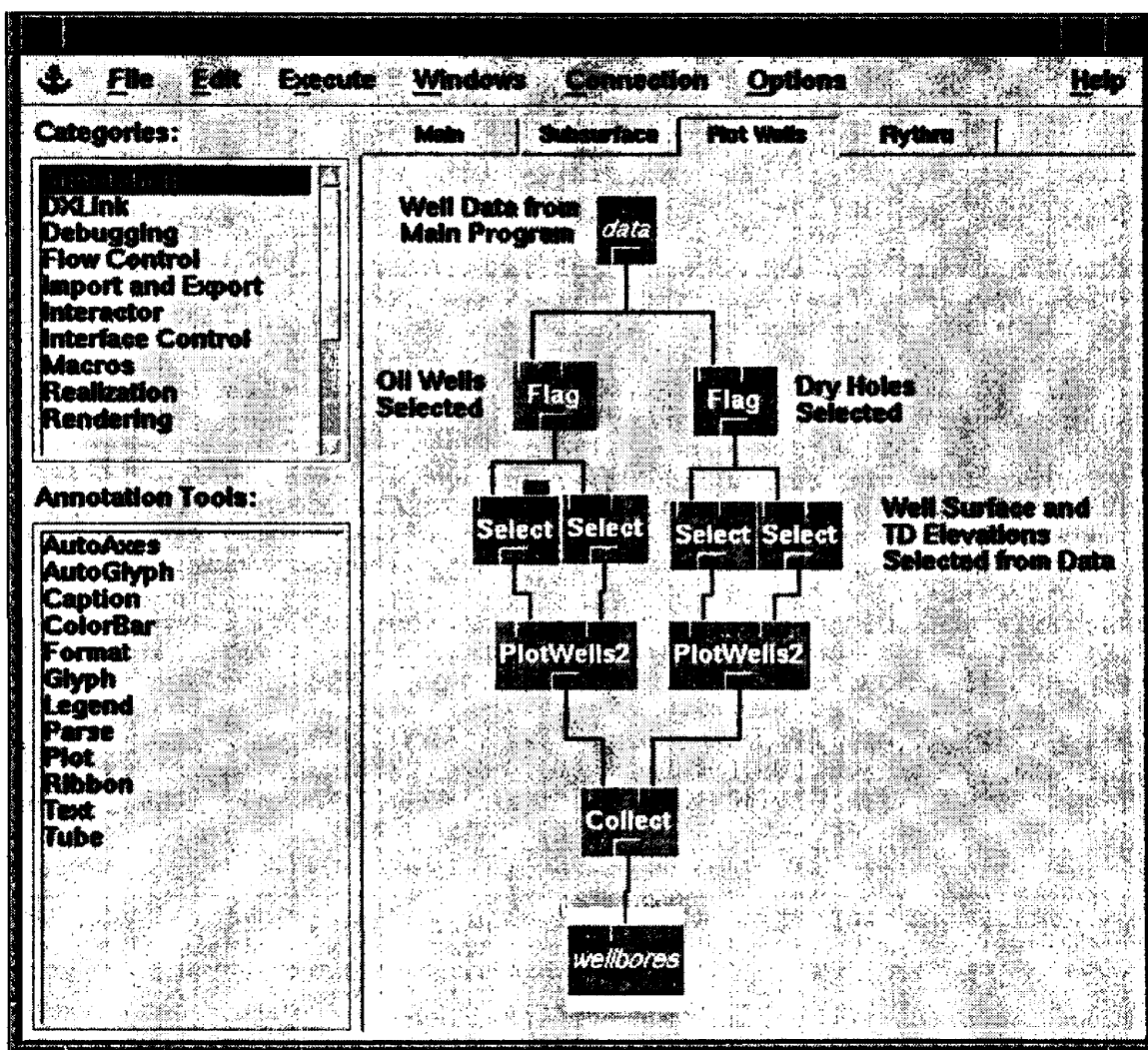


Figure 3-3: Visual Program portion for plotting well derricks and boreholes.

Plotting the well derricks and boreholes is done in two parts (Figure 3-3). The first part separates the wells into two categories: oil wells and dry holes. Then, each well category is plotted in a different color using the PlotWells macro. The oil well flag value designates which wells belong in each category. Data Explorer does not provide a means for selecting multiple data fields based on flag values, so a custom module, Flag, was created for this purpose.

Data flags are useful in instances where certain data members are chosen based on another parameter or field. Flags for individual Field objects can be represented by using a null value. In Table 3-1, a null value of -9999 excludes invalid formation top data for certain wells from any visualization analysis. Data Explorer does not support flags when Group objects are used, where one or more Fields are some type of flag parameter. For the data in Table 3-1, we want to use the surface (z) and total depth (td) elevation fields to plot the top and bottom of well bores with a color based on the oil well flag field. This is accomplished by using the Flag module (Braun, 1995).

The Flag module redefines the Group object so it retains the original Field objects, but only contains members with a flag value that satisfies a flag condition specified within the visual program. All data members that do not meet the flag conditions are removed or culled by the Flag module. Flag conditions are set with selection code input parameter in the Flag module dialog box (Figure 3-4). Greater than, equal to, or less than conditions are specified with integer values of 1, 0, or -1, respectively. The flag member input parameter allows the user to specify which Field in the Group contains the flag values. Another example of using the Flag module is in the Main program page (Figure 3-2) where the imported well data Group is redefined to contain wells drilled in or prior to a specified year. The cull input is used to remove invalid data from the DX Field objects. DX normally marks invalid data members but leaves them in the Field object.

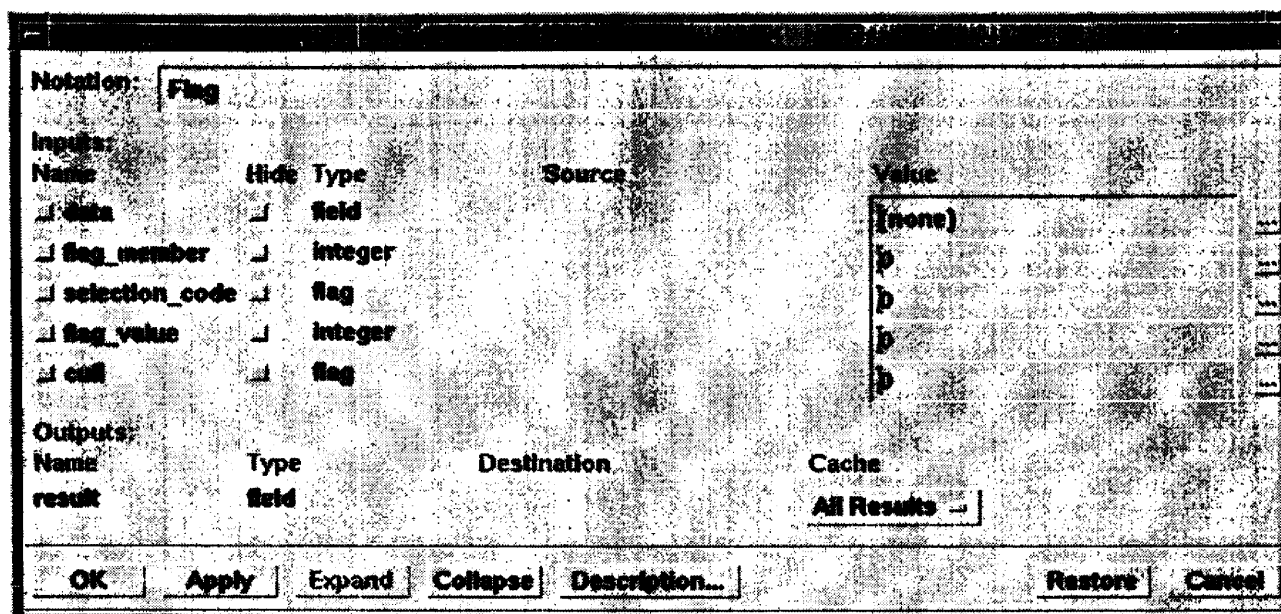


Figure 3-4: Flag module dialog box.

After the oil wells and dry holes have been separated using the Flag module, each well group is input into the PlotWells macro. This macro plots well derricks at the well's surface location and extends the bore hole in the subsurface to well's total depth. The macro's input parameters allow the user to select the well's color and the derrick's size. The output from both PlotWells macros are collected and used on the Main program page.

The last part of constructing the 3-D image is visualizing the geologic formations. The same SurfacePlot module used to visualize the earth's surface is used to display the subsurface. The SurfacePlot module regrids the scattered formation top elevation data onto a user specified 2-D grid. The 2-D grid data is then converted into a 3-D surface for the top of each formation. A separate macro, PlotWellTops, plots a sphere at the position of formation top in the borehole.

The well bores, section lines, surface and subsurface objects are collected in the lower left of the Main program page (Figure 3-2). In the lower right, annotations (color bars and captions) are added to this Group object and rendered in the Image window. The Image window supports interactive tools that allow the user to zoom in and rotate the final 3-D

visual object (Figure 3-1). The group of objects without annotation is also used for the subsurface fly-through completed on the Flythru page. Annotation is not added to the fly-through animation because it tends to clutter the display. The subsurface fly-through portion of the visual program is based on Holbrook's (1996) results. A fixed flight path is chosen to first fly over the oil field's surface to orient the viewer prior to flying through the subsurface structure. The fly-through completes the visualization analysis, though some changes to the visual program maybe needed to support specific final output requirements.

3.5 Final Visualization Output Requirements

Several different outputs can be created from this visualization. One desired output is a video animation of the subsurface fly-through. Creating video animation from the DX output is discussed in detail in Chapter 4. Another desired output is an interactive visual program that geoscientists can use to examine the subsurface geology. Several DX interactor tools can be added to the visual program to make it more interactive. Interactor tools placed in the Subsurface controls window (Figure 3-5) allow the user to turn on and off selected geologic formations. This allows the scientist to examine one formation in detail or to better examine the relationship between two or more geologic formations of interest. Additional interactors are used to turn on and off contour lines and specify the contour interval. The Year interactor allows the scientist to use formation top data from wells drilled in or prior to the specified year, giving the scientist insight in how the subsurface could have been previously mapped with less well data. The Show Flythru toggle box is used to turn on and off a separate Image window that displays the fly-through of the subsurface without any annotations. Turning off this second Image window saves memory and CPU time. The main Image display window supports additional tools needed to rotate and zoom in on the subsurface image.

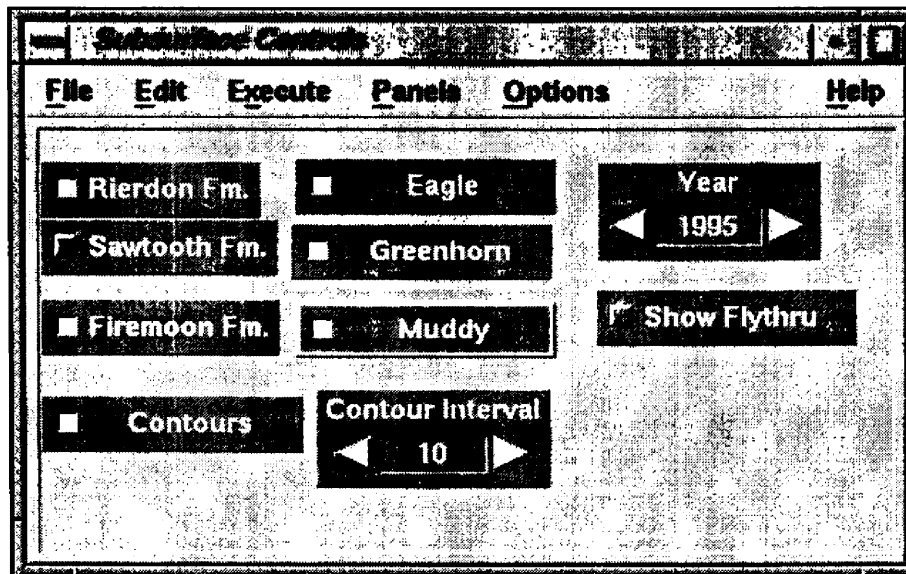


Figure 3-5: Subsurface Control Panel containing interactors

The resulting visual effects show that the subsurface geologic structures for the NE Rabbit Hills oil field are rather flat with elevation changes of less than 50 feet in the 1 square mile field area (Figure 3-1). This visual program for examining the subsurface would be more insightful in areas of complex geology.

Chapter 4: Visualization of Oil Production Data

Production data from the NE Rabbit Hills oil field are used to create a visual animation showing the field's drilling history and cumulative oil production. The production data are combined with contextual data discussed in Chapter 3 to produce the animation through the five step visualization process.

4.1 Monthly Oil Production Data

Production data from individual oil wells are recorded daily by the operating oil company. These data include the amounts of oil, gas, and water produced from the well. Oil and water production are measured in barrels (42 gallons) and gas production is measured in thousands of cubic feet (MCF). The operating company reports each month's total production and the number of days the well operated to the Montana Board of Oil and Gas Conservation. These data are stored in Excel spreadsheets and are publicly available. Several times a year the Montana Board of Oil and Gas sends updated NE Rabbit Hills production data to Dr. Wideman, the team leader of the Montana University System Petroleum Reservoir research team. These data show that the NE Rabbit Hills' wells produce a mixture of oil and water with no associated gas. A primary project goal is to produce a visualization of the NE Rabbit Hills oil production data that helps both petroleum experts and non-experts to better understand the production data.

4.2 Vision for the Oil Production Animation

Traditionally, each well's production data are displayed as 2-D plots of production rate versus time (Figure 4-1). The production rate is calculated by dividing the total month's production by the well's number of operating days. These graphs can be used to analyze

Amoco #1 Monthly Production Data

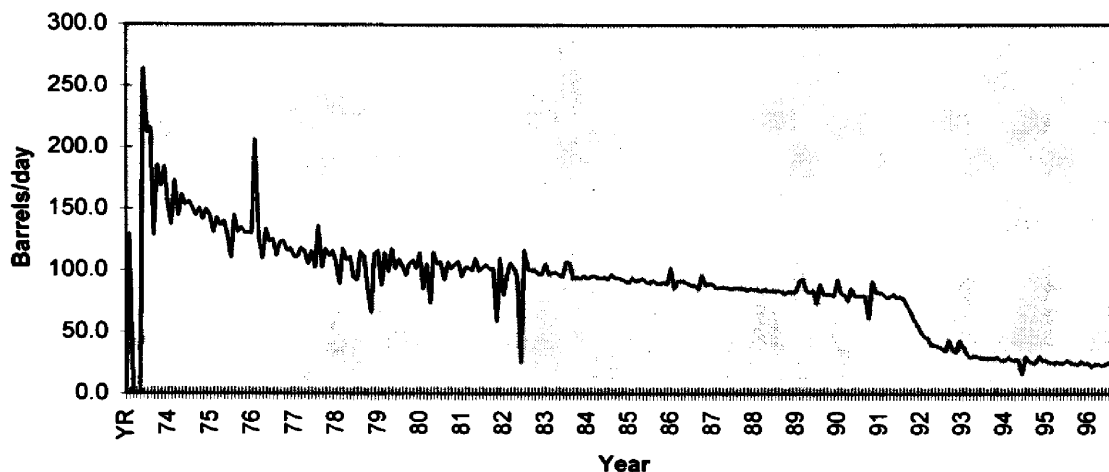


Figure 4-1: Monthly oil production rates for the Amoco #1 well.

the oil decline rate and changes in water production rates to help predict future well performance. Production data from multiple wells can be plotted on the same graph to study relationships between the field's producing wells.

The time varying production data are well suited for visual animation, to complement the simple 2-D plots. An animation can show monthly oil production for all the field's wells simultaneously varying through time. The data types to animate are each well's total monthly production, each well's average daily production, or each well's cumulative production. After looking at various types of displays, I decided it would be more informative to animate the cumulative oil production for each well. Viewing cumulative oil production is easier for the viewer to understand because the cumulative data are non-decreasing. Animating total monthly production and daily production rates are more difficult for the viewer to assimilate since an overwhelming amount of varying data are presented. It is difficult for the viewer to track short term oil production increases and decreases in the different wells. Another advantage of using cumulative oil production is that a drainage area in the oil reservoir can be approximated from the well's total production.

The visualization goal for the oil production visualization is to show which portions of the reservoir have produced the most oil. The actual depleted area of the reservoir is impossible to calculate from monthly production data alone (i.e., it would require detailed knowledge of the underground structure). However, cylindrical drainage areas can be used to approximate the cumulative production effect. The volume of oil produced can be converted to a cylindrical in situ reservoir volume by using average reservoir rock properties. The cylindrical volumes can be shown as circular disks in the visualization. Each disk is color coded to represent the cumulated amount of oil produced by the well. These disks are displayed in the subsurface at the level of the producing zone. Well bores tie the disks to the well derricks located on the earth's surface. As in Chapter 3, dry holes are plotted and colored differently than the productive wells, and section lines are placed on the earth's surface to orient the viewer.

The final visualization product is an animation showing the drainage area disks growing in size and changing in color as the wells produce more oil. Annotation showing the current year is also included. Both oil wells and dry holes are only displayed after the well's completion date. Thus, the resulting animation shows the drilling history of the wells and how much each well has produced.

4.3 Preparing and Importing the Oil Production Data into Data Explorer

Three Excel spreadsheet files contain the NE Rabbit Hills monthly production data for the years 1973-94, 1995, and 1996. Figure 4-2 shows a portion of the 1995 spreadsheet. Wells are identified by their API number in the spreadsheet's first column. The second and third columns contain the year and month of the production data. All wells have the total monthly oil production and the total number of days the well produced. Some wells are missing water production data. The API number, month, year, and oil production data are exported from Excel and combined into a single ASCII text file.

APINO	YR	MO	DAYS	FM	OIL	GAS	WTR	AGAS
00521420	95	1	28	ST	583		4076	
00521420	95	2	28	ST	598		3951	
00521420	95	3	28	ST	547		3788	
00521420	95	4	5	ST	88		636	
00521420	95	5	31	ST	691		4443	
00521420	95	6	30	ST	705		4291	
00521420	95	7	31	ST	669		4391	
00521420	95	8	31	ST	648		4338	
00521420	95	9	30	ST	609		4134	
00521420	95	10	31	ST	633		4287	
00521420	95	11	30	ST	617		4174	
00521420	95	12	31	ST	650		4318	
00521511	95	1	30	ST	807		422	
00521511	95	2	28	ST	718		400	
00521511	95	3	31	ST	774		424	
00521511	95	4	29	ST	751		385	
00521511	95	5	31	ST	766		419	
00521511	95	6	29	ST	729		411	
00521511	95	7	27	ST	743		443	
00521511	95	8	31	ST	772		400	

Figure 4-2: Spreadsheet containing a portion of the 1995 NE Rabbit Hills production.

The ASCII text file has an irregular format that is not readily imported into DX. A conversion program (Appendix A) was written to organize the data into a time series, calculate each well's cumulated oil production at each time step, and assign positions to the data based on the well's API number. The conversion program outputs regular time series data that can be described by the general array importer and imported into Data Explorer for the visualization analysis.

4.4 Oil Production Visual Analysis

The visual analysis for the oil production data is similar to the subsurface visual analysis presented in Chapter 3. The visualization is broken down into three main components: the earth's surface, wells, and cumulative oil production. The well formation top data (Table 3-1) are input into a similar network of DX modules used in the subsurface visual

program to render the earth's surface, well derricks, and well bores. The wells in this visualization are only visible for time steps after the well's completion date. Therefore, the wells must be turned on when the time series reaches the well's completion date. The Flag module was originally designed for this task. The Flag module selects wells with completion dates prior to the current time step. The current well data are then used to plot the well derricks and well bores in a color corresponding to the well type (oil well or dry hole). Surface elevations from all the wells are used to render the earth's surface.

Figure 4-3 shows the portion of visual program network that manipulates the cumulative oil production data into visual objects. The Sequencer provides the animation by incrementing the time steps. The Select module selects production data for the current time step. The Include module and NewZ macro give the data a depth position, with the larger production data receiving a slightly deeper z- position. Next, colors are assigned to the production data. The cumulative production amounts are then converted to a cylindrical drainage area based on an average reservoir thickness of 15 feet, average porosity of 15%, average oil saturation of 75% and average recovery factor of 30%. The colored disks, annotation, plotted wells, and the earth's surface are collected and rendered into a display image at the bottom of the visual program network.

Figure 4-4 displays the cumulative oil production for December, 1990. Ten wells had been drilled in the NE Rabbit Hills field area by that time, with four wells completed for oil production. The discovery well, Amoco #1, had produced over 600,000 barrels of oil and the Brown # 7-34-20 had produced just under 100,000 barrels of oil at the end of 1990. The Brown #7-1 located north of these two wells had minimal production and thus its disk is not visible. The Norfolk #7-15, the northeastern green well, was completed in December, 1990 and has no oil production at this point.

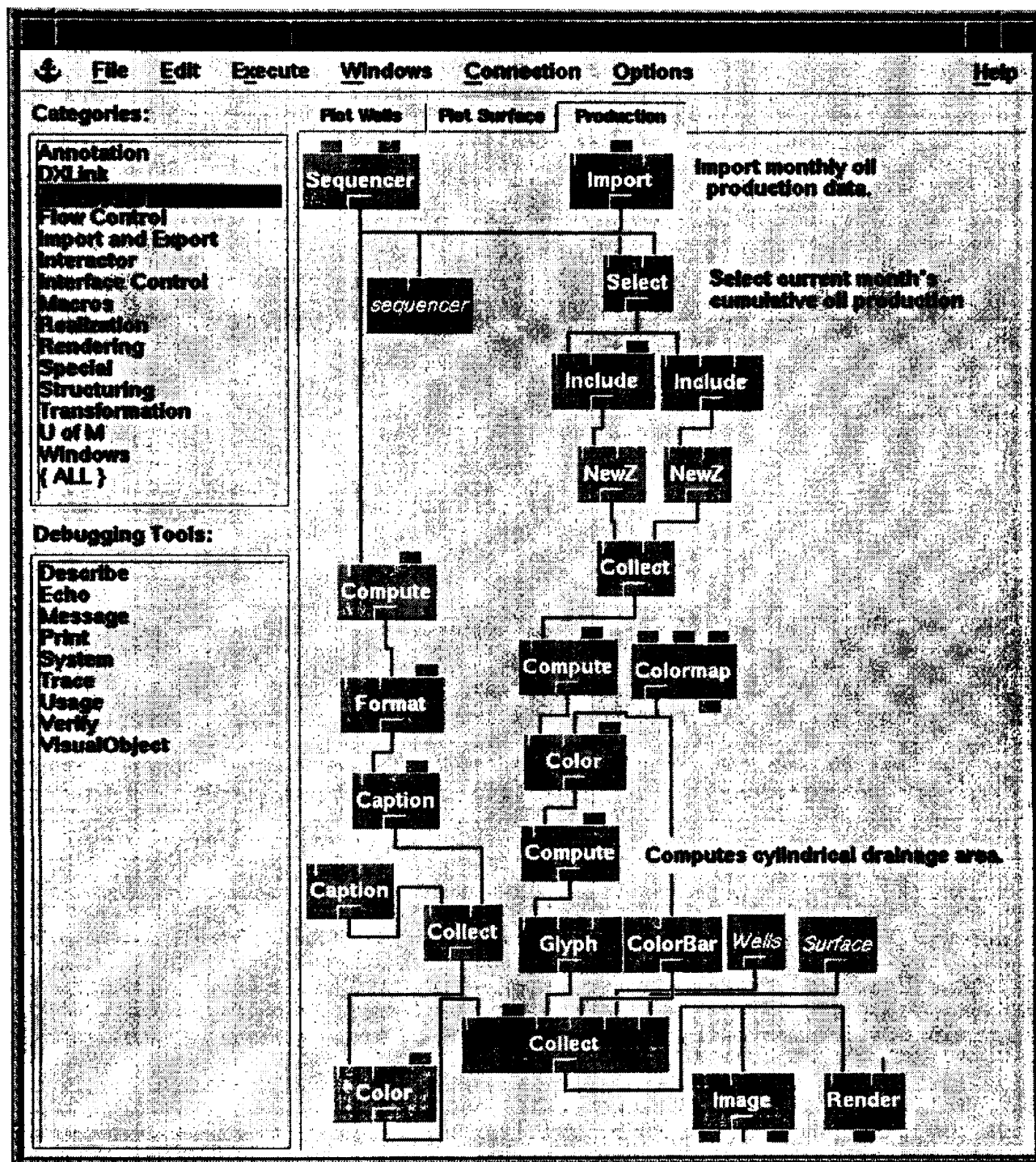


Figure 4-3: Main network for the cumulative oil production visualization.

Rabbit Hills Production Data

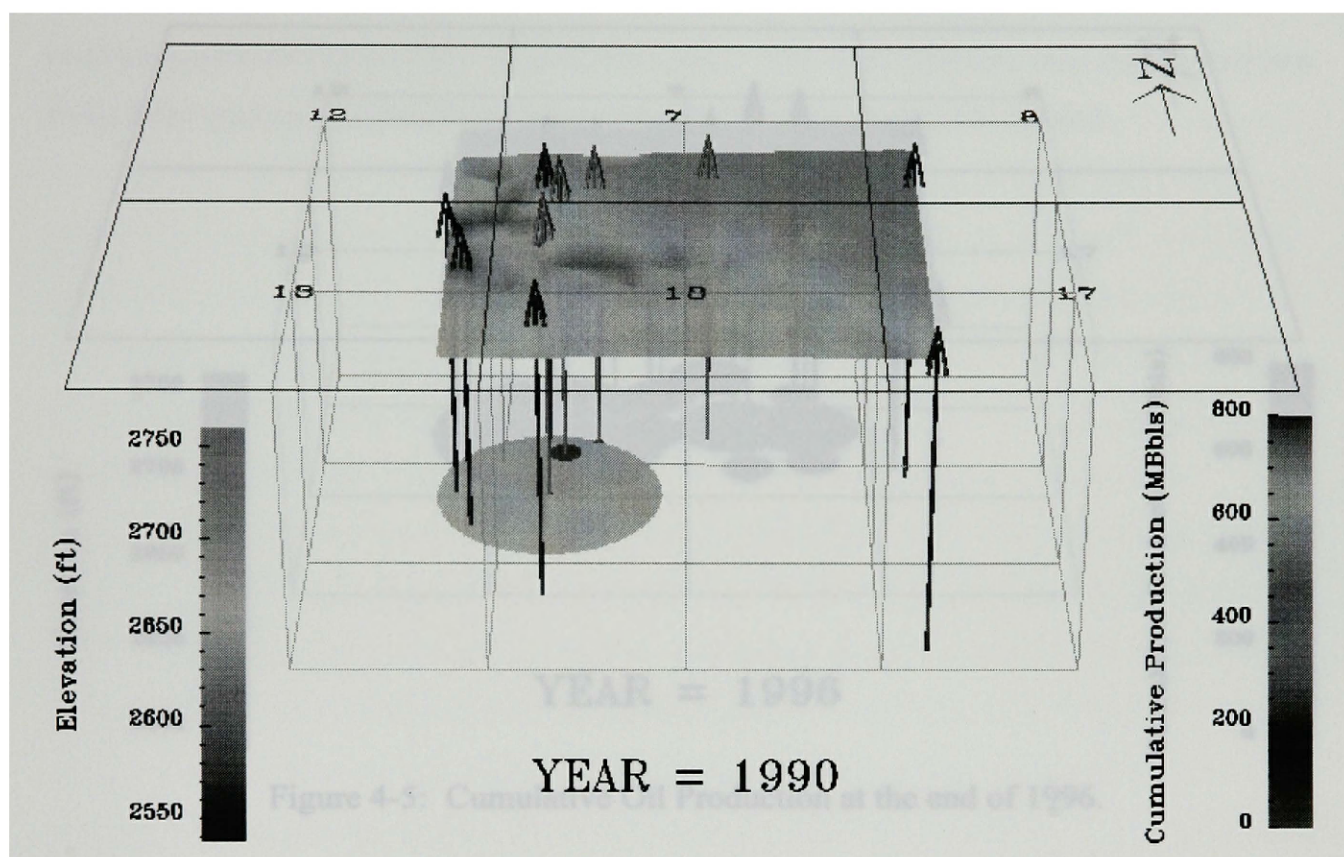


Figure 4-4: Cumulative Oil Production at the end of 1990.

The cumulative production data at the end of 1996 are shown in Figure 4-5. There are now a total of 22 drilled wells, 13 of which were completed for production. The Amoco #1 well has produced nearly 750,000 barrels of oil. At the end of 1996, the NE Rabbit Hills oil field had produced over 1.4 million barrels of oil. One inadequacy of using cylindrical drainage areas is exhibited in this figure by the Amoco #1 well, whose disk is penetrated by several dry holes that did not encounter commercial quantities of oil. In reality other data suggest that the Amoco #1 well appears to have drained portions of the reservoir eastward of its drainage disk.

Rabbit Hills Production Data

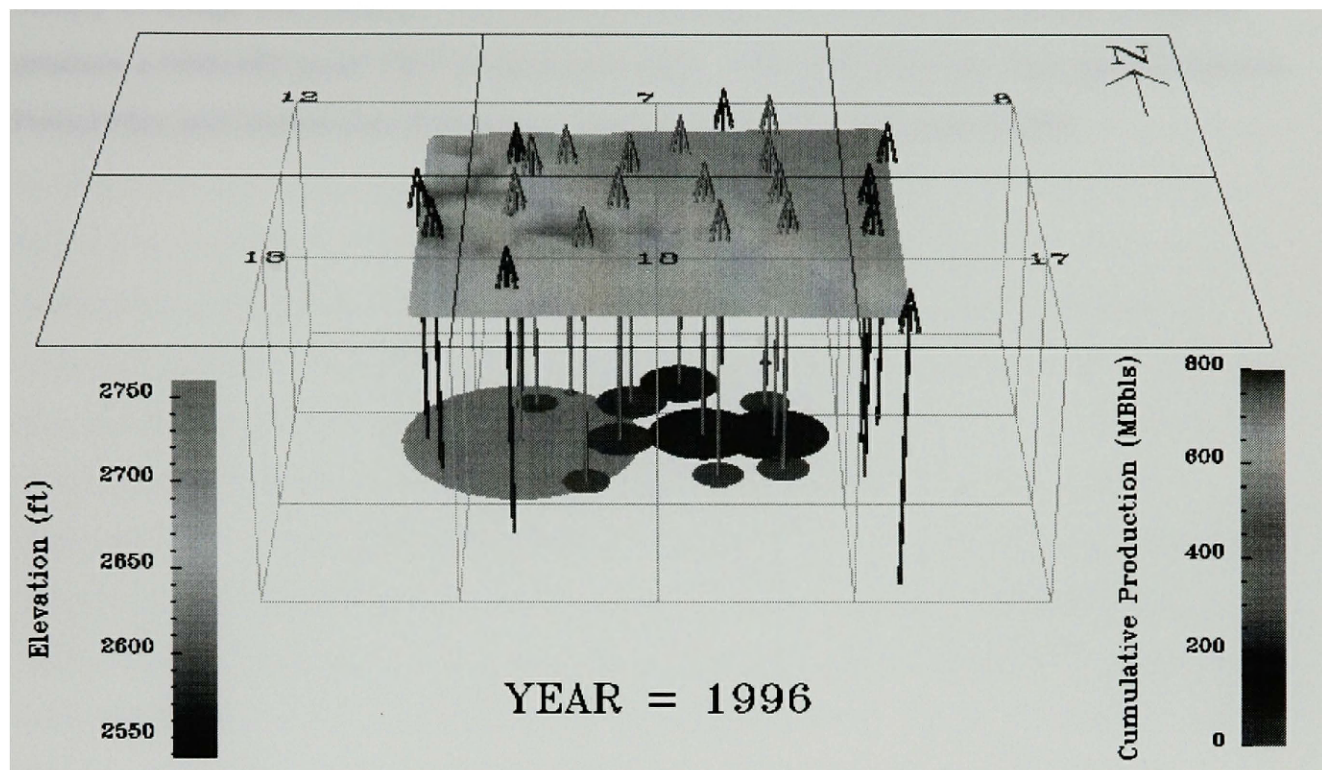


Figure 4-5: Cumulative Oil Production at the end of 1996.

4.5 Data Explorer Animation Output

Realtime animation of the oil production visualization within the Data Explorer environment is relatively slow because 300 frames must be rendered. Rendering every 12th time step (1 frame per year) produces an animation that finishes in a reasonable amount of time. However, the animation becomes rather choppy. A solution to this problem is to output the animation frames from Data Explorer and put them into a video. The process of transforming Data Explorer images into video requires several labor-intensive steps and additional software. Adobe Premiere is the software used to assemble the frames and add audio.

To save the images produced by DX, the WriteImage module is connected to the Render module in the visual program shown in Figure 4-3. The WriteImage module supports a variety of image file formats. For video production, the WriteImage module is used to produce a 640x480 pixel TIFF at each time step. The TIFF files are then transferred to a PowerMac and loaded into Premiere, where the video animation is created.

Chapter 5: Visualization of Reservoir Model Parameters

The reservoir characterization study culminates with the integration of all the data sets to produce a 3-D reservoir model used in a reservoir simulation. Reservoir simulations are an important part of understanding and depicting the complex fluid movements within a petroleum reservoir. Petroleum engineers use computer simulations to predict reservoir performance required for economic forecasts and to evaluate different enhanced oil recovery scenarios. Simulators output a time series of reservoir model parameters that any specialized modeler support graphical tools will provide only a limited means of displaying. Petroleum scientists need more powerful visualization tools to better understand the characteristics of a reservoir (Huang, 1996). For example, the reservoir computer simulation program, SimMT (Ahmed, 1993), used in the NE Rabbit Hills reservoir characterization study provides no graphical support. Simple 2-D visualization tools have previously been applied to display single time steps from the SimMT output. Here, I describe the use of the Data Explorer visualization system to develop an animation of the entire SimMT reservoir parameter time series output.

5.1 Reservoir Simulation Data

Two types of data are collected for visualization from the reservoir model simulation study: initial model input parameters and time series output parameters. The input parameters include formation top elevation, thickness, porosity, permeability, initial reservoir pressure, and initial oil and water saturation. These data are derived from integrating core data, well log data, and seismic data. Traditionally, porosity and permeability distributions are interpolated between wells using conventional gridding techniques. The NE Rabbit Hills reservoir model discussed in this chapter is constructed from conventionally gridded well data. Additional work completed by Ahmed et al. (1997) uses artificial neural networks to assimilate the NE Rabbit Hills reservoir data and

predict the reservoir parameter distributions. Their neural network approach correlates well log porosity and permeability with seismic attributes to more accurately constrain the porosity and permeability distribution between wells.

The model parameters for the reservoir simulator are distributed onto a 24 x 30 x 2 irregular grid covering a 9000 ft by 9000 ft area. The horizontal grid spacing is not constant, varying from 150 ft to 1500 ft. Finer grid spacing is used near the producing wells, and coarser grid spacing is used at the field's perimeter. The model consists of two layers of cells. Flow only takes place between neighboring cells within a layer or between layers via overlying cells. The thickness of the cells vary from 0 to 25 feet.

The reservoir simulation study consists of two steps. First, a **history match** is performed that modifies and tunes initial model parameters until the output matches observed field production data (George, 1994). After the reservoir model is thus calibrated, various production scenarios can be tested. A base case scenario predicts the ultimate oil recovery with continued operation of the existing oil wells. Other enhanced oil recovery scenarios (e.g., water flooding, polymer flooding) can also be evaluated with the reservoir simulator. For all types of simulations, the SimMT simulator outputs the state of the reservoir model at various time steps. However, the time step interval is irregular, being determined dynamically by changes in the field's operation history (e.g., shut in wells, new wells). Reservoir pressure and oil saturation are two of the model parameters output from the simulator as time series data, both of which are used in the visualization analysis discussed next.

5.2 Visualization Goals

One visualization goal is to show some or all of the input parameters for the simulation in a 3-D representation that allows scientists to view and inspect the parameters. Model consistency can be readily checked with this type of data visualization (Slatt et al., 1996).

Of particular interest is confirming that all model cell neighbors are connected to one another. To check cell continuity, a 3-D visualization showing the cell positions is created from the reservoir elevation and thickness parameters. If a cell does not connect with a neighbor within the same layer, then there must be an error in the specification of the reservoir elevation or thickness parameters. Other model parameters are also visually examined for inconsistencies. To differentiate the cell boundaries, a Rubbersheet function is used to transform the 2-D data into 3-D images (Figure 5-1). The layers are separated so that all model parameters are viewable within the image display.

Another visualization goal is to animate the changes in key model parameters during reservoir depletion. In particular, changing reservoir pressures and oil saturation during production are known to have a dramatic effect on production. Animating these changes helps scientists understand fluid migration within the reservoir. Areas of the reservoir that are depleted or have additional potential can be identified from these animations. Different treatment scenarios can also be depicted through various parameter changes,

Thickness

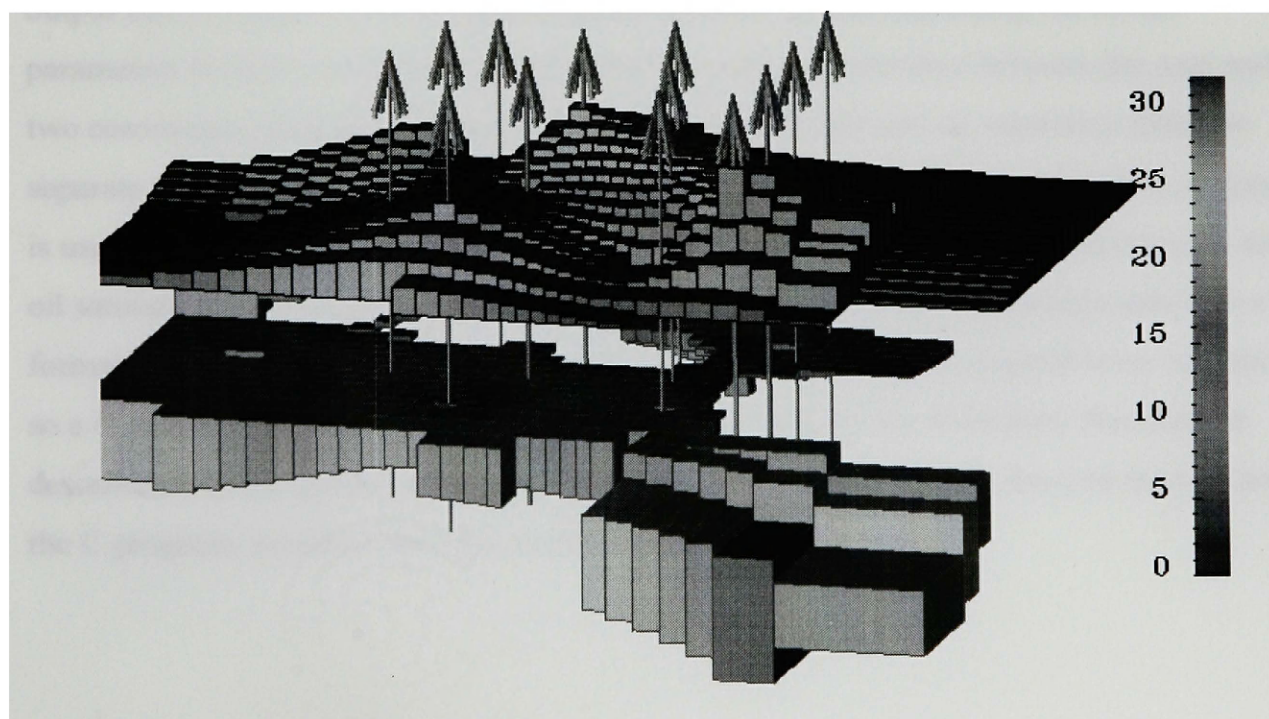


Figure 5-1: NE Rabbit Hills Reservoir Thickness.

with impacts being visually displayed to allow engineers to evaluate their effectiveness in recovering additional oil. Animation of various simulator outputs can thus give the petroleum scientists a more complete understanding of the complex physical processes that occur within the oil reservoir.

5.3 Preparing and Importing Reservoir Model Parameters

The reservoir simulation input parameters are contained in a single data file that is readily described by the general array importer. The model parameters are imported into DX as 2-D cell centered data on an irregular grid. The data are dependent on connections in the DX data model, meaning that the reservoir model parameter values are constant within a cell that is bounded by connections between points specified on the grid. Position dependent data are valid at node points; values between nodes are interpolated.

The output data from the simulator cannot be directly imported into DX. The SimMT output file is irregular with an unpredictable amount of data separating the model parameters at each time step. A shell script is used as an interface between the user and two conversion programs that output the reservoir pressure and oil saturation data into separate formatted data files. Within the shell script, an **awk** (Dougherty, 1992) program is used to identify each time series by finding a key phrase at the start of the pressure and oil saturation data. The **awk** program outputs all the pressure and saturation data into a formatted file. Unfortunately, this file cannot be described by the general array importer, so a C program reads this file and saves the data to disk as two formatted files that are describable by the general array importer. The **awk** output can be redirected directly into the C program, however the time performance remains the same.

5.4 Visualization Analysis

The first step in the visualization analysis is to display the various reservoir model input parameters. The Rubbersheet module is used in the visual analyses to transform the 2-D model cell data into 3-D blocks. The reservoir thickness is input into the Rubbersheet module to produce blocks of the correct thickness. Positioning the blocks in their correct location in the subsurface requires a custom module, RSPosition, which uses the reservoir top elevations to reposition the thickness blocks. Figure 5-2 shows the two layers of blocks correctly positioned in the subsurface. Because most of the reservoir cells are hidden from view, interactive controls are provided to view into the model's interior (Figure 5-3). Both of these figures vertically exaggerate the reservoir's structure and thickness because the reservoir is relatively flat and thin. The complete visualization shows an animation of the two layers separating, resulting in the display in Figure 5-1, which shows the layers separated so they can be more readily studied. This animation helps the audience understand that the two reservoir layers are connected, and then

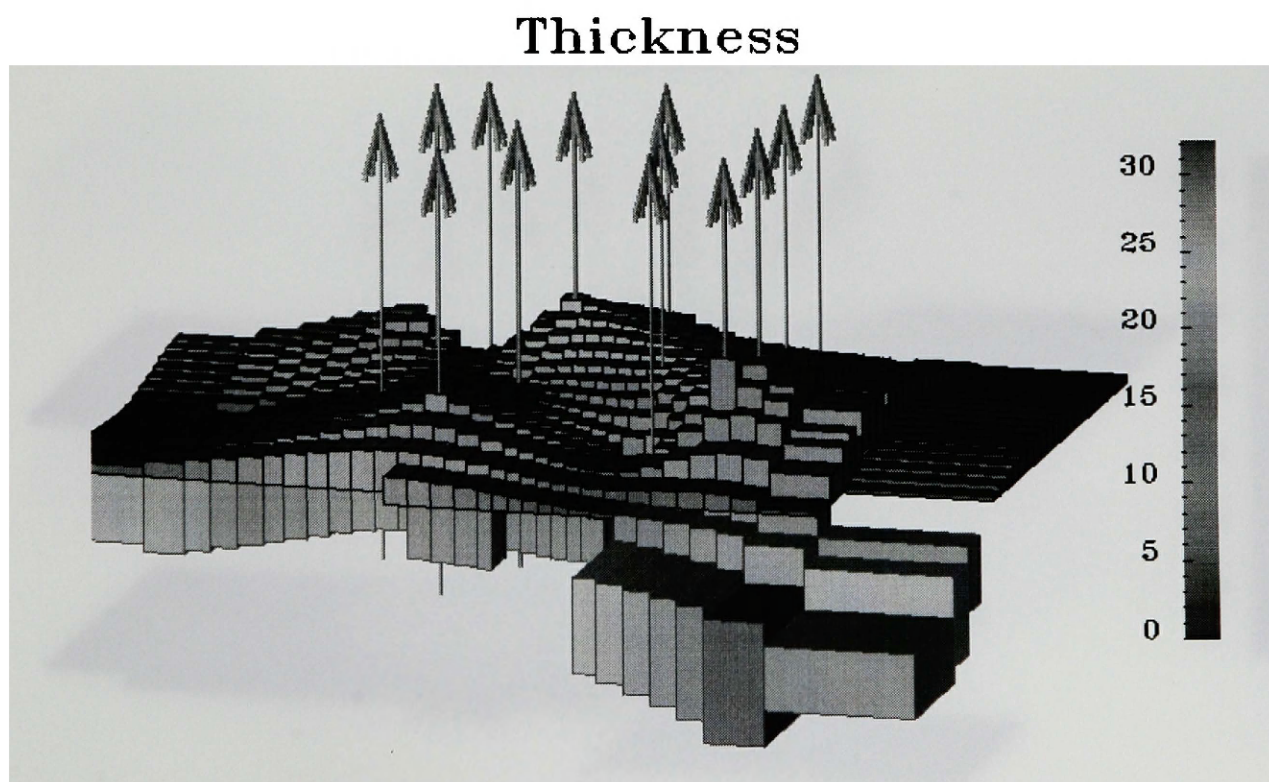


Figure 5-2: NE Rabbit Hills reservoir structure and thickness parameters.

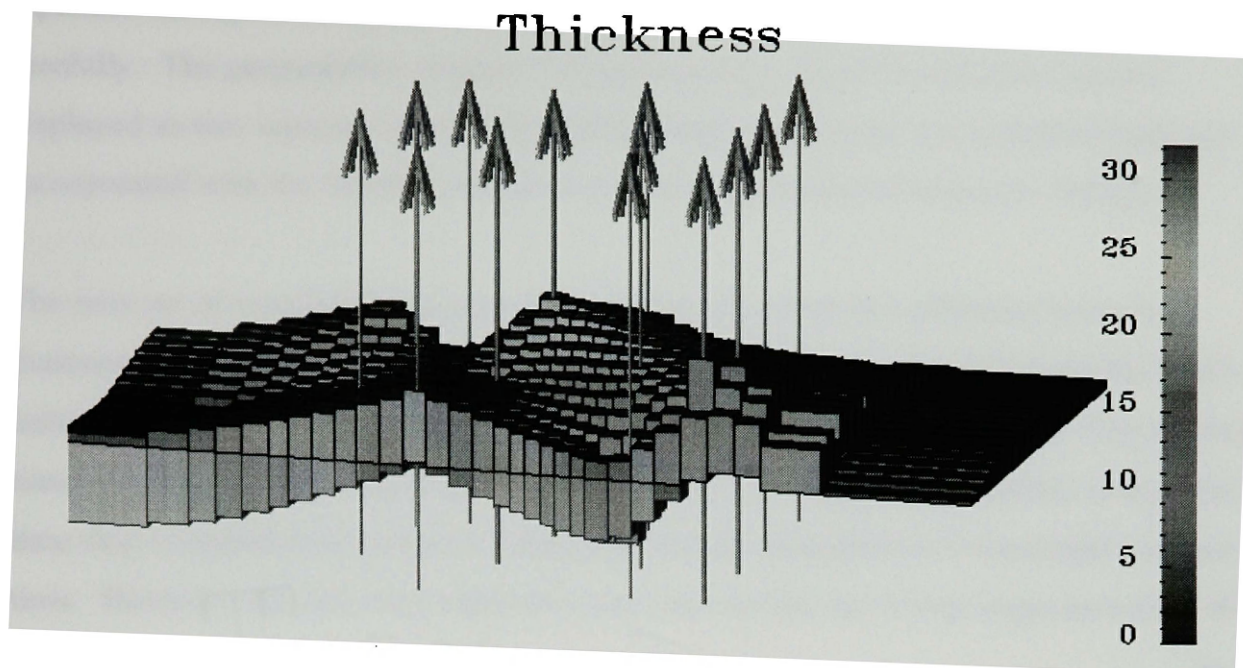


Figure 5-3: Reservoir structure and thickness model with 3000 ft removed.

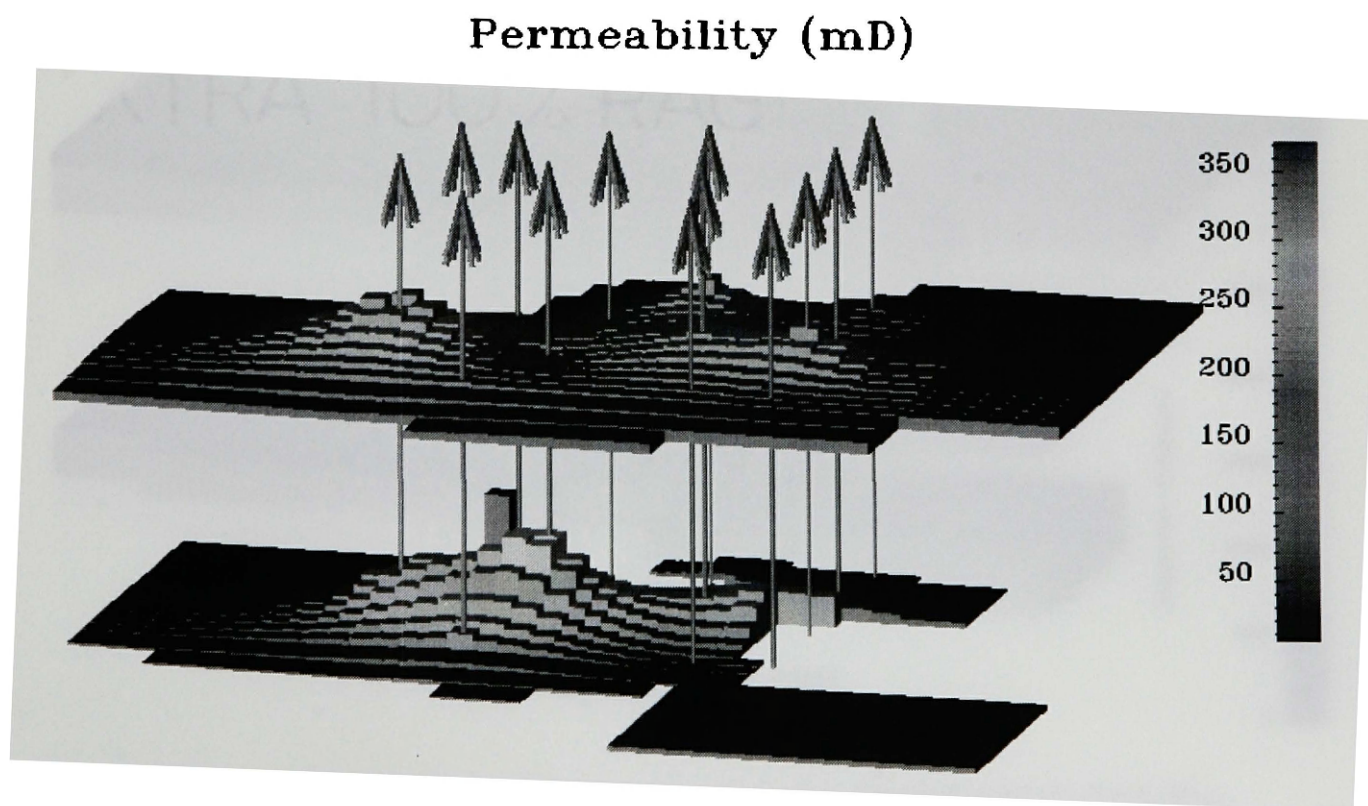


Figure 5-4: Reservoir Permeability.

separates the layers so that their individual model parameters can be studied more carefully. The permeability (Figure 5-4) and porosity values for the cells are also displayed as two separate layers. The initial reservoir pressure and saturation data are incorporated with the SimMT output to produce animations of reservoir changes.

The next set of visualizations come from output data from five different reservoir treatment scenarios. The base case scenario consists of 311 time steps from the field's initial production in 1973 to the field's predicted final production in 2005. The uneven time step interval varies from one day to one year. For realtime animation, every tenth time step is rendered to produce a relatively smooth animation in a reasonable amount of time. However, the time step interval is one year for the last 9 time steps between 1997

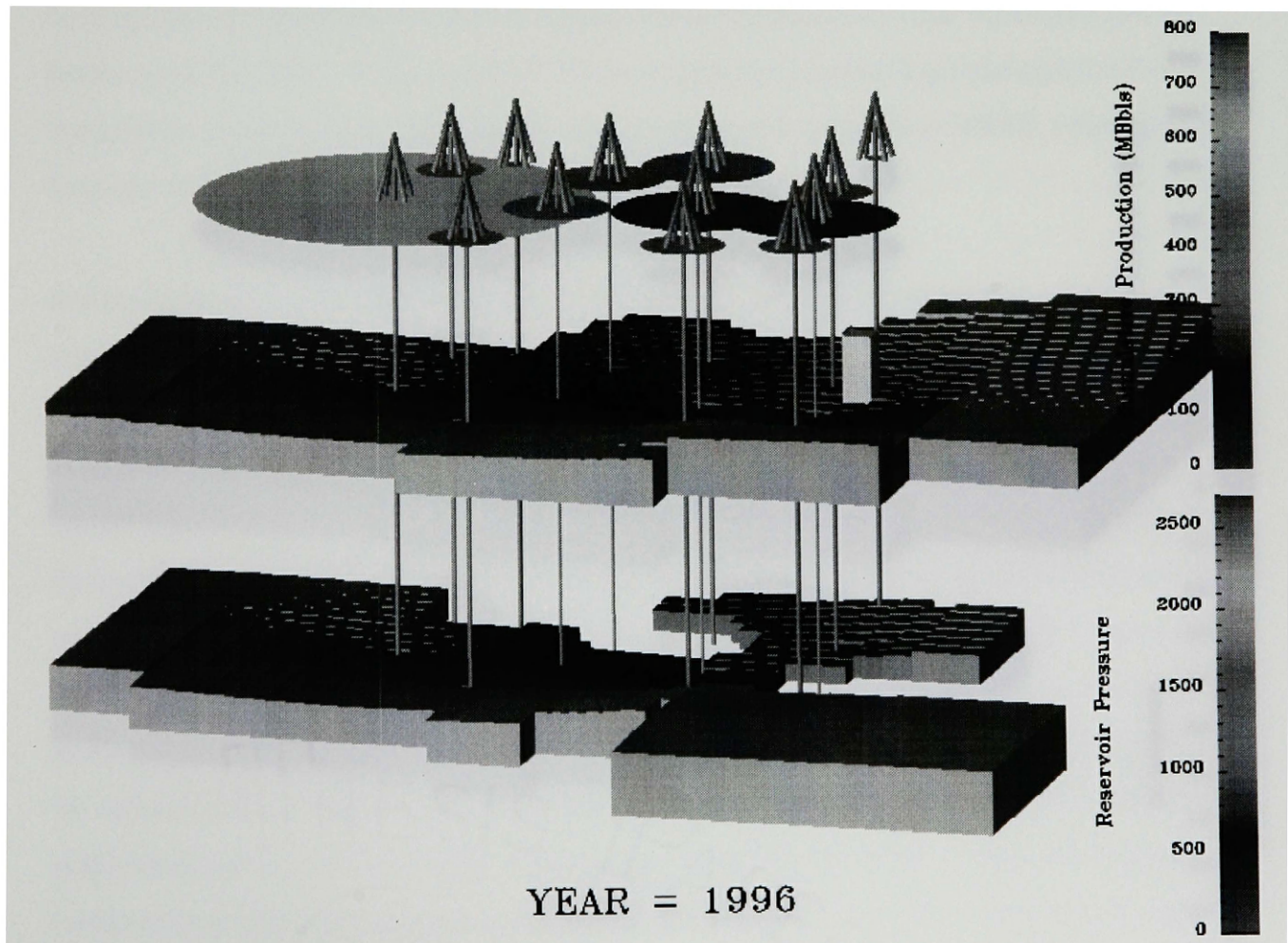


Figure 5-5: Simulated reservoir pressure with cumulative production data.

and 2005, which results in a large 8 year skip in the realtime animation. To closely examine this time period, the visual program must be adjusted to display every time step. Figures 5-5 and 5-6 show the pressure and saturation data in two layers for a 1996 time step. In addition to the well derrick at the top of each well, the cumulative production data presented in Chapter 4 are displayed at the top of each well.

Figure 5-5 also illustrates how visualization can be used to check model specifications. Note that a single model cell stands out in mid-center with a very high pressure. This cell also remains at a constant pressure throughout the simulation which is highly suspect behavior. It turns out that this anomalous behavior is caused by a reservoir simulator error that previously had gone undetected.

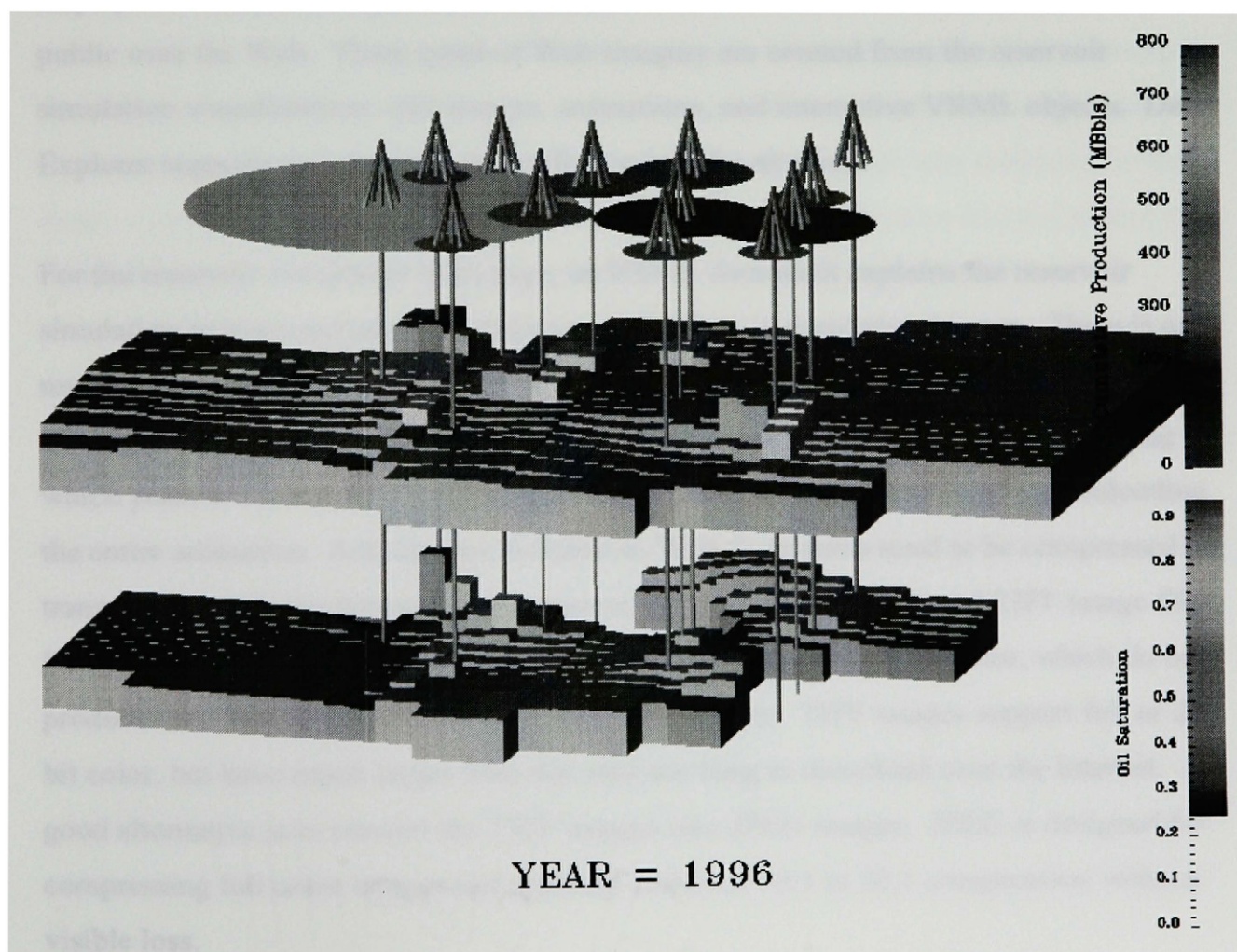


Figure 5-6: Reservoir oil saturation with cumulative production data.

The other four treatment scenarios have output for the time period between 1997 and 2005. Every time step is shown for animations of these different scenarios. One scenario considers the conversion of one producing well to a water injection well. Another scenario considers the conversion of five producing wells to water injection wells. The other two scenarios convert these same wells to polymer injection wells. Wells that are converted to injection wells are colored blue in the visualizations.

5.5 Visualization Output Requirements for Web Site Imagery

Besides being used to provide modeling support, visualization is used to create visual displays for disseminating reservoir characterization results to other scientists and the public over the Web. Three types of Web imagery are created from the reservoir simulation visualizations: still images, animations, and interactive VRML objects. Data Explorer supports compressed output file formats for all three.

For the reservoir simulation Web page, an HTML document explains the reservoir simulation process and provides links to a collection of simulation images. There is one reservoir pressure image and one reservoir oil saturation image for each year of the simulation. The HTML document provides the viewer greater flexibility of selecting which years to view and the resulting download time is much quicker than downloading the entire animation. Still images included in Web documents need to be compressed for transmission over the Internet. DX supports the production of GIF and TIFF image files for incorporation into Web sites. Compressed GIF images use 8-bit color, which do not produce very satisfactory displays for full color images. TIFF images support full or 24-bit color, but have much larger files that take too long to download over the Internet. A good alternative is to convert the TIFF images into JPEG images. JPEG is designed for compressing full color images and typically achieves 10:1 to 20:1 compression without visible loss.

In addition to still images, animations of the reservoir pressure and saturation data are included on the Rabbit Hills Web site. Animations are displayed over the Web as either MPEG movies or Quicktime movies. To create MPEG movies, the DX animations are saved in the MIFF format using the Image window dialog box. The MIFF format can be converted to MPEG movie using ImageMagick software. The disadvantage of the MPEG format is that it does not support audio. Quicktime movies include audio and are created using the same process discussed in Chapter 4 to create video animations.

An additional consideration that must be addressed in producing animations from the reservoir simulator is the irregular time interval between images. All 311 time steps were output as TIFF images from DX; however, some were not used in producing the animation. Approximately 12 TIFF images per year are used in the animation, with the extra TIFF files deleted. The TIFF images are repeated for approximately five frames in the animation, so there is approximately two seconds of the animation for each year. For the last 9 years from 1996 to 2005, only one TIFF image per year was produced, so the single image is displayed for the entire two seconds of the animation devoted to that year. Thus, the animated sequence appears rather smooth until the final 9 years.

All the reservoir model cells are not visible in the GIF or JPEG images (Figures 5-1 through 5-4). Instead of displaying a still image, an interactive 3-D object of the reservoir model can be transmitted over the Web using Virtual Reality Modeling Language (VRML). VRML objects are saved within the DX visual program with the Export module. A viewer must have a VRML capable browser to view the VRML object. With the VRML depiction, the user can manipulate (zoom, rotate, pan, etc.) it to better examine model parameters of interest.

Chapter 6: Discussion and Conclusions

As presented in this paper, the visualization process is a straightforward, five step process. In practice, the order of the five steps may vary, the process usually involves several iterations, and other complications emerge. The visualization goals may change as the visual analysis progresses. It may turn out to be quite time consuming to import the data into the visualization system. Other difficulties arise from combining data sets defined on different coordinate systems. These problems and real world complications are alluded to in the previous chapters, but summarized below for clarity.

6.1 Visualization Process Revisited

The order of the visualization steps, especially the first three steps, varies depending on the problem. After the data are collected, the scientist may decide to import them into the visualization system before identifying a visualization goal. This allows the scientist to confirm that the data can be described and imported into the visualization system before defining specific goals. The scientist can then use a simple visual analysis to preview the data, which allows the scientist to gain additional insight into the data and possibly some initial ideas for a visualization goal. An alternative ordering also occurs when the scientist starts with a visualization goal, but no data. The scientist must determine how to create or collect the data necessary to satisfy the visualization goal.

Keller and Keller (1993) point out that determining the ultimate visualization goal is often evolutionary. The visualization process usually involves many trial-and-error data representations and image adjustments during the visual analysis step. During these iterations, new images are compared to previous images, and the scientist must determine which image reveals more information about the data. A new visualization goal may result from these comparisons. The iterative process of visual analysis should involve

other scientists or team members whenever possible. Other individuals provide additional insight into the data, identify problems with the visualizations, and offer suggestions on how to improve the visualization. These suggestions often mean changing the visualization goal.

For a cross discipline research team, like a reservoir characterization team, it is important that the scientists work closely with the visualization specialist. Simply giving the data to the visualization expert will not result in a good final product. Ideally, the scientist should provide a general visualization goal along with the data. During the visual analysis the visualization specialist needs to get feedback from the other scientists on the visual images that are being produced. Then visualization changes can be made to satisfy all individuals involved.

Preparing and importing the data are often the most time consuming tasks. The data often need to be reformatted into a form that can be described by the DX data model. This requires a good understanding of the data model and the ability to write conversion programs to transform the data. For example, the production data are contained in a spreadsheet, with each well identified by an API number. The well data need positions to be of any use in DX. A conversion program assigns positions to the well data based on its API number.

This brings up another problem, that of determining which coordinate system to use to define the data positions. The various NE Rabbit Hills data sets are defined on four different coordinate systems. The wells were originally located by references to the nearest section lines. Earlier visualization work done by Holbrook (1996) converted these well locations to an arbitrary coordinate system with an origin located to the southwest of the NE Rabbit Hills field. The 3-D seismic data are defined on a real world coordinate system. These data include surveyed elevation locations that are used in producing images of the earth's surface. The reservoir simulation model is defined on another arbitrary coordinate system, with the x-axis running in the north-south direction.

Wells are referenced in this coordinate system by cell location, so wells are assigned a position in the center of the cell for visualization work. Combining the different data sets requires transforming the data positions into one coordinate system. However, the exact relationship between the different coordinate systems is not known, so that the conversion must be approximated. For the reservoir pressure and oil saturation visualizations, initially the cumulative production disks did not line up with the well derricks and well bores. The well locations defined in the reservoir model coordinate system could not be used, instead the real world well locations are used and converted to the reservoir model coordinate system. There is some error in the conversion (less than 100 feet), but it is not noticeable to the viewer. Ideally all data should be collected in the same coordinate system.

6.2 Conclusions and Directions for Future Research

The multi-step visualization process presented in this paper is an effective way of creating visual displays from various petroleum data sets. The examples discussed in detail in Chapters 3, 4, and 5 illustrate how this process can be applied to petroleum data sets, such as geologic formation tops, production data, and reservoir simulation model data. This work illustrates both the advantages of visualization, and many of the practical problems which must be addressed.

For example, geologic formation tops can be transformed into a 3-D rendering of the subsurface, which allows the subsurface image to be interactively examined within the visualization system. However, interactive performance is greatly enhanced when the visualization system is run on a computer that supports hardware rendering, like an SGI Octane. Context data can be used to create a display which enhances the display's presentation. However, as illustrated in the NE Rabbit Hills field, if the area does not exhibit much vertical relief, modifications must be made to the SurfacePlot macro to

vertically exaggerate the subsurface structure. Different visualization adjustments are required to depict areas of complex geologic structures or very high relief.

The display of oil production data illustrates how rendering artifacts can also affect the imagery. Figures 4-4 and 4-5 show well bores that exhibit different diameters, caused by image aliasing. An AntiAliasing macro available at the IBM DX Web site (<http://eagle.almaden.ibm.com/dx>) can be used to create images containing well bores with a constant diameter. The problem with this macro is that the section lines become dimmer, the surface topography colors are altered, and other changes to the captions and color bars must be made.

The production display also demonstrates the potential advantage/disadvantage of visual models that are simple, but not in accordance with physical phenomena. Cylindrical disks used to represent each well's drainage area have been praised as an effective means for representing cumulative production, but also criticized as an over simplification of reservoir drainage. The reality is that there is insufficient data to show exact drainage areas. These disks show an approximated area of the reservoir that has been affected by each producing well, and the extent to which the approximation is judged as appropriate depends largely on the viewer. Finding a better approximation that is consistent with data, visually appealing, and efficiently computable is a long range goal. A similar, but even more simple visual improvement involves the simple well derrick glyph used to represent the well's surface location. Some viewers do not recognize this glyph as a well derrick. A pumping jack glyph needs to be created to replace this well derrick glyph in future visualizations. Another proposed modification would allow the viewer to interact with the visualization by selecting a well with the mouse cursor. The visual program would then incorporate the well name and current production total into the display.

Finally, computer visualization performance needs to be and can be enhanced with several modifications to the existing visual programs. When time series data are used in realtime computer animations that use every 10th or 12th time step, it is beneficial to

import only one series member at a time. This reduces the amount of computer memory used by DX and reduces the time to read the series data from disk. Significant time performance can also be improved by using Run Time Loadable custom modules instead of Outboard custom modules. Outboard modules are less efficient because they require the DX system to create a separate process to load and run the executable of the custom module. The net result of these and other similar changes would be to significantly enhance the ability to produce better realtime animations and interactive visualizations.

APPENDIX A
CONVERSION PROGRAMS

```
/* This program takes monthly production data (rhne2.txt) exported from
   EXCEL and outputs a SERIES of cumulative production data for each
   month (from 1/72 to 12/96)
```

```
To run: a.out <rhne2.txt>test.data
        mv test.data production.data
```

```
Output is only for those wells that had production.
```

```
*/
```

```
#include <stdio.h>
#include <stdlib.h>
#define NYEARS 25
#define NWELLS 22
#define NDATA 1095

main()
{
    int wellid, nwells=0, i, j, k, l, m;
    int prod[NWELLS][NYEARS][12], apinum[NWELLS];
    int x[NWELLS], y[NWELLS], total[NWELLS];
    int napis, apiloc[25], apix[25], apiy[25];
    char junk[25];
    FILE *ifp1;

    for (i=0; i<NDATA; i++){
        scanf("%d %d %d %d", &j, &k, &l, &m);
        gets(junk);
        if (j!=wellid){
            wellid=j;
            nwells++;
        }
        prod[nwells-1][k-72][l-1]=m;
        apinum[nwells-1]=j;
    }
}
```

```
/* READS well locations for api numbers from file loc.api */
```

```
ifp1 = fopen("spudlocn.api", "r");
fscanf(ifp1, "%d", &napis);
for (j=0; j<napis; j++){
    fscanf(ifp1, "%d %d %d", &apiloc[j], &apix[j], &apiy[j]);
    fgets(junk, 40, ifp1);
}
fclose(ifp1);
```

```
/* Finds location by matching api numbers */
```

```
for (j=0; j<nwells; j++){
    for (i=0; i<napis; i++){
        if (apinum[j]==apiloc[i]){
            x[j]=apix[i];
            y[j]=apiy[i];
        }
    }
}
```

```
/* Generates reformatted output*/
```

```
for (k=0; k<NYEARS; k++){
    for (l=0; l<12; l++){
        for (j=0; j<nwells; j++){
            total[j]+=prod[j][k][l];
            printf("%7d %7d %2d %2d %8d\n", x[j], y[j], k+72, l+1, total[j]);
        }
    }
}
```

```
#!/usr/bin/sh
#Shell script convert.all to convert output from SimMT to DX data files
#Converts pressure and oil data
#Prompts user for number of rows, columns and layers
```

```
echo "Enter input file: "
read filename
```

```
echo "Enter number of rows"
read r
```

```
echo "Enter number of columns"
read c
```

```
echo "Enter number of layers"
read l
```

```
awk -f /eis3/braun/production/Awk/convall.awk -v col=$c -v row=$r -v
layer=$l $filename > all.out
grep days all.out|wc|awk '{print $1}' >junk
```

```
/eis3/braun/production/C_programs/convall
```

```
echo "Enter pressure output file #1 name:"
read filename
mv prs1.out $filename
```

```
echo "Enter pressure output file #2 name:"
read filename
mv prs2.out $filename
```

```
echo "Enter oil saturation output file #1 name:"
read filename
mv oil1.out $filename
```

```
echo "Enter oil saturation output file #2 name:"
```

```
read filename
mv oil2.out $filename
```

```
rm junk
rm all.out
```

```
#convall.awk
#Used by convert.all to search for pressure and oil saturation data in
#SimMT output files. The number of rows, columns, and layers must be
#specified
#by the user.
```

```
BEGIN {
    printf "%d\n",col
    printf "%d\n",row
    status = getline
    while (status == 1) {
        if ($3 == "Elapsed") printf "%s %s\n", $6, $7
        if ($1 == "Reservoir"){
            getline
            k=0
            while(k++<col/15){
                c=(k-1)*15
                l=0
                while(l++<layer){
                    i=0
                    while(i++<6) getline
                    i=0
                    while(i++<row){
                        getline
                        j=c
                        while(j++<col && j<c+16){
                            printf "%5d ",$(j-c+1)
                        }
                        printf "\n"
                    }
                }
            }
        }
    }
}
```

```
if ($1 == "Oil" && $2 == "Saturation"){
    getline
}
```

```
k=0
while(k++<col/15){
    c=(k-1)*15
    l=0
    while(l++<layer){
        i=0
        while(i++<6) getline
        i=0
        while(i++<row){
            getline
            j=c
            while(j++<col && j<c+16){
                printf "%s ",$(j-c+1)
            }
            printf "\n"
        }
    }
}

status = getline
}
```

```
/* convall.c
```

```
    This program converts pressure and saturation data from awk program
    output
    file "all.out" for use in DX*/
```

```
#include <stdio.h>
#include <stdlib.h>
```

```
#define R 50                /*Maximum number of rows & cols in
SimMT*/
#define C 50
```

```
main()
```

```
{
    int i,j,k,l,row,col,c1,c2,steps;
    int a[R][C];            /*Pressure data for 2 layers*/
    int b[R][C];
    float f[R][C];          /*Oil saturation data*/
    float g[R][C];
    float days;
    FILE *ifp, *ofp1, *ofp2, *ofp3, *ofp4, *ofp5;
    char filename[60], junk[12];
```

```
    ifp=fopen("junk","r");
    fscanf(ifp,"%d",&steps);    /*Number of time steps*/
    fclose(ifp);
```

```
    ifp=fopen("all.out","r");
    fscanf(ifp,"%d\n%d",&col,&row);
```

```
    ofp1=fopen("prs1.out","w");
    ofp2=fopen("prs2.out","w");
    ofp3=fopen("oil1.out","w");
    ofp4=fopen("oil2.out","w");
    ofp5=fopen("days.out","w");
```

```
/*Loops through each time series of the data*/
```

```
for(i=0;i<steps;i++){
    fscanf(ifp,"%f%s",&days,filename);
    fprintf(ofp1,"%f days\n",days);
    fprintf(ofp2,"%f days\n",days);
    fprintf(ofp3,"%f days\n",days);
    fprintf(ofp4,"%f days\n",days);
    fprintf(ofp5,"%f\n",days);
```

```
    c1=0;                /*Converts pressure data*/
```

```
    c2=15;
    for(l=0;l<col/15+1;l++){    /*Layers are intermeshed*/
        for(j=0;j<row;j++)
            for(k=c1;k<c2 && k<col;k++) fscanf(ifp,"%d",&a[j][k]);
        for(j=0;j<row;j++)
            for(k=c1;k<c2 && k<col;k++) fscanf(ifp,"%d",&b[j][k]);
        c1=c2;
        c2=c2+15;
    }
```

```
    for(j=0;j<row;j++){
        for(k=0;k<col;k++) fprintf(ofp1,"%4d ",a[j][k]);
        fprintf(ofp1,"\n");
    }
    for(j=0;j<row;j++){
        for(k=0;k<col;k++) fprintf(ofp2,"%4d ",b[j][k]);
        fprintf(ofp2,"\n");
    }
```

```
    c1=0;                /*Converts oil saturation data*/
```

```
    c2=15;
    for(l=0;l<col/15+1;l++){
        for(j=0;j<row;j++)
            for(k=c1;k<c2 && k<col;k++) fscanf(ifp,"%f",&f[j][k]);
        for(j=0;j<row;j++)
            for(k=c1;k<c2 && k<col;k++) fscanf(ifp,"%f",&g[j][k]);
```

```

c1=c2;
c2=c2+15;
}

for(j=0;j<row;j++){
    for(k=0;k<col;k++) fprintf(ofp3,"%5.3f ",f[j][k]);
    fprintf(ofp3,"\n");
}
for(j=0;j<row;j++){
    for(k=0;k<col;k++) fprintf(ofp4,"%5.3f ",g[j][k]);
    fprintf(ofp4,"\n");
}
}
fclose(ifp);
fclose(ofp1);
fclose(ofp2);
fclose(ofp3);
fclose(ofp4);
fclose(ofp5);
}

```

```

#!/usr/bin/sh
#Shell script autoconv.all to convert output from SimMT to DX data files
#Converts pressure and oil data
#Automatically read number of rows, columns and layers from input file
#Can only handle 2 layers in current form (number of output files need
#to be increased.

```

```

echo "Enter input file: "
read filename
awk -f /eis3/braun/Production/Awk/autoconv.awk $filename > all.out
grep days all.out|wc|awk '{print $1}' >junk

```

```

/eis3/braun/Production/C_programs/convall

```

```

echo "Enter pressure output file #1 name:"
read filename
mv prs1.out $filename

```

```

echo "Enter pressure output file #2 name:"
read filename
mv prs2.out $filename

```

```

echo "Enter oil saturation output file #1 name:"
read filename
mv oil1.out $filename

```

```

echo "Enter oil saturation output file #2 name:"
read filename
mv oil2.out $filename

```

```

rm junk
rm all.out

```

```
#autoconv.awk
#Used by autoconv.all to search for pressure and oil saturation data in
#SimMT output files. The number of rows, columns, and layers must be
specified
#in the output file.
```

```
BEGIN {
    status = getline
    while (status == 1) {
        if ($5 == "Columns") {col=$7;printf "%d\n",$7}
        if ($5 == "Rows") {row=$7;printf "%d\n",$7}
        if ($5 == "Layers") {layer=$7}
        if ($3 == "Elapsed") printf "%s %s\n",$6,$7
        if ($1 == "Reservoir"){
            getline
            k=0
            while(k++<col/15){
                c=(k-1)*15
                l=0
                while(l++<layer){
                    i=0
                    while(i++<6) getline
                        i=0
                        while(i++<row){
                            getline
                            j=c
                            while(j++<col && j<c+16){
                                printf "%5d ",$(j-c+1)
                            }
                            printf "\n"
                        }
                    }
                }
            }
        }
    }
}
```

```
if ($1 == "Oil" && $2 == "Saturation"){
```

```
    getline
    k=0
    while(k++<col/15){
        c=(k-1)*15
        l=0
        while(l++<layer){
            i=0
            while(i++<6) getline
            i=0
            while(i++<row){
                getline
                j=c
                while(j++<col && j<c+16){
                    printf "%s ",$(j-c+1)
                }
                printf "\n"
            }
        }
    }
    status = getline
}
```


Bibliography

- Abram, G. and L. A. Treinish, 1996, An Extended Data-Flow Architecture for Data Analysis and Visualization, in *Proceedings of the 1996 Data Explorer Symposium*.
- Ahmed, T., 1993, An In-House Computer Simulator, Department of Petroleum Engineering, Montana Tech of the University of Montana, Butte, Montana, 1993.
- Ahmed, T., C. A. Link, K. W. Porter, C. J. Wideman, M. Ziaja, P. Himmer, S. Corrigan, J. Putnam, and J. Braun, 1997, Integrated Reservoir Analysis Methodology - A case History: The NE Rabbit Hills Field, North-central Montana, in *Proceedings of the 48th Annual Technical Meeting of the Petroleum Society of the Canadian Institute of Mining, Metallurgy & Petroleum*, June, 1997, paper no. 97-29.
- Braun, J. A., 1995, Flag Module, <http://www.cs.umt.edu/DX>.
- Brodlie, K.W., L.A. Carpenter, R.A. Earnshaw, J.R. Gallop, R.J. Hubbard, A.M. Mumford, C.D. Osland, and P. Quarendon, 1992, *Scientific Visualization: Techniques and Applications*, Springer-Verlag, 284p.
- Dougherty, D., 1992, *Sed and Awk*, O'Reilly and Associates, 397p.
- Earnshaw, R.A., and N. Wiseman, 1992, *An Introductory Guide to Scientific Visualization*, Springer-Verlag, 156p.
- GQS, 1989, *IES User's Manual*, Geoquest Systems, Inc.
- George, K. G., 1994, A Reservoir Simulation Study of the N. E. Rabbit Hills Field, Montana, M.Sc. thesis, Montana Tech of the University of Montana.
- Haber, R. B., and D. A. McNabb, 1990, Visualization Idioms: A Conceptual Model for Scientific Visualization Systems, in *Visualization in Scientific Computing*, ed. G. M. Nielson, B. Shriver, and L. J. Rosenblum, pp. 74-93.
- Holbrook, S. L., 1996, *Data Visualization Tools for the Production of Data Visual-Animations*, M.Sc. thesis, University of Montana.
- Huang, P.Y., 1996, Reservoir Visualization with IBM Data Explorer, in *Proceedings of the 1996 Data Explorer Symposium*.
- IBM, 1997, *Data Explorer Training Manual*, <http://www.cs.umt.edu/DX/DXLECT>.
- IBM, 1995a, *Visualization Data Explorer User's Guide*.

- IBM, 1995b, Visualization Data Explorer Programmer's Reference.
- Keller, P. R., and M. M. Keller, 1993, Visual Cues - Practical Data Visualization, IEEE Press, 229p.
- Landmark, 1997, Seisworks version 3.0, <http://www.lgc.com>.
- Moeller, K., 1997, Porting Graphical User Interfaces from X/Motif to Microsoft Windows, M.Sc. thesis, University of Montana.
- Olson, T. M., J. A. Babcock, K. V. K. Prasad, S. D. Boughton, P. D. Wagner, M. H. Franklin, and K. A. Thompson, 1997, Reservoir Characterization of the Giant Hugoton Gas Field, Kansas, *AAPG Bulletin*, v. 81, pp.1785-1803.
- Porter, K. W., 1997, Geologic Data in a Reservoir Characterization, Sawtooth Formation, NE Rabbit Hills field, Blaine County, Montana, in press *Montana Bureau of Mines and Geology Open-File Report 354*.
- Schlumberger, 1997, Eclipse 100 Reference Manual.
- Slatt, R. M., M. R. Thomasson, P. R. Romig, E. S. Pasternack, A. Boulanger, R. N. Anderson, and H. R. Nelson, 1996, Visualization Technology for the Oil and Gas Industry: Today and Tomorrow, *AAPG Bulletin*, v. 80, pp. 453-459.
- Tearpock, D. J., and R. E. Bischke, 1991, Applied Subsurface Geologic Mapping, Prentice Hall, 648p.
- Tinker, S. W., 1996, Building the 3-D Jigsaw Puzzle: Applications of Sequence Stratigraphy to 3-D Reservoir Characterization, Permian Basin, *AAPG Bulletin*, v.80, pp. 460-485.
- Thompson, D., 1998, Design and Implementation of a Digital Multimedia Lab and Digital Media Optimizing Techniques, M.Sc. thesis in preparation, University of Montana.