Visual interpretation of multidimensional computations and transistor design

by E. J. Farrell
P. A. Appino
D. P. Foty
T. D. Linton, Jr.

As digital simulations and computations become more complex, larger volumes of output are generated; the engineer must select a concise method of displaying the output and extracting relevant information. In this paper we describe an experimental system called the Visual Interpretation System (VIS), which provides a wide range of tools for managing the visualization of simulation data. The effectiveness of VIS results from an interactive interface which controls a database manager and a visualization manager. The database consists of entities called *data sets* that carry a complete description of the geometry and time-dependent behavior of various properties of a simulated physical object. Visualization management involves both 2D and 3D imaging in multiple display windows and animation. Three-dimensional data imaging is based on optical modeling with back-to-front perspective projection. The optical model assigns color and attenuation to each point on the basis of its data value. With the appropriate choice of attenuation and color, the user can display multiple 3D regions, either as solids or transparently. This approach is not based on surfaces, nor does it require the data to have spatial continuity. The usefulness of VIS is demonstrated with data from a large-scale simulation of a transistor. We demonstrate how 3D visualization techniques provide insight into the physics of isolation-trench-bounded devices at both room and low temperatures, which facilitates the development of improved designs.

1. Introduction

Complex computations hold the promise of new discoveries and improved technology. However, engineers and scientists are often confronted with the difficult task of interpreting the interrelationships between several variables in a three-dimensional region.

Tools are required in order to visualize the results and fully utilize the results of costly computations [1–4]. Visualization requirements occur in postprocessing and concurrent processing. Visualization for postprocessing involves interpreting results of a completed computation. Concurrent visualization involves interpreting intermediate results in a long computation; the goal is to evaluate convergence and possibly adapt the computations to improve accuracy or convergence.

A system for visual data interpretation should address format conversion, data storage and retrieval, the user interface, visualization methods, and capture of results on film or videotape. Frequently, data *format conversion* is required to convert floating-point numbers generated at irregular mesh points to integers in a rectilinear array. To ensure interactive speeds, it is important to consider *data storage and retrieval.* Typically, the output of a large-scale computation/simulation involves several variables; each variable may involve two to ten megabytes. The output is a series of 2D or 3D arrays corresponding to variable values in a 3D region; array dimensions are usually around one or two hundred data points. Even though the arrays are small, they may be the output of computer runs taking several thousand CPU minutes. The *user interface* and *display methods* are key aspects of a visualization system, and are the subjects of this paper. *Capturing* the results of visualization on film or video allows the user to present results remote from the display, and allows several scientists to review results with a common visual representation of the data. The authors have implemented a video system to generate tapes based on a simple command script [5].

Traditional display methods based on contour plots, mesh surfaces, or tiled structures are not adequate for visualizing complex multiple-variable data, and a surface-based model of 3D data arrays does not have the capacity needed for visual interpretation. Data interpretation is more complex than everyday viewing of surfaces, and volume visualization is required. The user may wish to visualize a nebulous 3D region, data without distinct surfaces or structures, or a region buried inside a larger structure. Also, often a vector field must be displayed, which may involve sources and drains. To be useful, these techniques need to be part of a software system which is convenient to use. The system must provide visualization tools in terms of the engineer's application, not graphics jargon.

Volume visualization is the major display requirement of many data interpretation systems. The volume-rendering algorithm presented by Drebin is based on probabilistic material classification; each voxel (three-dimensional cubic element representing a data point in three-dimensional space, or "volume pixel") is assigned material percentages [6]. Color, attenuation, and density are associated with each material, and are the basis for imaging translucent regions or solids with surface reflection. This approach has been used effectively for 3D medical imaging [7]. The approach presented by Levoy is based on "soft classification" of surfaces, i.e., a continuous variation of attenuation [8]. Surface artifacts are reduced with soft classification. Surface shading is based on light direction and the viewer's line of sight relative to the normal of the surface. The V-BUFFER [9] method presented by Upson is a general visualization approach for small 3D arrays. Vertices of each data cell are mapped into color, attenuation, and texture, and then interpolated and projected onto appropriate image pixels. This generalized cell approach has the advantage that a nonuniform mesh can be imaged. Schlusselberg [10, 11] uses a similar approach based on ray tracing for medical imaging. Prior work [12–14] by Farrell, Zappulla, Christides, and Appino demonstrates how an optical-model technique provides an effective tool for volume visualization with a range of interactive options by simply changing a color map.

The objective of this presentation is to describe a system for visual data interpretation which has been used successfully in several application areas. In the following section, the Visual Interpretation System (VIS) is described. In the third section, the volume visualization methods used in VIS are presented. In Section 4, the utility of VIS is demonstrated by using it to visualize and interpret results from a large-scale simulation of a transistor. The paper concludes with a discussion and references. (System hardware is described in the Appendix.)

2. Visual Interpretation System

The Visual Interpretation System (VIS) is a tool for understanding complex multidimensional data sets. Scientists and engineers are the intended users of the system. The design goals of VIS are the following:

- *Ease of use.* The process of creating a useful image from multidimensional data is inherently complex. Many tasks and parameters are required, including data selection, type of visualization, color, amount of transparency, cutout, spatial orientation, and scale. Without a flexible, consistent, and convenient user interface, the scientist/engineer is quickly overwhelmed. User interface features that enhance ease of use are multiple windows, a close matching of user tasks to window design, and the ability to manipulate data in its original application context.
- *Interactivity.* Visualization of data sets from the computational sciences is an iterative process where the "best" image is not known *a priori.* To be useful in this context, a visualization tool should let the scientist or

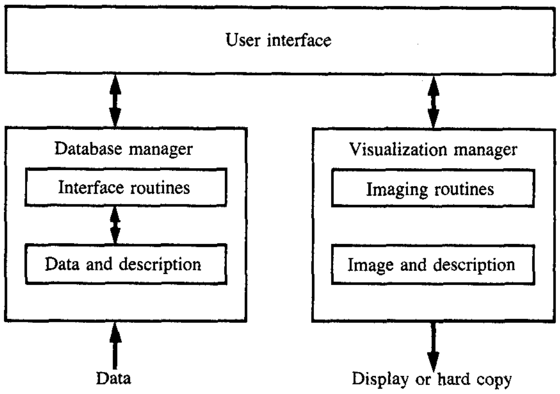


Figure 1

Components of the Visual Interpretation System.

engineer generate many different images from the same data set [15]. VIS emphasizes interactive visualization to create an environment for exploration and experimentation. Many visualizations can rapidly be created from variables within a data set or from related variables in different data sets. Photorealistic rendering takes a back seat to the need for interactivity.

- *Discipline-independence.* Although the computational sciences are a diverse collection of disciplines, there is general agreement that considerable common ground exists in their visual needs [16]. Many visualization techniques have applicability to a wide range of disciplines, which opens up the possibility of generic visualization. The requirements of discipline-independent visualization include the ability to handle data sets from disparate sources and the ability to visualize data in many different ways [17]. VIS addresses these requirements with a database manager and visualization manager, respectively. The database manager provides consistent methods for dealing with multidimensional data sets and separates data management from visualization. The visualization manager provides methods for creating 2D, 3D, and 4D (dynamic 3D) images of multidimensional data. Particular emphasis is placed on multivariant techniques [13] such as variable masking, classification, and derived variables.

A review of the steps involved in the data interpretation process is useful for the discussion to follow. The data interpretation or computational analysis cycle is composed of three processes: filtering, mapping, and rendering [16, 18, 19]. A filter takes data as input and produces data in another form. Examples of filtering operations include interpolation, smoothing, and histogram normalization. The mapping process converts data into geometric (points, lines, surfaces) or image data. Rendering creates an image from the data generated by the mapping process. The user must specify many parameters before a rendering routine can produce an image. Typical parameters needed by a renderer include color, transparency, shading, orientation, and placement.

VIS concentrates on volume mapping and rendering techniques. The scientist's data are assumed to be scalar values on a uniform grid. Any filtering required to reformat the data is not at present included in the system. Tasks faced by the user of VIS are selection and retrieval of data, selection of rendering routine, and specification of rendering parameters.

The main components of VIS are the database manager, user interface, and visualization manager (see **Figure 1**). The backbone of the system is the database manager, which isolates the details of data management from the rest of the system and provides routines for data definition and manipulation. Central to the usability and success of VIS is the user interface through which all system components are accessed. The visualization manager maps data to 2D, 3D, or 4D images, renders the images, and accesses a visualization database for saving and retrieving images. The following sections describe the database manager, user interface, and visualization manager in greater detail.

- *Database manager*

The primary goal of the VIS database manager is data-independence, i.e., consistent handling of data from a variety of sources. The VIS database contains original data plus a high-level data description creating a self-describing data format. A standardized data format facilitates writing generic visualization routines. The data description makes data interpretation easier because the scientist manipulates data with familiar names and physical units. Another benefit of the database manager is the separation of low-level data management activities from the rest of the system.

The idea of using descriptive information about data to achieve data-independence does not originate with the VIS database manager. Prior efforts such as the Block Data Set [20] and the FLATDBMS [21] used the same concept but were designed for programmers, not scientists. The VIS database manager is an implementation of the Common Data Format (CDF) concept [22], which presents a view of the data similar to the scientist's own.

The architecture of the VIS database manager is divided into two levels, conceptual and physical.[1] The

[1] Many texts on database systems [23, 24] include a third, or external, level, which is the view of the database for an individual user. For the VIS database manager, the external level is identical to the conceptual level.

conceptual level represents the view of the database shared by all users, and the physical level describes the physical organization of the database [23, 24]. The rest of this section examines the conceptual view of the database presented to end users, the view presented to application programmers, and the physical organization of the database.

The conceptual view of the VIS database system is designed to match the scientist's view of the data. A scientist works with collections of data that represent solutions to a computational model. A single data collection or data set is the result from one run of the model with certain underlying assumptions, initial conditions, and boundary conditions. In VIS, the scientist manipulates a conceptual entity called a *data set* that represents a hierarchical collection of entities called *variables* (see **Figure 2**). For example, a *data set* representing simulation results for a MOSFET device at a specific temperature might contain gate voltage, electron density, and electrical potential *variables*. Both *data sets* and *variables* have attributes. *Data set* attributes describe some aspect of the *data set*, such as name and size of the computational grid. *Variable* attributes describe some aspect of a *variable*, such as type (dependent or independent), name, valid range, and units. *Variables* also contain original data and attributes describing the data, such as resolution and format.

The VIS database system includes two tools for end-user database interaction. The database lister creates individual views of the database. *Data sets* can be retrieved and sorted by a number of attributes, including name and keyword. The database editor lets the scientist add, delete, or modify *data set* and *variable* entities in an interactive windowed environment.

While the scientist end user can interactively define and edit *data sets* without writing a program, an interface exists to satisfy the needs of application developers. A set of routines are available for defining and editing *data sets*, *variables*, and their respective attributes. The database manager component of VIS can be incorporated into other programs with the accompanying advantages of data-independence and encapsulating data management.

The VIS database is organized into an indexed file plus one binary, random-access file per *variable* per *data set*. The indexed file contains *data set* records implicitly connected [25] to *variable* records. Both record types are variable in length and consist of a sequence of attribute/value pairs. *Variable* records also hold a pointer to the file containing the *variable's* original data. Database files are transparent to both the scientist, who accesses the database through the database lister and editor, and the application programmer, who accesses the database through the interface routines.

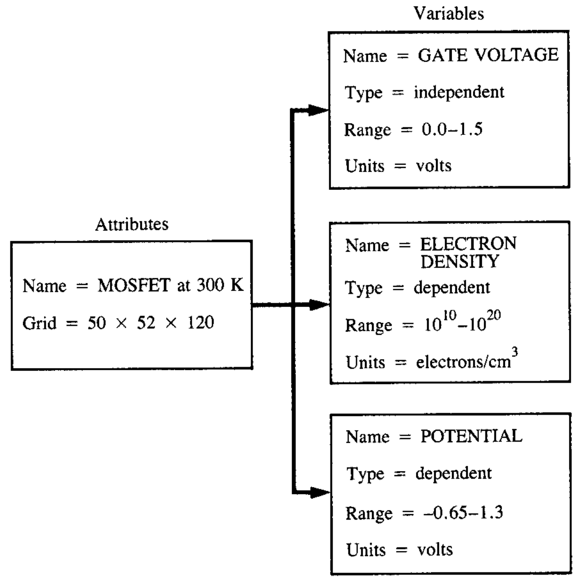


Figure 2

Hierarchical structure of a *data set*.

• *User interface*

The VIS user interface affects usability more than any other component of the system. It is the scientist's sole line of communication with the system and must be carefully designed to assist and not hinder the interpretation of multidimensional data sets. Design goals of the user interface include flexibility, consistency, and convenience. The rest of this section briefly describes the VIS two-screen system before looking at features of the user interface that impact our design goals.

The VIS two-screen system includes a control screen for user interaction and a display screen for showing the resulting image. The two screens have different uses and different requirements with respect to the user interface. The control screen is used for all of the tasks in the data interpretation process except display of the final image. Some of the tasks involved are data selection, scene description, and color specification. A user interface utilizing the "desktop" metaphor with overlapping windows seems the best match for the intended uses of the control screen. Windowing environments enjoy great popularity because separate activities can be placed in physically separate parts of the screen [26]. Control screen activities are separated into windows based on the dimensionality of the image, e.g., 2D, 3D, or 4D.

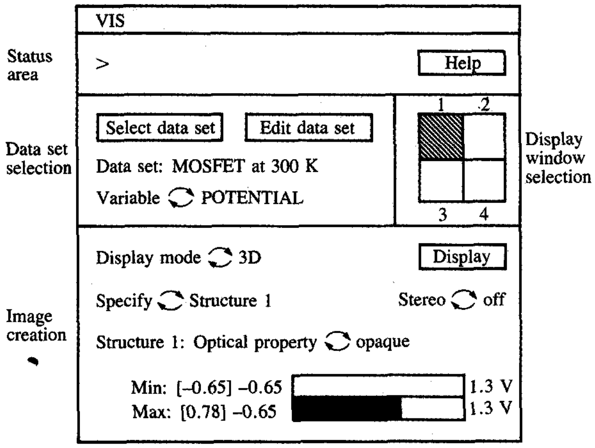


Figure 3

Design of a control screen window.

Unlike the control screen, which is responsible for many tasks, the display screen is solely responsible for image presentation. Different images are shown on the display screen, where they can be compared and contrasted without the clutter of the windows, menus, and sliders on the control screen. The display screen is separated into four tiled windows to avoid the added confusion of overlapping images and to simplify data interpretation. Display window selection and window operations are managed by an icon on the control screen.

The overlapping windows on the control screen allow the scientist flexibility. The novice user can use one window at a time in a sequential manner, while the more experienced user can bring up multiple windows and move easily among different display modes. Many scientists like to bounce back and forth between 2D and 3D display methods until they have isolated a particular area of interest in their data. The VIS user interface accommodates users at various levels of expertise by adapting to the way they like to work.

While overlapping windows afford the scientist a degree of flexibility, we must make sure consistency is not compromised. Consistency can be thought of in two ways. There is consistency in the user's mental model of the system and consistency in the way the user controls the system [27]. VIS is a tool for data interpretation for which typical user tasks are data selection, imaging routine selection, imaging parameter specification, and image display. The user interface reinforces this model by matching user tasks to specific areas of a control screen window. Consistency in user control is addressed in two ways. First, each control screen window is responsible for a particular display mode; 2D, 3D, and 4D display modes are controlled from the 2D control screen window, 3D window, and 4D window, respectively. Second, each control screen window obeys a consistent design (see **Figure 3**). A control screen window is divided into the following subwindows:

- *Status area.* The status area keeps the user informed about the task that is currently executing. A help button is also available.
- *Data set selection.* This subwindow lets the user select and edit *data sets* and *variables.*
- *Display window selection.* This subwindow contains an icon for selecting a display window and a pop-up menu for selecting window operations such as erase and zoom.
- *Imaging creation.* In this subwindow the user selects imaging routines, specifies parameters, and creates images.

The consistent layout of control screen windows fosters consistent control of the tasks in the data interpretation process and a consistent user model of the system.

Features of the user interface intended to make the system more convenient are intuitive data handling and profile customization. The VIS user interface takes advantage of the high-level data description kept in the database by letting the user manipulate data in its original context. The scientist examines variables with original names and physical units. Information on the valid range for a variable is used to establish sensible defaults. The end result is a more intuitive visualization mapping subwindow which simplifies the tasks of parameter specification and facilitates the data interpretation process. Profile customization lets the user override system defaults such as window color, window location, and scene description parameters.

- *Visualization manager*

The VIS visualization manager is responsible for creating all 2D, 3D, and 4D images. The tasks of the visualization manager are twofold, creating the parameter map and creating the image. Intuitive data handling simplifies parameter specification because the scientist manipulates data with familiar names and physical units. The imaging task is critical to system interactivity because this is where the user spends most of his time. The remainder of this section describes the 2D, 3D, and 4D imaging routines and concludes with a discussion of the visualization database.

The 2D component of the visualization manager is an important adjunct to the 3D visualization routines because scientists are more familiar with 2D displays of their data. The scientist can display 2D slices along a coordinate axis for a specific value of the independent variable or examine 2D slices for a specific grid value while the value of the independent variable changes. Image slices can be displayed in gray level or pseudocolor with or without contours.

While 2D imaging routines are useful, VIS emphasizes data interpretation in the 3D domain. The visualization manager uses a flexible optical-model approach to volume visualization. To display multiple 3D variables, the visualization manager implements a multivariant technique based on volume texturing. Details of the 3D imaging methods are deferred to the following section.

The 4D component of the visualization manager creates dynamic 3D images and is more aptly called the animation manager. The animation manager uses key frame animation and linear interpolation of data attributes to produce animation sequences. Individual frames are created by the 3D imaging routines. Animation playback is based on the rapid display (at least ten frames per second) of precomputed frames. The methods used to store and access animation sequences dictate the limitations of the animation. The animation manager displays precomputed frames stored in the frame buffer of the image processing computer (see the Appendix).

The animation manager allows a variety of effects including rotation, moving a cutout through the data volume, animation based on different values of the independent variable, and changing structure transparency. A special type of animation sequence called a browse sequence can also be created. A browse sequence contains animation frames with spatial orientations clustered around a central key frame. When the sequence is replayed on the display screen, the scientist uses a mouse to interactively browse different orientations of his original key frame (3D image). The 3D image moves left, right, up, and down with the movement of the mouse.

Creating images is an important feature of the visualization manager, but equally important, if not more so, is the ability to save and retrieve interesting images. The visualization manager contains a visualization database for storing images from the 2D, 3D, and 4D imaging routines. The visualization database holds image data plus a high-level image description. This is similar in concept to the database described in the preceding section on the database manager. The descriptive information about an image is a complete record of the parameter map used to create that image. The scientist can retrieve the image, the parameter map used to make the image, or both. This provides a flexible mechanism for parameter map reuse or templating. The visualization parameters of an image in the database can easily be applied to another *data set* or used as the starting point for further exploration of the current *data set.*

Another benefit of the visualization database is the ability to create hard copy that is faithful to the original. In the words of Thomas A. DeFanti, "If you can't record it, it ain't science." Access to the descriptive information about an image lets application programs generate hard copy using original image and data set attributes. We have implemented a program that creates and records video scripts. The program extracts information (number of frames, frame size) from the visualization database before recording animation sequences [5]. We can imagine a more sophisticated program that lets the scientist browse the visualization database, select an image, interactively label the image with the descriptive information in the database, and create a 35-mm slide or record a frame on videotape.

3. Visualization techniques

Visual data interpretation is a process handled by the visualization manager of VIS. It involves several basic tasks. The user first reviews the data with 2D imaging to confirm that it is loaded correctly, and that preprocessing and reformatting have not introduced artifacts. The next task is to determine the general shape and locations of key regions in three dimensions. After interrelating these regions, the user may wish to determine the distribution of values within them. When several variables are involved, it is necessary to compare regions in different variable spaces. In some cases, these 3D data sets may also depend upon an independent variable, such as time; and an animation series is required. The user may iterate through these tasks several times to obtain a complete understanding of the data.

These basic interpretive tasks impose four operational requirements on the visualization techniques. *First,* the user must be able to translate the image on the screen into an understanding of 3D relations in the data. He needs a unique interpretation, i.e., to be able to invert the displayed image to obtain an unambiguous understanding of the data. Distortions can be introduced by the visualization method itself. For example, the surface shading method can introduce artifacts comparable to data fine structure. Distortion can also be introduced by overprocessing the data. The data are filtered to form a smooth structure, but small variations are lost. *Second,* visualization techniques must be versatile, since applications often involve a wide range of data structures. Data interpretation is exploratory; both 2D and 3D options are required. A *third* requirement is interactive imaging. The user needs a response within five to fifteen seconds for volumetric imaging of complex data. The *fourth* requirement is ease of use; underlying imaging operations should not get in the way.

These four operational requirements are the guidelines used in developing VIS. They dictate which visual clues are implemented, and the approach used to form images. Easy-to-use methods are preferred over sophisticated methods which produce a "realistic" image. The goal is to express the data in a visual form which suggests the spatial relations. Rendering based on a realistic model is computationally intensive and more difficult to use, since a larger number of parameters are required.

Cues used to express depth and shape are discussed in the following section. In the subsequent two sections, on the optical model and image formation, the technique of forming a 3D image is described. Interpretation of volumetric data and multiple-variable methods are presented in the last two sections.

• *Depth and shape cues*

Artists and video animators use a variety of visual cues to express the shape and location of objects [28]. Their task is basically similar to that of the computer graphics user, namely, to create a 2D image that expresses 3D relationships. The cues used by artists are a valuable collection of techniques for visual data interpretation. There are three general types of cues: microscopic, mesoscopic, and macroscopic. The four operational requirements (invertability, versatility, interactivity, ease of use) determine which cues are implemented in VIS and which cues are excluded. Some cues may be useful, but have not yet been evaluated for VIS.

Microscopic

Microscopic cues are related to local surface and depth properties. Surface color tiling expresses the shape and curvature of a surface and is simple to interpret [29]. The use of data-dependent surface colors is also a useful technique; the user can display two variables in one image. Volume texturing is based on mapping each point of a region into a color based on the data value, and is a simple method of forming surface colors [30–32]. In addition, since the entire volume is colored, even the surfaces of 3D cutout regions have data-related colors [13, 33]. Transparency is also a powerful microscopic cue for visualizing the internal distribution of data values. Transparency and volume texturing are implemented with VIS.

Surface reflection is effective in displaying the curvature of surfaces. However, for irregular regions and small data arrays, surface reflection may complicate interpretation, especially with transparency or volume texture. Surface texture or roughness, which depends on data values, is feasible but not appropriate for most data visualization applications. Also, reflection and texture require additional computation and hinder interactivity; they are not implemented in VIS.

Mesoscopic

Mesoscopic cues relate to the overall shape and location of structures or regions. Depth shading is a very useful cue. It is based on the visual association of objects of comparable value (lightness), and does not require data continuity [12]. Depth shading is implemented in VIS by using ambient attenuation in data space, as described below. Depth shading as implemented in VIS does not increase the time required to form a 3D image.

Relative motion during rotation is a powerful mesoscopic cue. It improves spatial perception in three ways. *First*, as the data rotate about their center in space, points in the front and back can easily be distinguished because they move in opposite directions in the 2D image seen by the viewer. *Second*, a clearer perception of 3D shapes is obtained, because structures are seen from several different orientations. *Third*, disjoint fragments of a region are seen as parts of the same structure since they move together in a coherent manner. Motion is achieved in VIS using animated display of a series of 3D images.

Stereographic imaging is seldom used in visual arts, but is effective for 3D data interpretation. Methods based on shuttered glasses or polarizing plate over the CRT screen are particularly useful and simple to use. A method which does not require special hardware is based on forming two images, one rotated six to eight degrees relative to the other, and viewing them with crossed eyes. VIS provides an option to form two adjacent images on the screen for direct stereo viewing, two interlaced images for shuttered systems, and scaled images for field-switched stereo.

Position and scale are often used in art, but seldom used in technical imaging. Objects positioned at the base of the image appear closer than objects near a central horizon line. Also, a larger shape is interpreted as closer than a smaller one. Color can also be used to suggest position; stronger colors (red vs. blue, saturated vs. unsaturated) are perceived as closer. Position, scale, and color cues are available with VIS, but require evaluation for data interpretation. Perspective representation of objects is frequently used in classical drawing and painting, and is implemented in VIS. Shadows formed by a light source improve perception of spatial relationships, but require significant computation and may cause ambiguous interpretation with volume texturing and transparency. Shadows are not formed with VIS.

Macroscopic

Macroscopic cues concern the overall image. Objects displayed in a three-dimensional "room" are interpreted as three-dimensional; the 3D environment suggests that the objects are also 3D. These macroscopic cues also provide a spatial reference frame. The object can move relative to the frame, or the point of view can change;

i.e., the user can move around the object, viewing it through the display window.

A 3D space is suggested with linear and aerial perspective in VIS. Linear perspective is based on linear bars that diminish toward a horizon line. Aerial perspective is based on a color shift to less saturated and bluer colors. These two cues are used to form the base plane in the 3D images formed by VIS. Aerial perspective is assumed to change exponentially with distance from the viewer. Changes in base-plane texture also provide depth cues; however, texture changes require more computation and may conflict with data changes when transparency is used; texture is not implemented.

• *Optical model*

An optical model is implemented in VIS to display and interpret 3D data; the selected visual cues have simple implementations, and the four operational requirements (invertibility, versatility, interactivity, and ease of use) are met. The optical model is based on assigning a color and attenuation to each point on the basis of its data value. It is an effective method for displaying and interpreting volumetric data [6, 9, 12, 34]. The model accommodates several options, such as cutouts, region highlighting, volume texturing, and domain stripping [13].

The optical model is a mathematical representation of a three-dimensional region divided into cubes, called voxels. A parameter map determines the transformation of data values into voxel attenuation and brightness. Different features in the data array are visualized by selecting the appropriate combinations of voxel attenuation and brightness, i.e., parameter maps. The map may include dependence on position to form a cutout, or changes from key frames to form an animation series. This approach is not based on surfaces, nor does it require the data to have any internal structure.

Since the optical model is based on volume-filled data, the input data array may require preprocessing [29]. In computer simulations, output data values are often computed at mesh points which are not uniformly distributed in space. For the approach used in VIS, it is necessary to interpolate the data to a uniform rectilinear grid. If the data are obtained from a scanning device, the distance between scan plans may be larger than the pixel separation. The scan images can be replicated or interpolated to fill between the scan planes [12].

• *Image formation*

To display the 3D optical model, i.e., a "cube" of colored voxels, it is necessary to project it onto a plane corresponding to the display screen. The model is defined in object space; its projection is in image space. A back-to-front (BTF) method is used in VIS. The rear data plane is projected and merged over the projection of the background, taking into consideration the attenuation and color of the data points. (The background is a precomputed perspective base plane and uniform rear plane.) The next-to-rear data plane is then projected and merged over the composite of the background and first data plane. This procedure is repeated until all the planes are projected.

This BTF method satisfies the four operational requirements, and has several advantages. As the final image is being formed on the screen, meaningful intermediate images are obtained. The user does not have to wait for the final image to obtain results; there is no wasted time. Further, the method is simple and permits high-speed implementation.

The z axis (the axis toward the viewer) need not be a spatial dimension. It can correspond to time; the 3D image then presents an entire 2D process in one image [13]. The z axis can also correspond to an independent variable, such as gate voltage in the case of a switching transistor. The process of device switching can then be seen in one image.

In VIS a perspective projection is used, as opposed to an orthogonal projection. There are three basic steps to add each successive plane to a 3D image. First, the data values are mapped into color and attenuation. Second, this image is warped to form a perspective image at the correct position and orientation. Third, this transformed image of the data is merged over the current 3D image in the BTF projection sequence. The merge operation consists of multiplying the current 3D image by the transmission of the transformed image and then adding the transformed image [6, 13]. The same attenuation is used for each color; i.e., there is no color dependence in transmission.

The BTF projection method provides rapid 3D imaging with a wide range of interactive options. The computation structure and data access are the same in all cases; only the color/attenuation map is different. The time required to form the image is *independent* of which options are used.

A solid structure is formed by using maximum attenuation (zero transmission) for selected voxels, such as all voxels whose values lie in a specified range. The same region is displayed transparently by setting attenuation near zero. Depth shading is achieved by assigning a small attenuation without color to unselected voxels. Multiple solid or transparent structures are formed by selecting multiple value ranges. A 3D region is removed, or "cut out," by setting the color and attenuation of the cutout region to the background value.

• *Volume visualization*

One of the most difficult tasks of visual data interpretation is visualizing the 3D distribution of values in a selected region. The optical model with various visual cues implemented in VIS provides three options: volume texturing, colored ranges, and transparency. These options are described here and illustrated in the following section with output from a large-scale simulation.

The volume texture option is based on mapping each voxel in the selected region to a color which reflects its value [9, 12, 13, 30]. Typically, a spectral-type map is used, with blue corresponding to low values, yellow to intermediate values, and red to the highest values. With this option, interior values may be seen by "cutting out," or removing, a region in front. The cutout region can be "swept" through the data by forming several 3D images with different cutouts, using animation for display. An alternate approach is to split the region open with a "cutting" plane and rotate the two segments in opposite directions. This approach has the advantage that nothing is removed and the interior is visible. With animation the segments can be opened and closed to clarify the interrelation of left and right parts [12].

The color range option is based on selecting several value ranges and using a distinctive color for each range. The data are then displayed as concentric shells, with different colors corresponding to different values. When appropriate ranges are selected, the shells will have space between them, and the distribution of values can be visualized. Unselected voxels have zero color and small attenuation for depth shading.

Transparency is a third option for visualizing the interior of a region. There are three aspects of transparent imaging. The *first* aspect of transparency is spatial dependence; there are three types of spatial dependence. The first dependency is based on displaying the surface or exterior shell of the selected region transparently, like a soap bubble [9, 35]. This type of imaging displays the extent of the selected region and its curvature if exterior light sources are used. With the second type of spatial dependence, the selected region can be imaged with a uniform color. The viewer can then perceive the thickness of the selected region. The third spatial dependence assigns color and/or attenuation based on data values. For this case the viewer can interpret the distribution of values within the select region [6, 9, 13, 30, 34, 36].

Value dependence is a *second* aspect of transparent imaging, i.e., the dependence of attenuation and color on the data value. If voxel attenuation depends on data values and if voxel color is set to zero, an absorptive 3D region is modeled. By projecting this region against a bright background, an X-ray-like image is obtained. On the other hand, if attenuation is set to zero and color depends on the data values, a luminous region is seen. A third option is to have both color and attenuation depend on the data values in the selected region. In VIS the relation is modeled as dense emitters, i.e., small luminous opaque particles. The concentration of particles is proportional to data value [34]. The dense emitter model, which has a natural interpretation as a luminous cloud, has been useful in several applications.

The *third* aspect of transparent imaging is the method of combining overlapping voxels. One approach is based on averaging. For example, if the voxel has a transmission of 80 percent, with averaging, 20 percent of the rear voxel is added to 80 percent of the front voxel. The second approach is based on addition: 20 percent of the rear voxel is added to the front voxel. The first approach has the advantage that the resultant colors never exceed unity. The second approach has the advantage that it represents a physical process; attenuation is an exponential function of path length. The second approach is used in VIS.

The optical model in VIS can implement most transparency options. A transparent 3D image provides more information than an opaque image; however, it is necessary to increase perceptual "dimensionality" with stereo or animation. Small-angle animation with stereo is very effective, as demonstrated by Wixson [37, 38]. Transparent imaging is especially useful for imaging data with multiple, finger-like regions [12, 13, 39].

- *Multiple variables*

Large-scale simulations of complex physical processes frequently produce several variables. For example, a simulation of a solid-state device involves 3D distributions of electron density, hole density, and electrical potential. Visual interpretation of these results requires special options, since the user must perceive and interrelate several 3D distributions. This kind of composite perception is outside ordinary experience, and therefore requires some care in visualization.

Three options are available with VIS. The user can view several windows on the display screen with a different variable in each window [29]. This approach has the disadvantage that it is difficult to relate corresponding 3D positions, although it does provide a complete display of the variables. A second option is to combine a couple of the variables to form a new variable. Variables can also be combined using multiple-variable classification to map each voxel into a class. A 3D image of class index is displayed with different colors for each class [12, 40, 41].

A third option is based on a volumetric texture map. Voxels in the data space are colored with reference to the first variable. A structure is then selected based on a range of values of the second variable; i.e., only voxels with values in this range are displayed with colors based on the first variable. The user sees a region defined by the second variable with surface colors defined by the first variable. If a section is removed from the structure, values of the first variable are seen in the interior of the structure, and on the faces of the cutout.

4. VIS and transistor design

In the preceding sections, data management and visualization features of VIS have been described. Several concepts are illustrated in this section with the output of a large-scale transistor design. However, data interpretation is an interactive, exploratory process; the utility of VIS can be only partially presented with still frames and video.

• *Background*

The basis of the present complementary metal-oxide-semiconductor (CMOS) technology is the metal-oxide-semiconductor field-effect transistor, or MOSFET. As an element in a circuit, this device acts as a voltage-controlled current source; a voltage applied to one terminal controls the amount of current that flows between two other terminals.

A simple MOSFET structure is depicted in **Figure 4**. A voltage applied to the metal or heavily doped polysilicon gate contact, which is isolated from the silicon substrate by the gate oxide, modulates the charge density at the silicon surface. When a sufficiently large positive gate voltage is applied, a large electron density appears at the surface. In general terms, the gate voltage required to do this is known as the threshold voltage; it can be adjusted by changing the impurity doping level in the silicon. This allows the electron-rich source and drain regions to be connected electrically; with a positive voltage on the drain and the source grounded, an electron current flows from the source to the drain. When no voltage (or a negative voltage) is applied to the gate, there are no electrons at the silicon surface and no current flows.

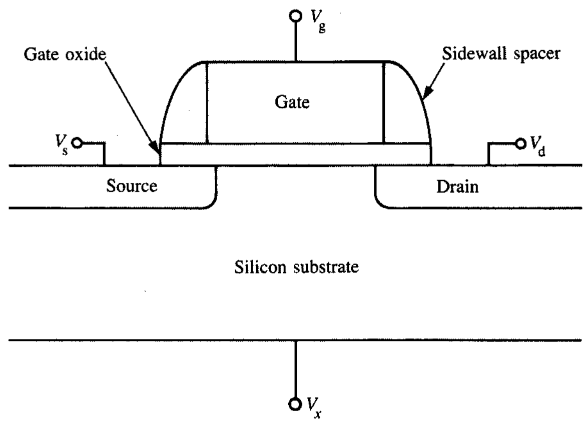


Figure 4

Simplified MOSFET structure: key electrical and structural components.

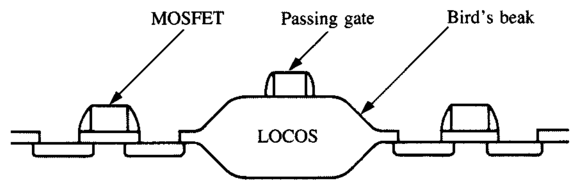


Figure 5

Two MOSFETs separated by LOCOS isolation, showing the LOCOS bird's beak.

As device dimensions continue to decrease, isolation of one FET from another has become a more difficult issue. To date, local oxidation of silicon (LOCOS) isolation has been sufficient to accomplish this task for devices with dimensions larger than one μm. A basic LOCOS structure is depicted in **Figure 5.** This structure is formed by depositing silicon nitride and patterning it lithographically to cover only the regions that will eventually contain the FETs. Conventional wet-furnace oxidation is used to grow the oxide. This prevents leakage from the source of one FET to the drain of an adjacent FET. In addition, as shown in Figure 5, this allows a gate that will eventually reach another FET to pass between two existing FETs without forming an unwanted FET under the LOCOS (due to the thick oxide).

Below 1 μm, the encroachment of the oxide "bird's beak" (Figure 5) into active device regions has proven to be a limiting factor. This leads to thicker oxide in the active device region, hampering device performance. In addition, dopant atoms migrate under the LOCOS into the device areas, further affecting device behavior.

An alternative technology involves etching a trench and filling it with oxide to provide isolation [42]. This shallow trench isolation (STI) approach has been demonstrated to be a feasible technology for submicron isolation, eliminating the bird's beak.

However, the introduction of this technology has radically changed device behavior. In LOCOS-isolated devices, a polysilicon gate running across the device turns upward and away when it reaches the isolation oxide [**Figure 6(a)**]. The gate over the isolation has no effect on the device characteristics. Since the situation is uniform

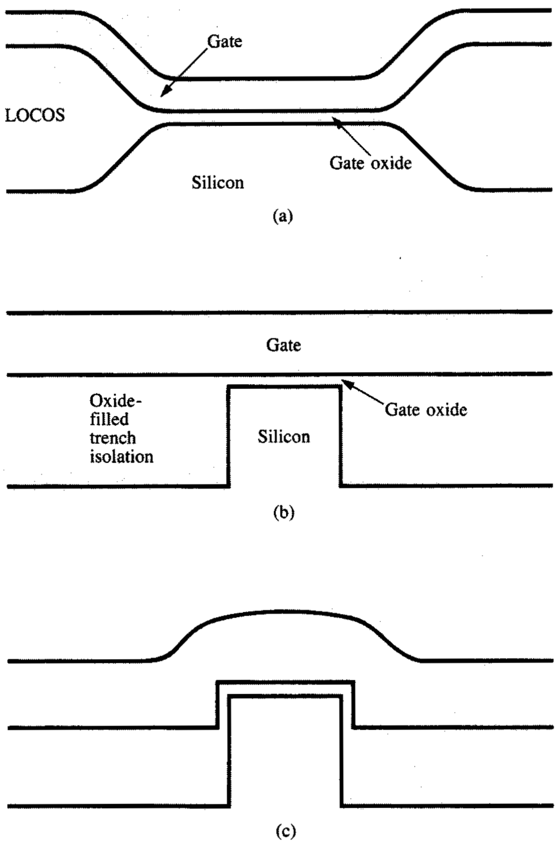


Figure 6

Cross section through MOSFET structure at mid-channel, perpendicular to the channel axis: (a) LOCOS-isolated device, (b) trench-isolated device, (c) trench-isolated device with isolation oxide recess and gate wraparound.

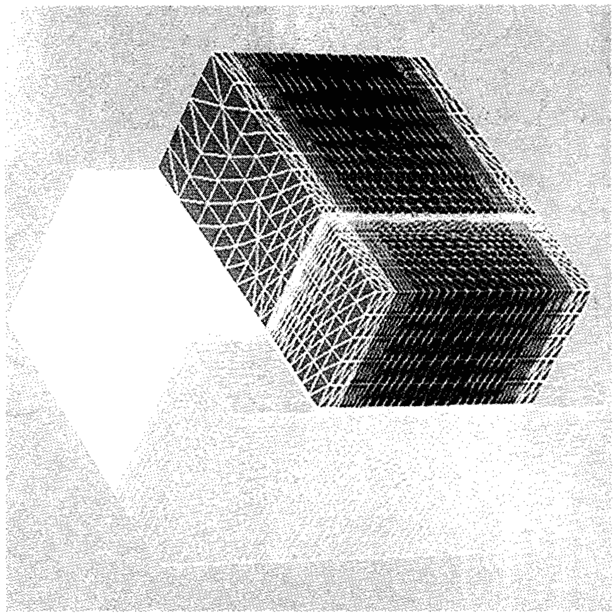


Figure 7

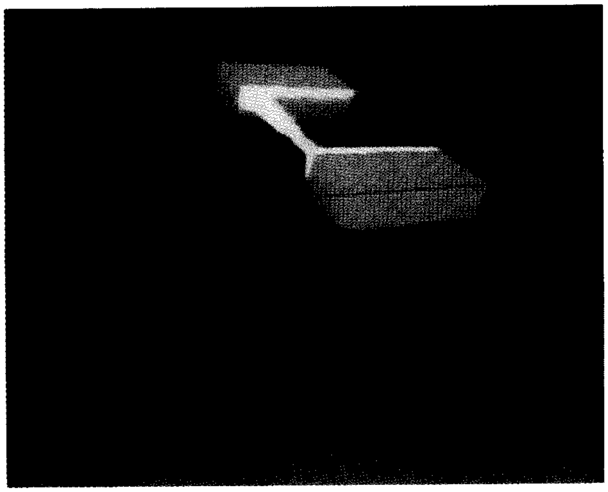
A typical three-dimensional mesh used for device modeling: gray = silicon, yellow = oxide, blue = polysilicon, red = nitride.

across the device width, two-dimensional device modeling (in the plane along the device axis) is adequate to describe the device behavior. (It should be noted that in very narrow LOCOS structures, the encroachment of the bird's beak discussed above leads to a nonuniform oxide thickness across the channel. This leads to the so-called "narrow-channel effect," in which the threshold voltage increases with channel width. However, such structures are undesirable and are generally not employed.)

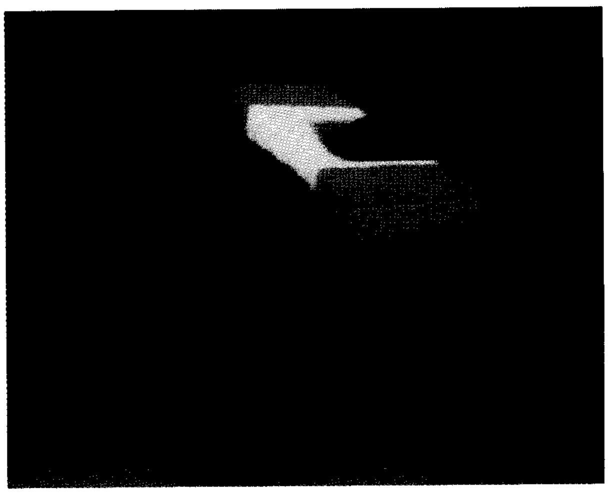
In the case of an STI-isolated device, the gate runs flat across the isolation [**Figure 6(b)**]. This allows the gate to exert a fringing effect on the corner and sidewall of the device mesa [43–45]. The effect is further strengthened if, due to erosion of the isolation fill, the gate wraps around the corner of the device mesa [**Figure 6(c)**] [44, 46]. Now, the uniformity that previously existed across the device width is no longer present. At this point, three-dimensional modeling is required to simulate the device characteristics.

• *3D device simulation*

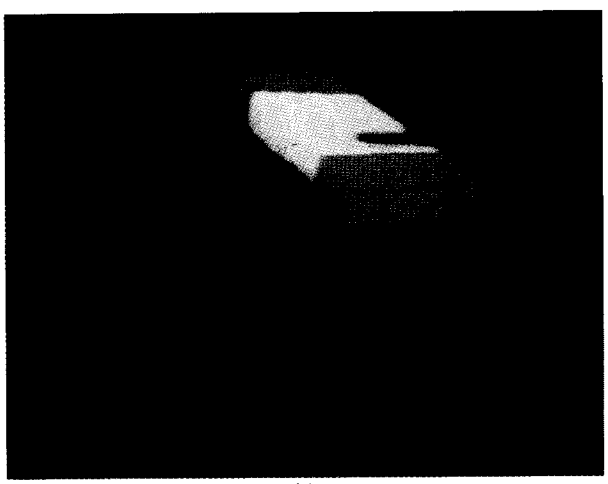
In moving from two-dimensional to three-dimensional modeling, the size of the simulation is greatly increased, since the entire device, rather than a two-dimensional cross section, must be examined. This requires more storage and significantly more computation time to arrive at useful results. Increases of one or two orders of magnitude in these parameters are common. Additionally, the introduction of another degree of freedom leads to a large expansion of the size of the solution data. In two-dimensional modeling, contour plots of the modeled device cross section (for parameters such as electron density or potential) provide all of the information in the final results. Three-dimensional modeling, however, leads to a large number of such cross sections. It becomes very difficult to examine and interpret the results in a meaningful way that is still concise. Improved interpretation of 3D results can be achieved with new scientific visualization techniques. Obviously, interactive 3D imaging enhances data interpretation. This approach can be further improved through the use of color and animation.



(a)



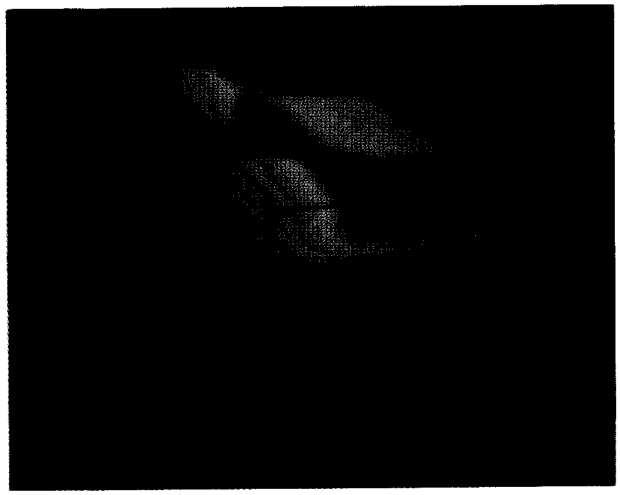
(b)



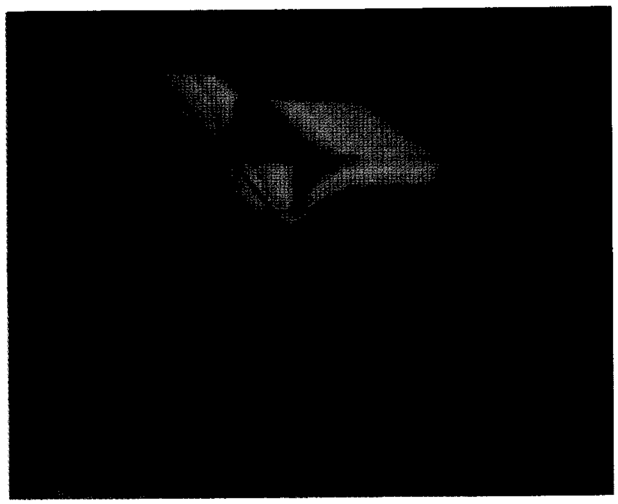
(c)

Figure 8

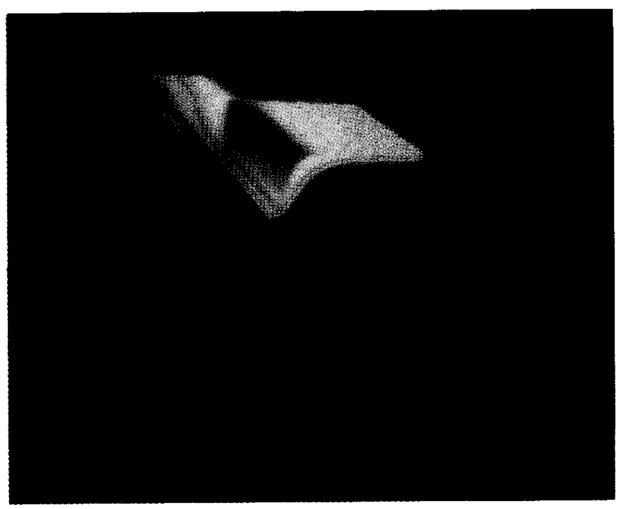
Electron density at three bias values: (a) bias = 0.0 V, (b) bias = 0.25 V, (c) bias = 0.40 V.



(a)



(b)



(c)

Figure 9

Electrical potential with three cutouts at a gate bias of 0.25 V.

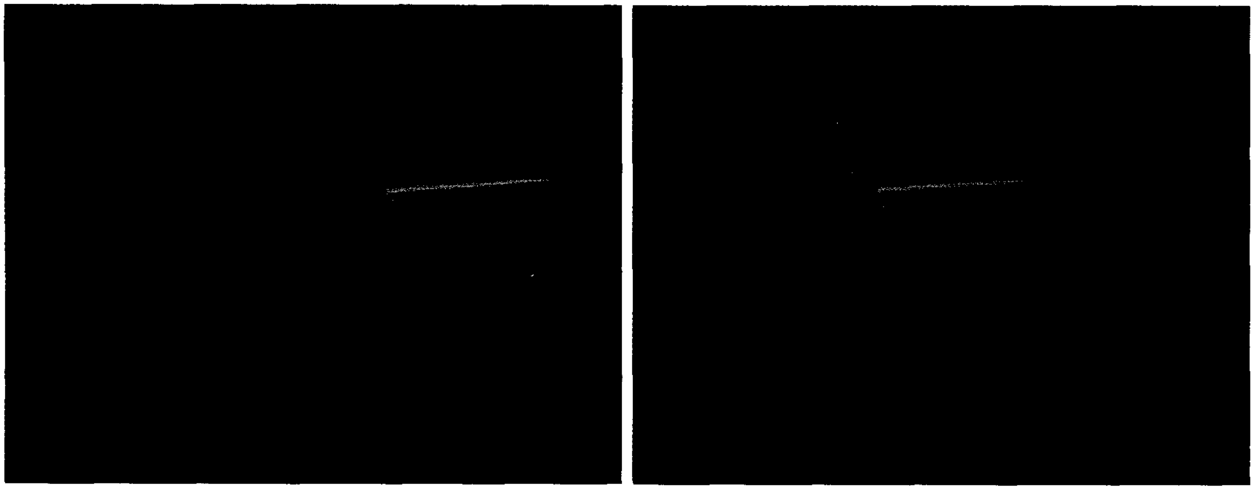


Figure 10

Transparent electrical potential with potential greater than 0.50 V displayed as opaque red stereo pair.

The objective of the device simulation to be described here was to study the current turn-on behavior of trench-isolated devices in which the corner effects described above are clearly important. To carry out this task, a three-dimensional mesh (**Figure 7**) was constructed for use in the FIELDAY device simulator [47]. This mesh contained approximately 30 000 nodes and 60 000 elements.

Device characteristics were studied at room temperature (300 K) and liquid nitrogen temperature (77 K). This was done to exercise the low-temperature modeling capabilities recently added to FIELDAY [48] and to examine the feasibility of employing isolation-trench technology in a low-temperature environment. In each case, a low drain bias was chosen (0.2 V), while the gate was swept from 0.0 V to 1.5 V in 0.05 V increments. The room-temperature simulation required 929 CPU minutes on an IBM 3090[2] computer, while the low-temperature simulation required 4053 CPU minutes.

• *Results*

The simulation output consists of 3D arrays of electron density and potential at 77 K and 300 K for 14 gate bias voltages. Since there are 3136 xy cross sections along the z axis, it is impractical to examine more than a small portion of the data with 2D images. Volume visualization is essential in determining the 3D distribution of electrons and potential, and in characterizing the turn-on behavior of the device.

The presence of electrons in the gate region indicates that a conducting channel has formed and that a significant current is flowing in the device. A basic problem is to visualize the 3D shape and location of the channel. The channel is imaged by using an opaque model for all voxels with electron density greater than 2.6×10^{13}. The 3D images are presented in **Figure 8** for three gate voltages, 0.0, 0.25, and 0.40 V. The drain is nearest the viewer, with the source in the rear. Voxels below the gate are highlighted. The left side of the 3D region corresponds to isolation oxide; electrons and holes are not modeled in that region. As the gate voltage increases, the channel widens and fills the gate region. Note that the channel widens both horizontally and vertically, with the widest portion near the source. With a 3D image the shape of the channel is characterized in a single image. An animation sequence of 3D images for increasing gate voltage demonstrates the relative speed with which different regions open.

The distribution of electrical potential within the device influences electron flow. An important problem is to locate regions with a large potential gradient, since impact ionization may occur at these sites and cause the device to fail. The distribution of electrical potential is displayed in **Figure 9** at a gate bias of 0.25 V. Volume texturing is used, with red corresponding to high potentials, green to intermediate, and blue to low. Potentials in both the silicon and isolation-oxide regions are imaged; the void in the center of the oxide corresponds to where the gate wraps around the silicon. A cutout is used to remove the front planes at increasing depth. In the first image (a), the drain is seen in the upper right corner. The second image (b) shows the potentials in a plane at the drain/gate interface. It illustrates the large potential gradient that occurs at the vertical edge of

[2] 3090 is a trademark of International Business Machines Corporation.

the gate. A similar gradient is seen in image (c) at the horizontal corners of the gate.

Transparency is also helpful in visualizing regions of high potential in relation to the entire device. A stereographic pair is presented in **Figure 10**. (A stereo effect can be produced by viewing with eyes crossed.) The regions of highest potential occur at the corners of the gate which contact the silicon. Interpretation is enhanced with animated rotation, as illustrated in the accompanying video.

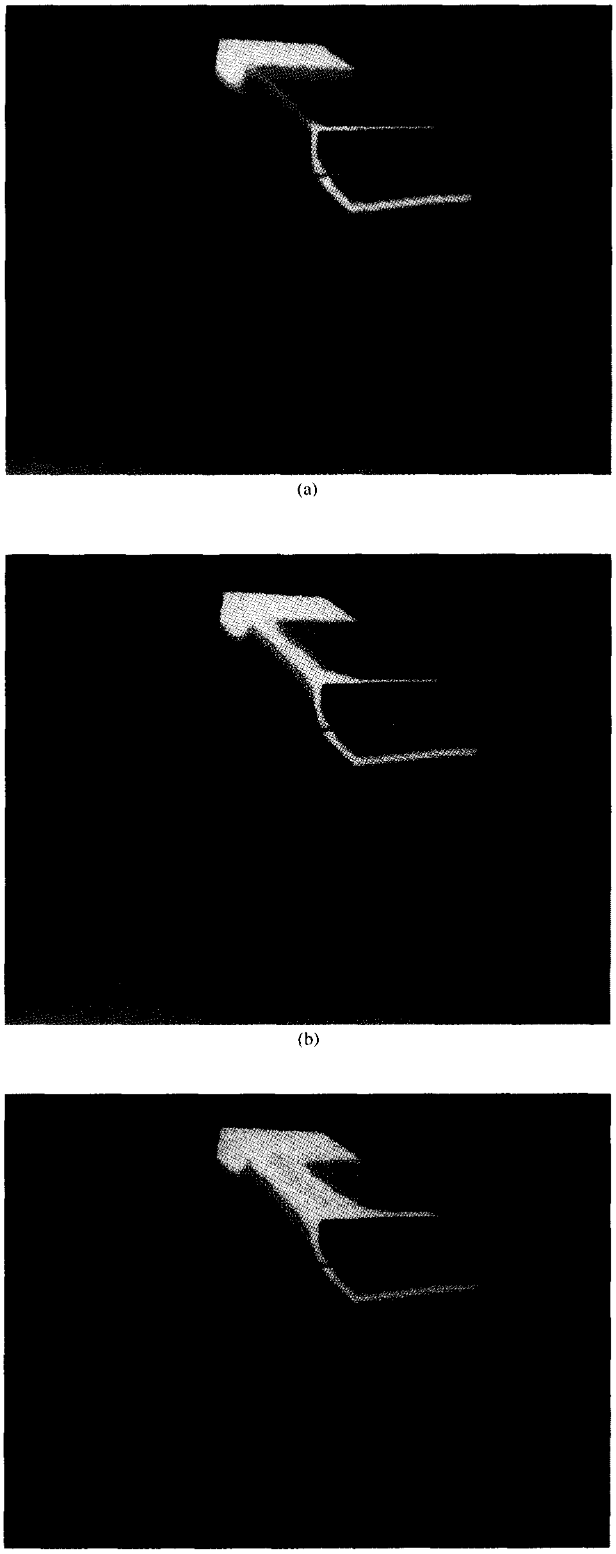
The relationship between electron density and potential can be visualized with volume texturing. The electron density (Figure 8) is colored by potential (Figure 9). The resulting images are seen in **Figure 11** for the same bias values used in Figure 8. Initially the channel is at a low potential; the potentials in the channel rise as the channel opens. Note that the potential is highest at the upper edge of the channel and decreases rapidly with distance from that edge. Even though a large potential gradient occurs at the lower corner of the gate (Figure 10), the channel is not open there.

As discussed above, the primary objective of this simulation study was an examination of how changes in the temperature affected the conduction of the conventional planar portion of the device compared to the effect on the corner conduction mechanism.

Channel current at $T = 300$ K for a full range of gate potentials is presented in **Figure 12**. A two-dimensional result has been included to demonstrate the behavior of the planar portion of the device. The lower turn-on present in the three-dimensional model clearly indicates the effect of the device corner; because of the higher field at the corner, this region of the device has a lower threshold voltage than the center of the device.

Channel current at $T = 77$ K for a full range of gate potentials is presented in **Figure 13**. Once again, because of the higher field, the device corner turns on well before the planar region. However, the corner conduction mechanism is clearly less sensitive to temperature than is the planar conduction mechanism. (Further details on device aspects and experimental results may be found in [45] and [46].) This is a very interesting result that can be further examined by 3D visualization techniques.

A cross section taken through the middle of the device perpendicular to the channel axis allows the development of the electron density to be examined as the gate voltage is increased. An interesting approach is to take this x–y cross section and plot the gate voltage on the third axis. **Figure 14** is such a plot for $T = 300$ K and 77 K. At 300 K, Image (a), the two conduction mechanisms are clearly displayed. At low gate bias, the device corner turns on. The extent of the electron density increases as the gate bias increases. At higher gate bias, the planar portion of the device turns on, as evidenced by the



(a)

(b)

(c)

Figure 11

Electron densities greater than 2.6×10^{13} colored by electrical potential at three bias values: (a) bias = 0.0 V, (b) bias = 0.25 V, (c) bias = 0.40 V.

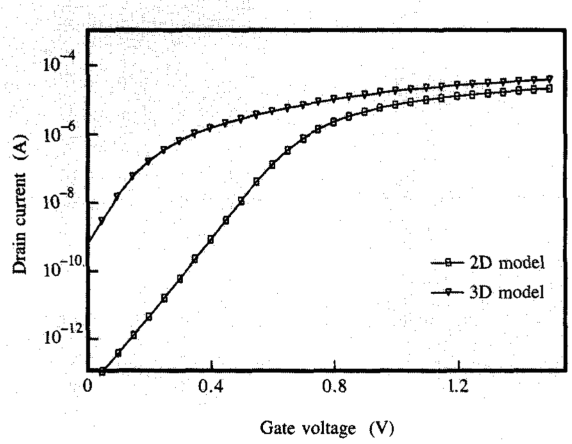


Figure 12

Channel current vs. gate potential for 2D and 3D device simulations at 300 K.

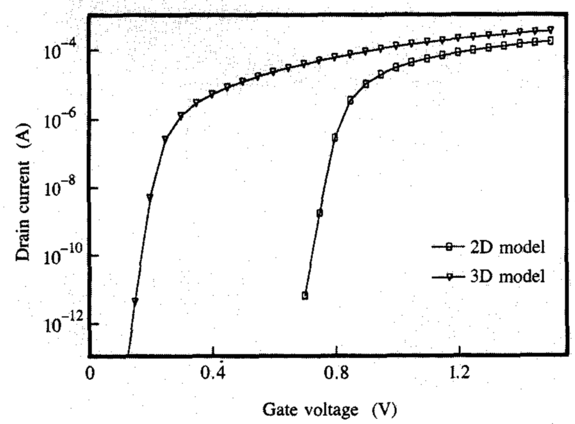


Figure 13

Channel current vs. gate potential for 2D and 3D device simulations at 77 K.

appearance of electrons across the entire device. Image (b) is a similar plot for $T = 77$ K. In comparing the two cases, several important differences can be noted. When only the corner conduction mechanism is active, the electrons are more closely confined to the corner. As the gate bias increases, the electron density does not spread out laterally as much as it does in the $T = 300$ K case. Finally, the separation in the gate bias between the two mechanisms due to the lower temperature is clearly evident.

5. Discussion

Visual data interpretation requires tools that provide unambiguous images of the data, that are versatile for a wide range of data structures, that form images rapidly for interactive interpretation, and that are easy to use. The Visual Interpretation System (VIS) satisfies these requirements. The effectiveness of VIS results from an interactive interface which controls a database manager and a visualization manager. The database consists of entities called *data sets* that carry a complete description of the geometry and data characteristics. Visualization management involves both 2D and 3D imaging in multiple display windows and animation.

Three-dimensional data imaging is based on optical modeling with back-to-front perspective projection. This approach has several significant advantages. First, imaging is not based on the concept of surfaces or structures in the data. Diffuse regions as well as structures can be displayed. Second, a single algorithm produces a wide range of 3D representations by simply changing the parameter map of data into color and attenuation. This allows a high-speed, efficient implementation. Third, the time required to form the image is independent of which options are used, i.e., the values of the parameter map. Fourth, with a back-to-front projection method, informative 3D images are being displayed as the final 3D image evolves on the screen. Intermediate images are valid 3D representations.

VIS has been used to address problems in several applications: transistor design, meteorology, seismology, medical imaging, computational fluid dynamics, and quantum-molecular dynamics. In this paper, the internal processes of a semiconductor device are studied with VIS. We have demonstrated how 3D visualization techniques provide insight into the physics of isolation-trench-bounded devices at both room and low temperatures, which facilitates the development of improved designs.

Since device performance frequently involves dynamic processes, animation is an important tool for interpreting the data. The videotape that supplements this paper presents several of these dynamic processes.

A variety of visualization tools have been presented in this paper. However, recent developments in science and technology have resulted in new visualization requirements; there remain important areas for additional research. The display and interpretation of vector fields has several problem areas which need to be solved [4, 33, 49]. Multidimensional data and tensor fields are a challenging area requiring new techniques. The combination of sound and imaging is a promising

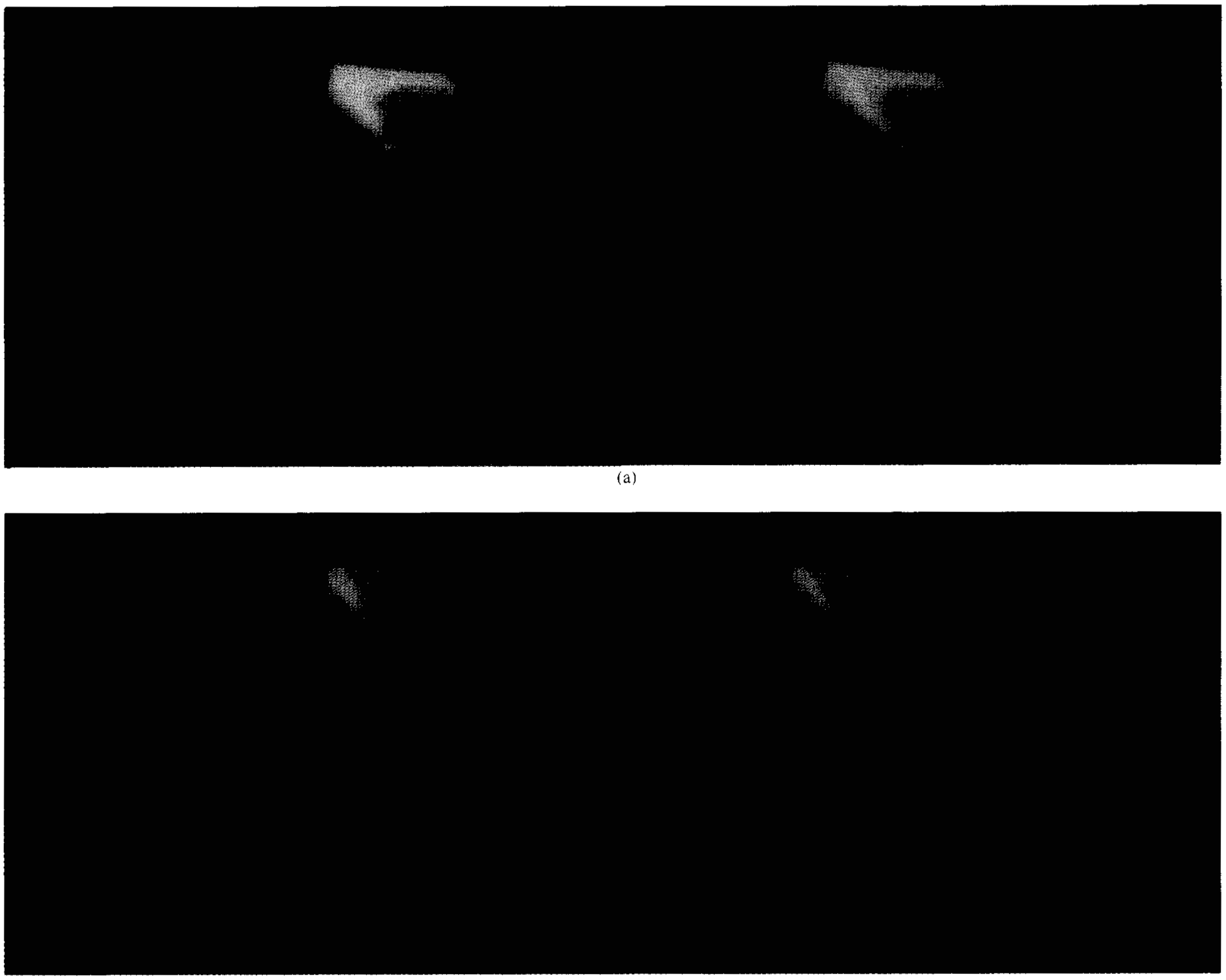


Figure 14

Electron density in a central cross section with gate bias increasing inward: (a) 300 K simulation, (b) 77 K simulation. Stereo images can be viewed with crossed eyes.

approach [50–52]. Visual data interpretation is clearly an expanding discipline which has only begun to offer significant benefits to science and technology.

Appendix: System hardware

VIS is an experimental system for research in the visualization of multidimensional data sets. System hardware consists of two basic components, the host workstation and the image processing computer. The host workstation handles data selection and retrieval, selection of a rendering routine, and specification of parameters for the renderer. The VIS host workstation is a Sun4/280[3] with eight megabytes of memory and one gigabyte of disk space. The user interface is written using the SunView[3] toolkit [53] that runs on the SunWindows[3] window system [54].

The image processing computer used to render the images is the PIXAR Image Computer[4] [55]. Data are transferred from the host workstation to the PIXAR memory over a VME bus. The PIXAR has 48 megabytes of memory, called a frame buffer, designed to accommodate 48-bit pixels stored as 12 bits each of red, green, blue, and transparency (alpha). Monochrome applications can use the alpha channel as a fourth frame buffer or as scratch space. Images of various sizes can be handled by reconfiguring the frame buffer through

[3] SunWindows, SunView, and the combination of Sun with a numeric suffix are trademarks of Sun Microsystems, Inc.

[4] Image Computer is a trademark of the PIXAR Corporation.

software. The linear address space of the frame buffer can be configured to any rectangular shape containing an even number of tiles.

Once the data are in the PIXAR frame buffer, they are processed by one of three CHAPs (Channel Array Processors). Each CHAP is a SIMD (single instruction multiple data) machine with one processor for each of the red, green, blue, and alpha channels. A processor runs at 10 MIPS, yielding a total speed per CHAP of 40 MIPS. The CHAP is a Harvard architecture machine with instruction memory separate from data memory. The code for 3D imaging in VIS is compiled and loaded into three CHAP processors.

Acknowledgments

During the course of this research, discussions with D. T. Ling, C. N. Liu, and R. H. Wolfe of the Veridical User Environment group at IBM Research have been very helpful. At IBM, Essex Junction, Vermont, we wish to thank Jack Mandelman, Toshi Furukawa, and Don Kenney for discussions of device physics, and Jeff Johnson and Steve Furkay for a great deal of assistance with the new version of the FIELDAY device simulator. One of the Vermont authors (D. F.) wishes to thank the Research authors for their efforts in preparing figures for an external presentation.

References

1. *Chapel Hill Workshop on Volume Visualization*, C. Upson, Ed., Department of Computer Science, University of North Carolina, Chapel Hill, May 1989.
2. W. E. Robbins and S. S. Fisher, Eds./chairpersons, *Three-Dimensional Visualization and Display Technologies, Proc. SPIE* **1083**, International Society for Optical Engineering, Bellingham, WA, 1989.
3. E. J. Farrell, Ed./chairperson, *Extracting Meaning from Complex Data, Proc. SPIE* **1259**, International Society for Optical Engineering, Bellingham, WA, 1990.
4. E. J. Farrell, Ed./chairperson, *Extracting Meaning from Complex Data II, Proc. SPIE* **1459**, International Society for Optical Engineering, Bellingham, WA, 1991.
5. P. A. Appino and E. J. Farrell, "MOVIETOOL: A System for Making Scientific Videotapes," *Extracting Meaning from Complex Data II*, E. J. Farrell, Ed., *Proc. SPIE* **1459** (1991).
6. R. A. Drebin, L. Carpenter, and P. Hanrahan, "Volume Rendering," *Computer Graph.* **22,** No. 4, 65–74 (1988).
7. E. K. Fishman, B. Drebin, D. Magid, W. W. Scott, D. R. Ney, A. F. Brooker, L. H. Riley, J. A. St. Ville, E. A. Zerhouni, and S. S. Siegelman, "Volumetric Rendering Techniques: Applications for Three-Dimensional Imaging of the Hip," *Radiology* **163,** 737–738 (1987).
8. M. Levoy, "Display of Surfaces from Volume Data," *IEEE Computer Graph. & Appl.* **8,** 29–37 (1988).
9. C. Upson and M. Keeler, "V-BUFFER: Visible Volume Rendering," *Computer Graph.* **22,** No. 4, 59–62 (1988).
10. D. S. Schlusselberg, W. K. Smith, and D. J. Woodward, "Three-Dimensional Display of Medical Image Volumes," *Proceedings of the 7th Annual Conference of the National Computer Graphics Association*, National Computer Graphics Association, McLean, VA, 1986, pp. 114–123.
11. D. S. Schlusselberg, T. R. Simon, W. K. Smith, and D. J. Woodward, "Three-Dimensional Analysis and Display of Regional Cerebral Blood Flow," *Proceedings of the 8th Annual Conference of the National Computer Graphics Association*, National Computer Graphics Association, McLean, VA, 1987, pp. 104–111.
12. E. J. Farrell and R. A. Zappulla, "Three-Dimensional Data Visualization and Biomedical Applications," *CRC Critical Rev. Biomed. Res.* **16,** No. 4, 323–364 (1988).
13. E. J. Farrell and Z. D. Christidis, "Visualization of Complex Data," *Three-Dimensional Visualization and Display Technologies*, W. E. Robbins and S. S. Fisher, Eds., *Proc. SPIE* **1083,** 153–160 (1989).
14. P. A. Appino and E. J. Farrell, "A Visualization Tool for the Scientist/Engineer," *Extracting Meaning from Complex Data*, E. J. Farrell, Ed., *Proc. SPIE* **1259,** 61–71 (1990).
15. F. P. Brooks, Jr., "Grasping Reality Through Illusion—Interactive Graphics Serving Science," *Proceedings of Special Interest Group on Computer/Human Interaction*, ACM, New York, 1988, pp. 1–11.
16. C. Upson, T. Faulhaber, Jr., D. Kamins, D. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, and A. van Dam, "The Application Visualization System: A Computational Environment for Scientific Visualization," *IEEE Computer Graph. & Appl.* **9,** No. 4, 30–42 (1989).
17. L. Treinish, "An Interactive Discipline-Independent Data Visualization System," *Comput. Phys.* **3,** No. 4, 55–64 (1989).
18. R. B. Haber, "Visualization in Engineering Mechanics: Techniques, Systems, and Issues," *Visualization Techniques in the Physical Sciences*, SIGGRAPH 88 Course Notes, R. S. Wolff, Ed., ACM, New York, 1988.
19. L. Gelberg, D. Kamins, and J. Vroom, "VEX: A Volume Exploratorium," *Proceedings of the Chapel Hill Workshop on Volume Visualization*, C. Upson, Ed., University of North Carolina, Chapel Hill, 1989, pp. 21–26.
20. R. L. McPherron, "A Self-Documenting Source-Independent Data Format for Computer Processing of Tensor Time Series," *Phys. Earth & Planet. Inter.* **12,** 103–111 (1976).
21. A. Q. Smith and C. R. Clauer, "A Versatile Source-Independent System for Digital Data Management," *EOS Trans. Amer. Geophys. Union* **67,** No. 15, 188–189 (1986).
22. L. A. Treinish and M. L. Gough, "A Software Package for the Data-Independent Management of Multidimensional Data," *EOS Trans. Amer. Geophys. Union* **68,** 633–635 (1987).
23. C. J. Date, *An Introduction to Database Systems*, Third Edition, Addison-Wesley Publishing Co., Reading, MA, 1981.
24. J. D. Ullman, *Principles of Database Systems*, Second Edition, Computer Science Press, Rockville, MD, 1982.
25. G. Wiederhold, *Database Design*, McGraw-Hill Book Co., Inc., New York, 1983.
26. B. A. Myers, "A Taxonomy of Window Manager User Interfaces," *IEEE Computer Graph. & Appl.* **8,** No. 5, 65–84 (1988).
27. J. R. Brown and S. Cunningham, *Programming the User Interface*, John Wiley & Sons, Inc., New York, 1989.
28. D. L. Lauer, *Design Basics*, Holt, Rinehart & Winston, New York, 1979.
29. E. J. Farrell, "Visual Interpretation of Complex Data," *IBM Syst. J.* **26,** No. 2, 174–200 (1987).
30. S. M. Jaffey, K. Dutta, and L. Hesselink, "Digital Reconstruction Methods for Three-Dimensional Image Visualization," *Processing and Display of Three-Dimensional Data II*, J. J. Pearson, Ed., *Proc. SPIE* **507,** 155–163 (1984).
31. D. R. Peachy, "Solid Texturing of Complex Surfaces," *Computer Graph.* **19,** No. 3, 279–286 (1985).
32. Ken Perlin, "An Image Synthesizer," *Computer Graph.* **19,** No. 3, 152–161 (1985).
33. E. J. Farrell, S. E. Laux, P. L. Corson, and E. M. Buturla, "Animation and 3D Color Display of Multiple-Variable Data: Application to Semiconductor Design," *IBM J. Res. Develop.* **29,** 302–315 (1985).
34. P. Sabella, "A Rendering Algorithm for Visualizing 3D Scalar Fields," *Computer Graph.* **22,** No. 4, 51–58 (1988).
35. K. Kazufumi, K. Harada, E. Nakamae, M. Yasuda, and A. G.

Sato, "Reconstruction and Semi-Transparent Display Method for Observing Inner Structure of Multiple Surfaces," *Visual Computer* **3**, No. 3, 137–144 (1987).
36. G. Russell and R. B. Miles, "Display and Perception of 3-D Space-Filling Data," *Appl. Opt.* **26**, No. 6, 973–982 (1987).
37. S. E. Wixson and R. C. Garrett, "The Display of Cardiovascular Data with Focal Depth Cues," *Proceedings, IEEE Computers in Cardiology*, IEEE Computer Society, New York, 1986, pp. 1–3.
38. S. E. Wixson, "True 4D Graphics Laboratory for Non-Geometric Grid Based Data," *True Three-Dimensional Imaging Techniques and Display Technologies*, A. McAllister and V. Robbins, Eds., *Proc. SPIE* **761**, 179–182 (1987).
39. R. Lenz, B. Gudmundsson, B. Lindskog, and P. E. Danielsson, "Display of Density Volumes," *IEEE Computer Graph. & Appl.* **6**, 20–29 (1986).
40. M. B. Merickel, C. S. Carman, J. R. Brookeman, J. P. Mugler, M. F. Brown, and C. R. Ayers, "Identification and 3-D Qualification of Atherosclerosis Using Magnetic Resonance Imaging," *Comput. Biol. Med.* **18**, No. 2, 89–102 (1988).
41. M. Vannier, R. Butterfield, R. Rickman, D. Jordan, W. Murphy, and P. Biondetti, "Multispectral Magnetic Resonance Image Analysis," *CRC Critical Rev. Biomed. Eng.* **15**, No. 2, 117–144 (1987).
42. B. Davari, C. Koburger, T. Furukawa, Y. Taur, W. Noble, A. Megdanis, J. Warnock, and J. Mauer, "A Variable-Size Shallow Trench Isolation (STI) Technology with Diffused Sidewall Doping for Submicron CMOS," *IEDM Tech. Digest*, pp. 92–95 (1988).
43. N. Shigyo, T. Wada, S. Fukuda, K. Hieda, T. Hamamoto, H. Watanabe, K. Sunouchi, and H. Tango, "Steep Subthreshold Characteristic and Enhanced Transconductance of Fully-Recessed Oxide (Trench) Isolated 1/4 μm Width MOSFETs," *IEDM Tech. Digest*, pp. 636–639 (1987).
44. T. Furukawa and J. Mandelman, "Process and Device Simulation of Trench Isolation Corner Parasitic Device," presented at the 1988 Electrochemical Society Fall Meeting.
45. K. Hieda, F. Horiguchi, H. Watanabe, K. Sunouchi, I. Inoue, and T. Hamamoto, "Effects of a New Trench-Isolated Transistor Using Sidewall Gates," *IEEE Trans. Electron Devices* **36**, 1615–1619 (1989).
46. D. Foty, J. Mandelman, and T. Furukawa, "Behavior of an NMOS Trench Isolated Corner Parasitic Device at Low Temperature," presented at the 1989 Electrochemical Society Fall Meeting.
47. E. M. Buturla, P. E. Cottrell, B. M. Grossman, and K. A. Salsburg, "Finite-Element Analysis of Semiconductor Devices: The FIELDAY Program," *IBM J. Res. Develop.* **25**, 218–231 (1981).
48. D. Cole and J. Johnson, "Accounting for Incomplete Ionization in Modeling Si-Based Semiconductor Devices," *Proceedings of the Workshop on Low Temperature Semiconductor Electronics*, Institute of Electrical and Electronics Engineers, 1989, pp. 73–77.
49. E. Farrell, T. Aukrust, and J. Oberhuber, "Vector and Scalar Field Interpretation," *Extracting Meaning from Complex Data II*, E. J. Farrell, Ed., *Proc. SPIE* **1459** (1991).
50. G. Grinstein and S. Smith, "The Perceptualization of Scientific Data," *Extracting Meaning from Complex Data*, E. J. Farrell, Ed., *Proc. SPIE* **1259**, 190–199 (1990).
51. W. M. Coughran and E. Grosse, "Scientific Animation Techniques," *Extracting Meaning from Complex Data*, E. J. Farrell, Ed., *Proc. SPIE* **1259**, 71–81 (1990).
52. D. A. Rabenhorst and E. J. Farrell, "Auditory Perception of Experimental Data," *Extracting Meaning from Complex Data*, E. J. Farrell, Ed., *Proc. SPIE* **1259**, 147–153 (1990).
53. *SunView™ 1 Programmer's Guide, Revision A*, Sun Microsystems, Inc., Mountain View, CA, May 9, 1988.
54. *Programmer's Reference Manual for SunWindows™, Revision 1*, Sun Microsystems, Inc., Mountain View, CA, May 15, 1985.
55. *PIXAR Image Computer ChapLibraries User's Guide, ChapLibraries Release 2.0*, PIXAR Corporation, San Rafael, CA, 1987.

Received November 22, 1989; accepted for publication October 4, 1990

Edward J. Farrell *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 704, Yorktown Heights, New York 10598.* Mr. Farrell received his B.S. from the University of Minnesota with a major in physics, and subsequently completed his doctoral course work in mathematical statistics. From 1959 to 1964 and 1966 to 1968, he was with UNIVAC. His work included reliability analysis of complex digital systems, waveform classification for sonar target identification, and spacecraft guidance system calibration and evaluation analysis. Mr. Farrell was at the Research Division of Control Data Corporation from 1964 to 1966. There, his research concerned the accuracy limits of scanning optical sensors for celestial navigation, and the design and performance evaluation of a star occultation scanner used for the navigation of the GEMINI vehicle. In 1968, Mr. Farrell joined the IBM Thomas J. Watson Research Center; he is a Research Staff Member in the Computer Sciences Department. His investigations have included the classification of patient recovery following open-heart surgery based on statistical and functional-simulation methods, medical ultrasonic and CT tomography, and methods for 3D color display. Mr. Farrell's current interest is the interactive visual interpretation of multidimensional data-based 3D color imaging with animation, multiple window display methods, and stereographics. His research addresses a range of problems in several engineering, medical, and scientific areas. Mr. Farrell is a member of the International Society for Optical Engineering, the IEEE Computer Society, and the National Computer Graphics Association.

Perry A. Appino *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 704, Yorktown Heights, New York 10598.* Dr. Appino received a B.S. in environmental chemistry from Lehigh University in 1979 and a Ph.D. in operations research from The Johns Hopkins University in 1984. He then joined the IBM Thomas J. Watson Research Center to work on tools for graphics postprocessing and image processing. Dr. Appino is currently a Research Staff Member in the Computer Sciences Department, working in the area of scientific visualization. His interests include user interface design, volume visualization, and virtual realities for data visualization. Dr. Appino is a member of the Association for Computing Machinery and the National Computer Graphics Association.

Daniel P. Foty *IBM General Technology Division, Essex Junction, Vermont 05452.* Dr. Foty received B.S. degrees in physics and in chemistry from Bates College in 1983, the M.S. degree in electrical engineering from the University of Illinois in 1985, and the Ph.D. degree in materials science from the University of Vermont in 1990. In March 1988 he joined the IBM Essex Junction, Vermont, laboratory, where he has been working on device physics in the semiconductor base technology area. His current interests are in three-dimensional effects on device behavior, novel structures and device physics requiring understanding beyond the drift-diffusion approximation, low-temperature electronics, and optical pulse propagation in dispersive media. In addition to numerous publications and conference presentations in these fields, Dr. Foty is the lead organizer for the Symposium on Low Temperature Electronic Device Operation, sponsored by the Electrochemical Society and currently scheduled for 1991 and 1993. Dr. Foty is a member of the Electrochemical Society and of the IEEE Electron Devices and Laser and Electro-optics societies.

Thomas D. Linton, Jr. *IBM General Technology Division, Essex Junction, Vermont 05452.* Dr. Linton is a staff engineer in the Technology Simulation group, working on software tools for simulation of semiconductor processes and devices. He received the B.A. degree in philosophy from Calvin College, Grand Rapids, Michigan, in 1976, the B.S.E.E. degree from the University of Michigan at Ann Arbor in 1981, the M.S.E.E. degree from Stanford University in 1983, and the Ph.D. in electrical engineering from the University of Texas at Austin in 1989. From 1983 to 1985 he was a device engineer in the MOS-8 wafer fabrication facility at Motorola in Austin, Texas. From 1985 to 1988 he was a Member of Technical Staff at the Microelectronics and Computer Technology Corporation in Austin, where he worked in the area of three-dimensional semiconductor device simulation. Dr. Linton joined IBM in 1989.