Applications of Large-Scale Computing and Scientific Visualization in Medicine

C.R. Johnson, D.M. Beazley, Y. Livnat, S.G. Parker,

J.A. Schmidt, H.W. Shen, and D.M. Weinstein

The authors are with the Department of Computer Science, University of Utah, Salt Lake City, UT 84112
E-Mail: (crj,beazley,ylivnat,sparker,jas,hwshen,dweinste)@cs.utah.edu Web: http://www.cs.utah.edu/~sci/

Abstract

In this paper we describe applications of large-scale computing and scientific visualization software tools to problems in computational medicine. We focus the application of these software tools on a class of bioelectric field problems that arise in cardiology and neuroscience.

Keywords

Computational Medicine, Computational Steering, Large-Scale Computing, Modeling, Software Tools, Visualization.

I. Introduction

Computer modeling and simulation continue to grow in importance in many fields of medicine. The reasons for this growing importance are manifold. First, mathematical modeling has been shown to be a substantial tool for the investigation of complex biophysical phenomena [1], [2]. Second, since the level of complexity one can model parallels existing hardware configurations and software tools, advances in computer architecture and application software have made it feasible to apply the computational paradigm to increasingly accurate models of biophysical systems. Hence, while biological complexity continues to outstrip the capabilities of even the largest computational systems, the computational methodology has taken hold in medicine and has been used successfully to suggest physiologically and clinically important scenarios and results.

This paper provides an overview of software tools developed by the Scientific Computing and Imaging (SCI) research group at the University of Utah for solving large-scale problems in computational medicine. We describe software tools for geometric modeling, computation, visualization, and computational steering. We focus the application of these software tools on bioelectric field problems that arise in cardiology and neuroscience.

II. Modeling

In most computational science applications, considerable geometric modeling must take place prior to simulation and visualization. Modeling efforts usually take the form of approximating a continuous physical structure with a discrete collection of nodes connected into polygons (polyhedra).

In this section we describe two (often essential) modeling procedures, segmentation and mesh generation. From MRI scans, we have constructed large-scale torso and head models, within which we solve bioelectric field problems, as shown in Figure 1 for example. To classify the relevant tissues we have developed two semi-automatic segmentation algorithms [3], [4]. To construct three-dimensional surfaces and/or volumetric meshes that are used for computation and visualization, we utilize two- and three-dimensional Delaunay triangulation algorithms [5], [6], [7].

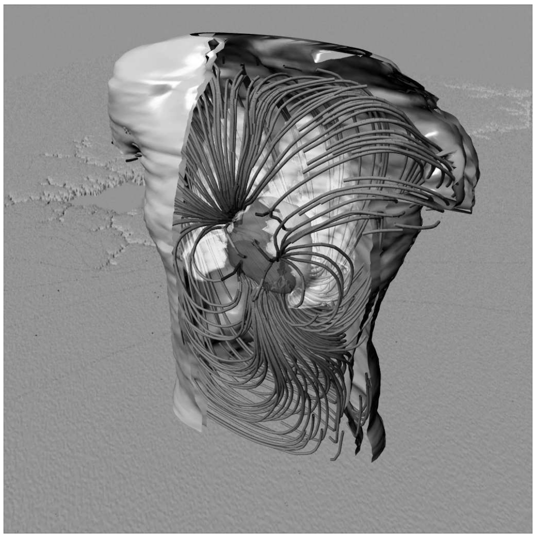

Fig. 1. A representation of the geometry and electrical flow in a model of the human thorax. The model was created from MRI images of a patient. Shown are segments of the body surface, the heart, and lungs. The colored loops represent the flow of electric current through the thorax for a single instant in time, computed from voltages recorded from the surface of the heart during open chest surgery.

A. Segmentation

Image segmentation, the process of defining boundary domains in 2D and 3D images, takes on an important role in building biomedical models. Before surface reconstruction, mesh generation, and other modeling operations begin, the scientist must define the boundaries of interesting (relevant) regions within the images. In spite of extensive research in the field, there is still no algorithm that can automatically find region boundaries unfailingly from clinically obtained medical images. There are two reasons for this. One is that most of the image segmentation algorithms are still noise sensitive. The second reason is that most segmentation tasks require certain specialized background knowledge about the region(s) of interest as modelers do not want to include (nor can include because of size restrictions) the detail that are yielded from most image segmentation algorithms.

To ensure the accuracy of the segmentation, many researchers still manually segment images by inserting control points on the images by hand and then invoking data fitting algorithms to fit curves. Because a "typical" large-scale model has complex geometry, model construction requires the processing of a large number of images. Furthermore, as suggested by the recent work of Kikinis et al. [8], physicians are interested in near real-time segmentation for use in time-critical (such as surgery) healthcare applications. This makes the segmentation process one of the most signficant challenges in the biomedical modeling process.

We have developed a semi-automatic image segmentation tool that combines conventional manual segmentation utilities with a novel automatic image segmentation algorithm. To achieve manual segmentation, the researcher "drops" control points, the program then automatically fits cubic splines to the selected points. In automatic segmentation mode, the researcher first selects a particular boundary and then using a bimodal thresholding algorithm within a local window of the target image, the program produces boundary pieces. The individual pieces are then connected to yield the entire region boundary. By combining these two segmentation methods, a user can obtain accurate boundary descriptions with a minimum of effort [3].

The manual segmentation procedure is straightforward. Given a 2D image slice, the user places numerous control points on a visible region boundary, then invoks a data fitting algorithm to interpolate between these control points to generate a region contour. Several different data fitting methods can be used. The methods differ based on either the degree of interpolation function used – i.e. linear or quadratic or on the basis function used – i.e. B-spline, Bezier spline, etc.

We developed an automatic algorithm that allows users to interactively steer the segmentation process. The program begins by prompting the user to select an initial starting point on the region boundary. From that point, local edge detection and contour following programs are used to find the region boundaries. The contour algorithm gives only a small contour segment piece at a time. The program then waits for feedback from the user. The user may inspect and if necessary, correct the identified contour segment. If no corrections are needed, the program continues to find the next contour piece.

To determine the boundary segments in the local region, we use a bimodal thresholding algorithm. When the user selects an initial point, the program places a small local window around this point. The program then determintes the values of pixels located inside the window and uses these values to compute a local histogram. If part of a region boundary passes through the local window, the histogram should have a bimodal appearance. The local minimum between these two peaks determines the threshold value used to segment the local image. After a boundary segment is found, the local window then automatically moves in the direction of the contour to the next position.

While our original segmentation software tool used a technique that considered a single grey-scale value for each voxel, we have recently created a new software module, called *VecSeg*, that extends the previously segmentation technique to include an n-dimensional vector for each voxel. This technique requires multiple volume data sets, all with identical spatial domains, as input. Ideally, each volume would show different properties of the material, and the resultant vectors could be easily segmented into the correct material types. In our example with MRI scans, the patient underwent two scans with different "weightings." An MRI weighting specifies the frequency of the magnetic pulse, the duration of the pulse, and the relaxation time between pulses. These parameters determine the "material to intensity mapping" that will result from the scan. For this example, we have two volumes, each containing 81 256×256 MRI slices from the same patient.

Initially, *VecSeg* reads in the data files and passes the volume data to clipping modules. The clipping modules are linked so that they will always define the same volume (i.e. if you move one volume's clipping planes, the others' move as well). These modules enable the researcher to segment the volume quickly and help to determine the best material ranges with a high degree of interactivity. They enable the user to interact with a small subset of the domain initially, and then, once the desired ranges have been determined, to pass the entire volume in to be segmented.

The next stage in the pipeline is the segmenting module, which accepts the clipped volume fields as input. The interface for this module is a 2D array of range sliders for specifying grey-scale values of materials from each input volume. In this case, the module accepts two fields as input and segments for several distinct materials. If a voxel's vector is contained within the space spanned by the ranges of that material, that voxel is set to be "on" for that material.

Finally, vector segmentation module creates a field of n-bit numbers, with one bit corresponding to each material. This field is passed on to a final module that extracts surfaces for each material volume, as well as for a volume of colored points corresponding to each voxel's identified material. The volume of points is a "quick-and-dirty" method of volume-rendering the segmented data, and is a fast way to look at the segmentation results. The module can also output boundary surfaces for each material.

B. Mesh Generation

Most numerical methods for solving boundary value problems require that the continuous domain be broken up into discrete elements, the so-called *mesh* or *grid*, which one can use to approximate the governing equation(s) using the particular numerical technique (finite element, boundary element, finite difference, or multigrid) best suited to the problem.

Because of the complex geometries and the enormous number of degrees of freedom (upwards to tens of millions elements) associated with many problems in computational medicine (and specifically the bioelectric field problems considered here), construction of the mesh can become one of the most time consuming aspects of the modeling process. After deciding upon the particular approximation method to use (and the most appropriate type of element), we need to construct a mesh of the solution domain that matches the number of degrees of freedom of our fundamental element.

There are several different strategies for discretizing geometry into fundamental elements.

Bioelectric field simulations require the modeling of complex geometric domains such as those found in the human thorax and head (eg., heart, lungs, skeletal muscle, vasculararture, body surface, brain, and skull). These different anatomical structures are extremely irregular and do not permit the efficient use of standard CAD/CAM descriptors commonly used to describe synthetic structures. Instead, the geometric objects that comprise the geometric model are described by segmented sets of 2D or 3D MRI or CT images consisting of contours or surfaces at the boundaries between organs and/or regions of different conductivity. The resulting contour or surface data is then used to construct a polyhedral representation of the solution domain.

For the construction and simulation of anatomically correct models we utilized unstructured meshes of triangular (surfaces) or tetrahedral (volumes) elements. Unstructured grids tend to capture the complex structures of human and animal anatomy more efficiently than structured grids. Meshes used for the research presented in the Selected Applications section range in size from thousands of elements for 2D models to tens of millions of elements for 3D models.

The method we used to create meshes is based upon the Delaunay tessellation algorithm originally proposed by Watson [6] and later extended by Weatherhill [7]. The Delaunay criterion states that the circumsphere of any tetrahedron (triangle)[1] contains no other mesh points. The thrust of the Watson/Weatherhill algorithm is to efficiently insert a point into an existing grid (bounding simplex) in such a way that the Delaunay criterion is met. Certain tesselations are then deleted and new ones are subsequently created using the new point and a subset of the the old points. The general method is applicable to N-dimensions, although engineering applications usually require implementations in two or three dimensions [9], [5]. A more detailed description of our mesh generation algorithm can be found in [5].

The mesh generation procedure consists of five steps as outlined in the following algorithm:

Mesh Generation Algorithm

Inputs: Boundary point representation (in contours)

Begin

Triangulate Surfaces

Construct Coarse Mesh - Delaunay Tessellation

Determine Interior Tetrahedra

Repeat:

Generate Interior Point

[1]In describing the algorithm, we use triangle or tetrahedron interchangeably.

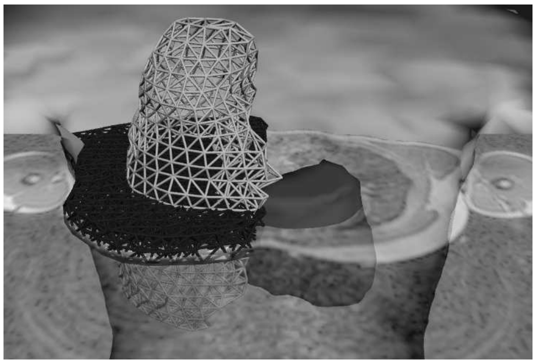

Fig. 2. The model contruction process. Starting from MRI scans, the geometric model is constructed by segmenting the image, triangulating surfaces (such as the lung), and automatically tetrahedralizing the volume.

Tessellate Interior Point

Until (point fails spacing and degree tests)

Classify Regions

End

Starting with the boundary points extracted during the segmentation, one then proceeds to adequately represent the surface mesh (line segments in two-dimensions and triangles in three-dimensions). The next step is to construct a coarse mesh of tetrahedra from the boundary points and then to determine tetrahedra within the surface of interest. These interior tetrahedra are used in the fourth step, which iterates between the generation of a new point and subsequent tesselation until certain spacing criteria are satisfied. The spacing criteria is what ultimately determines the size of the mesh.

The final step is to classify the tetrahedra by their material properties, which vary depending on whether the tetrahedra are considered to be in the heart, lungs, etc. Since the material property of interest in bioelectric problems, electric conductivity, can be anisotropic, each tetrahedron must be assigned a tensor describing local conductivity. To classify a tetrahedron we had to localize the position of its centroid relative to the surfaces separating regions of different conductivity. To localize the element, we utilized a ray tracing approach [5] that projects a ray from the centroid of an element to a point at infinity and counts the instances the ray intersects with a triangulated surface. If the ray crosses through one triangulated surface an odd number of times, the point is considered to be interior to the surface; conversely, if it crosses the surface an even number of times, the tetrahedron is considered to be exterior to the surface. A composite of the segmentation and mesh generation processes are shown in Figure 2.

III. Computation

Over the past several years we have developed several programs to solve computationally intensive problems in medicine (focusing primarily on bioelectric field problems), such as (1) interactive programs to construct, manipulate, and display large scale, three-dimensional surface and volumetric meshes [10], [11], [12], [13]; (2) three-dimensional, adaptive, finite element programs for solving partial differential equations with general boundary conditions and source terms [10], [14], [15], [5], [16]; and (3) programs that solve the resulting large system of equations and simultaneously apply regularization to ill-posed inverse problems [10], [17]. Because our work involves large-scale problems, much of the software we have developed has been designed to take advantage of parallel processing via high performance architectures. One area of recent research involves adaptively refining large-scale unstructured finite element meshes to improve solution accuracy.

A. Adaptive Methods

Discrete approaches to bioelectric field problems have usually centered around classical numerical techniques for solving partial differential equations: finite difference, finite element, and boundary element methods. To date, instead of developing and utilizing specific, quantitative methods, most biomedical scientists have assigned spatial discretization levels by relying largely on previous experience and their common sense, utilizing what has apparently worked before and perhaps reducing the discretization level in regions *a priorily* known to contain high gradients. This situation has arisen, at least in part, because of the computational load represented by the large, complex, inhomogeneous, even anisotropic geometries that characterize many bioelectric field problems.

We have found (as others have), however, that by using *a posteriori* estimates from the finite element approximation of the governing equations, we can locally refine the mesh discretization and reduce the errors in the direct solution. It has been assumed — and our findings support this notion — that improving the accuracy of the direct solution also improves the subsequent inverse solution. The novel aspect of our approach is that it uses local approximations of the error in the numerical solutions to drive an automatic adaptive mesh refinement. The basis of the error estimations comes from recent research on the finite element method [18].

The essential feature of the finite element method is that the approximate forms of the scalar field satisfies the governing equation in each element in some weighted sense. If we reduce the size of the elements, *h-refinement*, or increase the order of the basis function, *p-refinement*, we increase the accuracy of the approximation [19]. While either approach can be applied globally, computational limits make it more efficient to apply refinement locally, to regions where it is deemed most beneficial. One way to monitor the overall effect of refinement is to compute the total energy, which must converge monotonically if refinement is progressing effectively.

Improvements via an adaptive h-refinement technique can be impelmented by using an estimate of the element energy error, such as the one described here derived from methods suggested by Lewis [20].

The error in the potential, e_Φ, is defined as the difference between the exact potential, Φ, and the calculated potential, $\hat{\Phi}$.

$$e_\Phi = \Phi - \hat{\Phi} \tag{1}$$

Similarly, the error in the gradient of the potential (electric field), q, is

$$e_q = q - \hat{q} \tag{2}$$

where $q = \nabla\Phi$ and $\hat{q} = \nabla\hat{\Phi}$. The error norm is defined as:

$$||e_q|| = (\int_\Omega (\nabla\Phi - \nabla\hat{\Phi})^T \sigma(\nabla\Phi - \nabla\hat{\Phi}) d\Omega)^{\frac{1}{2}}. \tag{3}$$

Zienkiewicz [21], [22] has shown that

$$\int_\Omega (\nabla\Phi)^T \sigma(\nabla\hat{\Phi}) d\Omega = \int_\Omega (\nabla\hat{\Phi})^T \sigma(\nabla\Phi) d\Omega = \int_\Omega (\nabla\hat{\Phi})^T \sigma(\nabla\hat{\Phi}) d\Omega. \tag{4}$$

Using this result, (3) becomes

$$||e_q||^2 = \int_\Omega (\nabla\Phi)^T \sigma(\nabla\Phi) d\Omega - \int_\Omega (\nabla\hat{\Phi})^T \sigma(\nabla\hat{\Phi}) d\Omega \tag{5}$$

or equivalently,

$$||e_q||^2 = ||q||^2 - ||\hat{q}||^2 \tag{6}$$

Here, $||q||^2$ is a measure of the total energy in the domain. Using this, the percentage error, η, can be defined to be

$$\eta = \frac{||e_q||}{||q||} \cdot 100\% \tag{7}$$

This error measure gives a ratio of the energy difference and the total energy which is in effect a percentage error for each element.

Here we have used linear basis functions to describe the variation of the potential in each tetrahedral element. At every point in the mesh, the potential is uniquely specified. However, the gradient is not specified at every point in space but is defined to be constant over each element and discontinuous across element boundaries. Zienkiewicz showed that when the current densities arising from the gradients are globally smoothed, they provide a more accurate estimate of the energy than the energy using the constant current densities.[21], [22] Hence, (7) can be calculated using the energy due to the smoothed currents and the energy due to the constant currents. The smoothed current densities are now defined at the same nodal locations as the potentials and are continuous across element boundaries. The smoothing process uses the Galerkin technique [20] and involves minimizing the difference between the two current densities,

$$\sigma(\nabla\tilde{\Phi} - \nabla\hat{\Phi}) \tag{8}$$

where $\nabla\tilde{\Phi}$ is the smoothed gradient representing the exact gradient and $\nabla\hat{\Phi}$ is the constant gradient from the finite element solution. This method produces a system of equations in the following form:

$$(\int_{\Omega_e} \Psi_i \Psi_j d\Omega_e)(\sigma\nabla\tilde{\Phi}_i) = (\int_{\Omega_e} \sigma\nabla\hat{\Phi}_i \Psi_i d\Omega_e). \tag{9}$$

where Ψ_i is a linear basis function defined at node i, $\nabla\tilde{\Phi}_i$ is the value of the smoothed gradient at that point, σ is the conductivity, and $\nabla\hat{\Phi}_i$ is the constant gradient resulting from the finite element solution. In equation (9), the current density, $\sigma\nabla\tilde{\Phi}_i$ is the unknown that is solved for at each node, i of the grid. Thus we calculate a new estimate, $\tilde{q}$, which can be used in place of the exact gradient, q. The current density now varies linearly in each tetrahedron and is continuous everywhere.

Once the energy error has been computed, the mesh refinement can begin. The error must be related to some parameter which guides the mesh refinement. We utilized a spacing function, h_e, which controls the size and number of elements [20]. This spacing function, h_e is defined to be the linear interpolation of the spacing values at the four nodes of the tetrahedron. A variable β is defined as the ratio of the element error to the average of the element errors:

$$\beta_e = \frac{||e_q||_e}{||\hat{e}_q||_e}. \tag{10}$$

If $\beta_e > 1$, then a new spacing function can be defined as

$$h^* = \frac{h}{\beta_e}. \tag{11}$$

Thus, the spacing values at each node of the element are reduced. This new spacing function will then increase the number of points in the regions of large errors. This whole process is repeated until the global error estimate falls below the specified level.

The algorithm does not depend on how the modification was made such that a different error estimate other than the one just described can be used by only modifying the spacing values in some fashion. This allows any number of error estimators to be used to guide the adaptive mesh refinement process.

IV. Visualization

Visualization systems play an integral role in all phases of computational modeling, and we have developed a variety of application programs based on several common visualization paradigms [11], [12], [13], [23]. For high quality ray-traced images, we have developed graphics systems which exploit parallel processing and/or clusters of workstations. We have recently developed differential volume rendering and isosurface extraction techniques that allows nearly real-time rendering of three-dimensional flows and isosurface extraction for models with hundreds of thousands or millions of elements [24], [25], [26].

A. Isosurface extraction

Isosurface extraction is a powerful tool for investigating volumetric scalar fields. The position of an isosurface, as well as its relation to other neighboring isosurfaces, can provide clues to the underlying structure of the scalar field. In medical imaging applications, isosurfaces permit the extraction of particular anatomical structures and tissues. These isosurfaces are static in nature. A more dynamic use of isosurfaces is called for in many scientific computing applications, such as simulation of the electric field in a human torso during a defibrillation shock. In these applications, scientists need to be able to dynamically change the isovalue in order to gain better insight into simulation results.

As scientific computation demands higher accuracy and state-of-the-art medical scanners increase in resolution, the resulting data sets (millions of voxels) for visualization expand rapidly. The sheer size of these data sets, as well as their structure, pose major obstacles for interactive investigation. While medical imaging data is provided at structured grid positions, scientific data sets frequently consist of geometry represented by unstructured finite element grids.

Originally, isosurface extraction methods were restricted to structured grid geometry and earlier effort focused on extracting a single isosurface [27]. Recently, in an effort to speed up isosurface extraction, several methods were developed that could be adapted to multiple isosurface extraction from structured [28], [29] as well as unstructured geometry [30], [31]. Nevertheless, these methods do not provide interactive speed, especially for unstructured grids. Defining n as the number of data cells and k as the number of cells intersecting a given isosurface, the above algorithms usually have time complexity of $O(n)$. Although [28] has an improved time complexity of $O(k \log(\frac{n}{k}))$, it is only suitable for rectangular grids.

Current isosurface extraction methods are based on locating data cells intersecting the isosurface and then approximating the isosurface inside these cells. To accelerate this search the data cells are reorganized via pre-processing. For unstructured grids, the cells are organized based on the minimum and maximum values attained inside the cells, while geometric coherence is exploited for structured grids.

Our research primarily involves unstructured geometry and often requires interactive investigation of large data sets. Toward this goal we have developed two rapid isosurface extraction algorithms that work for both structured and unstructured grids.

The *Sweeping Simplices* [25] method accelerates the search by utilizing coherence between isosurfaces with close isovalues. In addition to sorting the data cells into two lists based on their minimum and maximum values, the sweeping simplices method employs an array of flags that signal data cells with a minimum value below the current isovalue. A neighboring isosurface can be found quickly by consulting the two sorted lists and *sweeping* through these flags.

Recently we developed a new representation for the underlying domain of the isosurface extraction problem, which we termed the *Span Space* [26]. Traditionally, the data cells of unstructured grids were viewed as intervals over the real line, $\mathbb{R}$, where the extrema of the interval are the minimum and maximum values attained in the cell. Isosurface extraction was then viewed as locating those intervals that include the given isovalue. The span space approach, on the other hand, is to view the data cells as points in the two dimensional space, $\mathbb{R}^2$, where the coordinates of a point associated with a cell are the minimum and maximum values as before. The isosurface extraction is achieved by locating all the points which are confined to the semi-infinite rectangle $(-\infty, v) \times (v, \infty)$, where v is the given isovalue.

Posing the isosurface extraction problem over the Span Space, we developed a *Near Optimal IsoSurface Extraction* algorithm (*NOISE*) [26], which has a worst case performance of only $O(\sqrt{n}+k)$. Considering the fact that a typical isosurface intersects $O(n^{\frac{2}{3}})$ cells, this algorithm is clearly optimal in the general case. The NOISE algorithm is based on a Kd-tree, which can be built in a preprocessing stage and quickly loaded at run time. Once the Kd-tree is built, the algorithm can rapidly answer queries requesting it, for example, to extract an isosurface, determine the number of cells intersecting a given isosurface, or compute rough estimates of the surface area of the isosurface as well as the volume encompassed by that isosurface. Using the NOISE method, we are able to extract isosurfaces in less then 50ms (before rendering) for an unstructured data set with over a million cells. Queries for the number of cells intersecting a given isosurface are answered in only 3ms.

In order to restrict the worst case performance of the sweeping simplices algorithm, a data decomposition scheme is introduced to subdivide the data cells into different groups. Since only a subset of the cells groups now need to be examined for a given isovalue, this scheme further speeds up the algorithm. Recently this algorithm has been implemented on a 64 node Cray T3D using a novel data distribution scheme. Preliminary results have shown that the load imbalance among all available processing elements can be limited to 2-3% of the total workload. Example images created from the Sweeping Simplices and NOISE algorithms are shown in Figures 3 and 4.

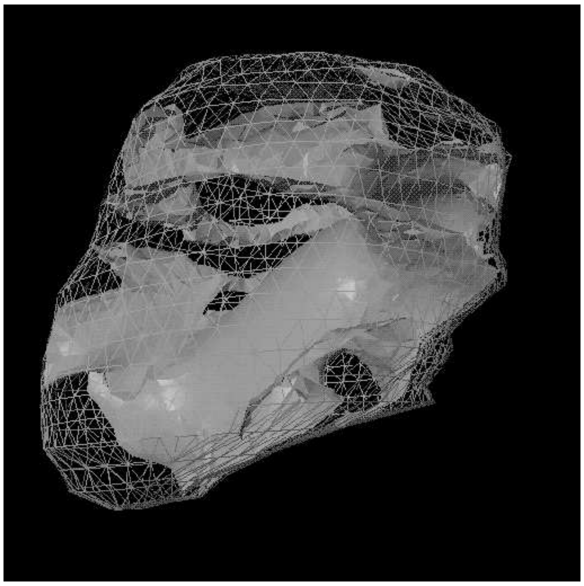

Fig. 3. Heart: Isosurfaces of constant voltage from a finite element simulation of cardiac defibrillation within the ventricles of the human heart.

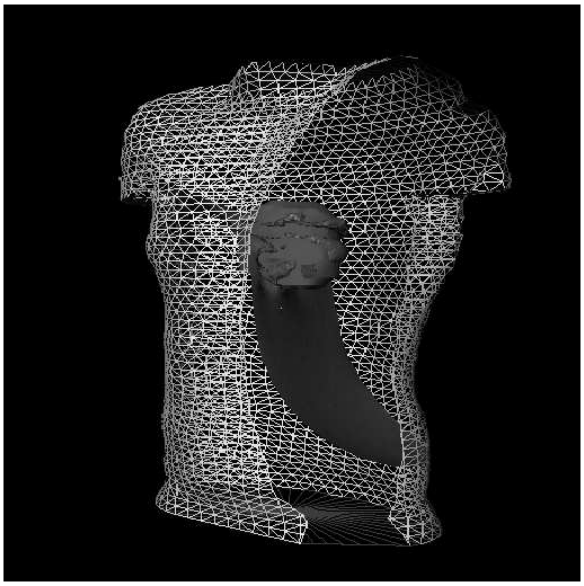

Fig. 4. Torso: An isosurface of constant voltage from a finite element simulation of the voltage distribution due to the electrical activity of the heart within a multi-chambered model of the human thorax.

B. Volume rendering

Direct volume-rendering techniques are effective tools for exploring 3D scalar data. Unlike surface-rendering methods, direct volume-rendering methods can be used to visualize 3D scalar data without converting to intermediate geometric primitives. By assigning appropriate colors and opacities to the scalar data, one can render objects semi-transparently to expand the amount of 3D information available at a fixed position. Volume-rendered images can also be superimposed upon surface-oriented icons or textures, thus allowing simultaneous scalar and vector field composite visualizations.

We were motivated to develop a more efficient way to visualize scalar fields by our attempts to visualize simulation data from a model of electrical wave propagation within the complex geometry of the heart and from large scale models of unsteady compressible fluid flow [32], [33]. Because of the regular structures used to characterize this particular set of simulation data, we can use direct volume-rendering techniques to visualize the states at each time step. We characterize the states within the model by assigning different colors and opacities. By animating the volume-rendered images at each time step, we can effectively investigate the propagation of waves through the volume.

Direct volume-rendering methods employing ray casting algorithms have become the standard methods to visualize 3D scalar data. To characterize the dynamic behavior of the flow field, one generates a series of volume rendered images at successive time steps and then records, stores, and animates the sequence. Because direct volume rendering is so time consuming to realize animations for models of any significant size (i.e. realistic problems in science and engineering), standard volume-rendering techniques prove prohibitive for animating hundreds of time steps interactively. Moreover, the disk space required for storing hundreds or thousands of sets of volumetric simulation data can be overwhelming.

We have developed an algorithm that significantly reduces the time to create volume-rendered flow animations of scalar fields. Furthermore, our algorithm reduces the amount of disk space needed for storing volume data. We achieve these reductions by implementing a differential volume-rendering method. The method utilizes data coherency between consecutive time steps of simulation data to accelerate the volume animation and to compress the volume data. The method is independent of specific volume-rendering techniques and can be adapted into a variety of ray casting paradigms which can be used to further accelerate the visualization process [34], [35], [36].

During preliminary studies of our electrical wave propagation simulations in the heart, we noticed that the only elements that changed values between consecutive time steps, when the time steps were small, were the activated cells and their neighbors. We hypothesized that only a fraction of elements in the volume change from any given time step to the next in simulations of physical flow phenomena. In addition, when a sequence of propagating images

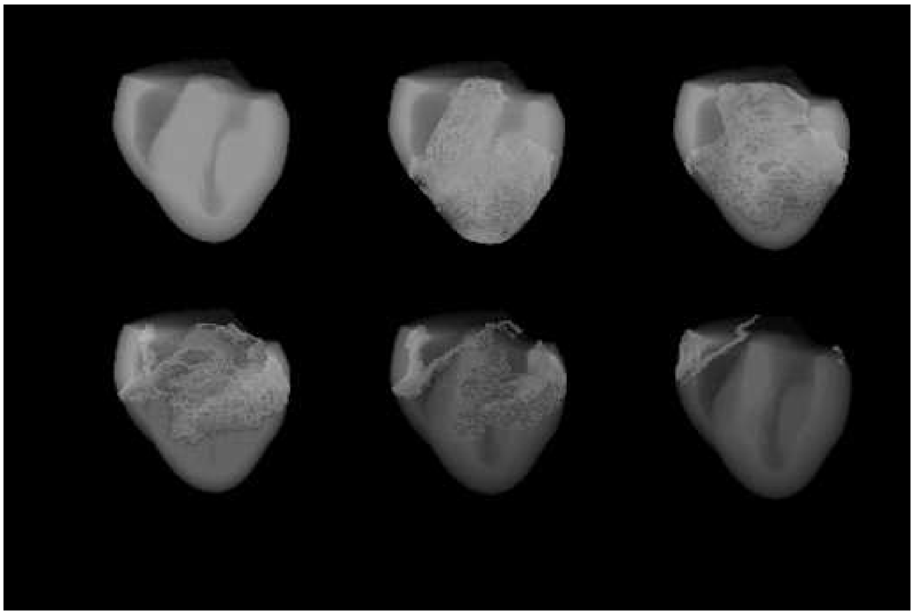

Fig. 5. Volume rendering at six different times steps in a model of electrical wave progagation within the heart

is animated, the viewing parameters usually do not change. In our ray casting method, we are able to cast rays along a path corresponding only to changed data elements. Therefore, the pixels in the new image keep the same colors that they had previously unless they correspond to changed data elements. Retaining the color values from the non-changing pixels results in significant time savings. The differential volume-rendering algorithm thus exploits the temporal coherence between sets of volume data from different time steps in order to speed up volume animation of the 3D flow.

The differential volume rendering method separates the data generation and data visualization processes. Scientists perform simulations to obtain sequences of data at different time steps. The differential volume-rendering algorithm extracts the differential information that contains the differences between the data files at each consecutive time step. According to the specified viewing direction, the pixel positions where new rays need to be cast can be computed from the differential information and then the ray casting process is invoked to produce the updated image. Because the variation between consecutive time steps is small, the differential information file, which replaces the whole sequence of volume data, can yield tremendous savings in terms of disk space (often saving more than 90% over other volume rendering methods) [37]. An example of the differential rendering method for visualizing electrical wave propagation simulations in the heart is shown in Figure 5. In addition to the differential volume rendering algorithm, we have also developed a data parallel volume rendering method. Ma et al. [38] have proposed a divide-and-conquer ray casting algorithm and a binary swap image composition method to achieve fast parallel volume rendering. Although the binary swap image composition operation is very efficient, the performance of the ray casting process is not very satisfactory. We have developed an algorithm that adopts a SIMD programming model to further parallelize the ray casting process by utilizing the vector units on CM-5. The preliminary results showed that we can accelerate the parallel ray casting process by a factor of two to three over previous results.

V. Selected Applications

In the next section we describe specific applications of computer modeling in the fields of neuroscience and cardiology. One is a diagnostic method utilizing non-invasive electrical measurements from the body or scalp surface to infer the electrical state of the heart or brain, respectively. The other simulates a therapeutic approach used in cases of severe electrical dysfunction of the heart, defibrillation. We end with an overview of a new software architecture we have developed that integrates modeling, computation, and visualization into a single framework.

A. Epileptic Seizures

Epileptic seizures are characterized by aberrant electro-cortical activity, during which the patient loses consciousness and is prone to spastic spontaneous muscular activity. This condition can, in general, be controlled by medication; however, when medication fails, the patient's only hope for recovery often lies in surgically removing the malfunctioning portion of the brain. This aberrant neural region, though totally responsible for evoking these seizures, is generally only a very small portion of the entire brain and can be removed without harm to other "normal" neural function. The problem then, is figuring out the precise location at which the aberrant activity is originating, so that this small region, known as the ectopic focus, can be extricated. If the focus is a result of a structural abnormality, a lesion or a tumor for example, it can often be identified in a magnetic resonance image (MRI) scan. Unfortunately, this is not always the case. The rest of the time, the source must be localized with more complicated techniques. One such solution is the so-called inverse-EEG method. An electrical current density map corresponding simulated temporal lobe epileptic electrical activity is shown in Figure 6, while an isosurface corresponding to a particular voltage level is shown in Figure 7.

The inverse EEG-method is a technique for correlating scalp electrical potentials with the electric potentials (1) on the surface of the brain (surface-to-surface problem), or (2) within the brain (surface-to-source problem). The scalp potentials are measured with surface electrodes, and are then (conceptually) projected back into the head via an inverse simulation to determine the cortical sources. This functional EEG data can then be correlated with anatomical MR data, to facilitate preoperative planning.

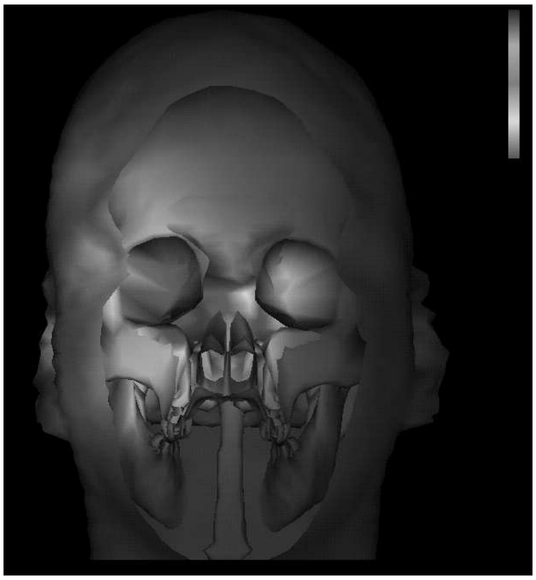

Fig. 6. Cutaway of the Utah Torso Model showing the different anatomical regions included in the model.

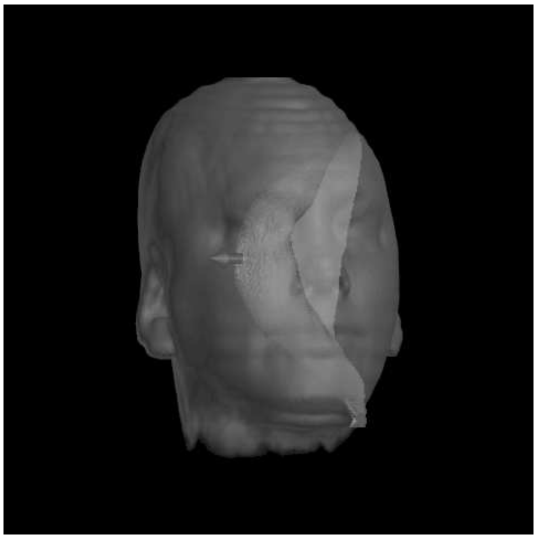

Fig. 7. Cutaway of the Utah Torso Model showing the different anatomical regions included in the model.

B. Cardiac Defibrillation

To stop the aberrant rhythms that characterize fibrillation, electrical shocks are delivered to the heart, the process of defibrillation. The design and placement of defibrillation electrodes is a critical issue for effectively restoring normal rhythm. Inefficient electrode design and placements can require unnecesarily large quantities of current that can cause significant damage to the heart. In the past, researchers would estimate effective electrode locations through several series of animal trials. Now, however, computer models have recently become sophisticated enough to assist in studying the defibrillation problem. Early computer models usually required such significant amounts of time to construct and change model parameters such that relatively few electrode configurations have been simulated [39]. We have developed a defibrillation modeling system that allows a researcher to interactively design and place electrodes into a high-resolution finite element model of the human thorax (the Utah Torso Model [10], [14]). The advantages of this system are that different electrode combinations can be tested computationally, reserving only the most promising candidates for animal studies. Reducing the number of experiments, could result in tremendous savings in device design and development costs.

B.1 Model Construction and Electrode Design

The Utah Torso Model [10], [11] was derived from MRI scans and digitized to provide the boundary points defining the major electrically relevant anatomical regions including subcutaneous fat, skeletal muscle, lungs, ribs (clavicle and sternum), epicardium, epicardial fat pads, blood cavities (atria and ventricles), and the major blood vessels (pulmonary artery and vein, aorta, superior and inferior vena cava, subclavian, innominate and azygous veins). Figure 8 depicts a cutaway of the model highlighting the different anatomical regions.

To represent subcutaneous patch electrodes within the model, we use rectangluar shpaes. Cylinders represent internal electrodes. The electrodes were deformed with an interactive modeling tool to fit inside the major vessels and/or to conform to the ribs. Once the electrodes were defined, a volumetric mesh was generated following the procedures described in II-B. Dirichlet boundary conditions were then assigned to the electrode surfaces. The model was defined by over 20,000 surface points (including the electrodes). The resulting volume mesh (we created nearly 50 unique meshes to handle over 170 different electrode combinations) ranged from 1.2 million to 1.5 million elements. The finite element method was then used to solve for the potential and current density distributions.

B.2 Defibrillation Criteria

Five different defibrillation criteria were used to determine the efficacy of various electrode configurations. These included *electrode voltage threshold (EVT)*, *efficiency (Eff)*, *damage*, *uniformity (Unif)*, and *lead impedance* Z_L. The EVT was defined as the voltage difference across the electrodes necessary to yield a gradient of either 5 V/cm, (monophasic shocks) or 3.5 V/cm, (biphasic shocks) in 95% of the heart tissue. The **efficiency** of the delivered shock was based on the ratio of the energy reaching the heart to the total energy delivered. The **damage** level was determined to be the percentage of heart tissue above a particular gradient level. The gradient levels used were 50 V/cm for monophasic shocks and 70 V/cm for biphasic shocks. The **uniformity** of the delivered shock was based on the ratio of the peak electric field in the heart volume to the average electric field in the heart volume. It ranges from 1 for a perfectly uniform field to ∞ for a very non-uniform

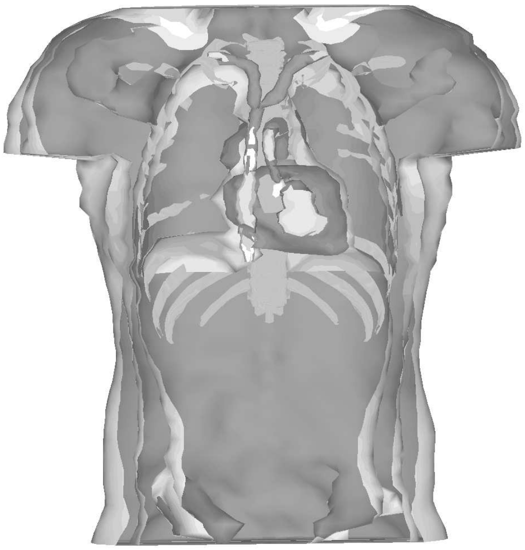

Fig. 8. Cutaway of the Utah Torso Model showing the different anatomical regions included in the model.

electric field. The **lead impedance** is the impedance across the electrodes and is commonly determined in actual experimental and clinical settings. This serves as an additional check of the model against experimental data.

B.3 Defibrillation Results

Over 170 different electrode combinations were implemented and computationally tested with this interactive modeling tool. Preliminary results showed that several of the defibrillation criteria parameters were within the ranges measured in clinical and experimental settings. Shown in Figure 9 is the current density distribution on the heart surface for a particular defibrillation electrode configuration. The darker regions indicate high current density magnitudes. Table I compares two different electrode configurations that we explored. The first case refers to two cylindrical leads positioned in a vessel and blood cavity. The second case refers to the same lead system with an additional anodal subcutaneous patch. Case 2 with a subcutaneous patch has a significantly lower threshold voltage compared to case 1. Based on the criteria that smaller electrode voltage thresholds are better, case 2 could merit further testing.

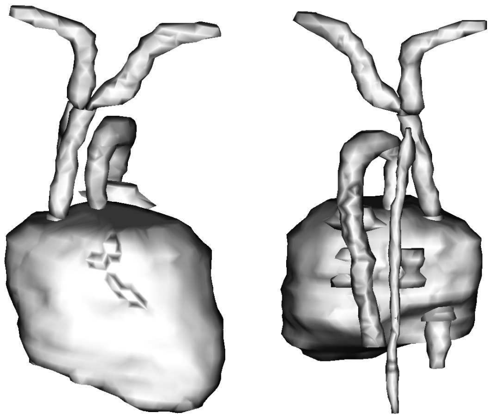

Fig. 9. Figure showing the current density magnitude distributions for both the anterior and posterior views of the mediastinum for a typical defibrillation configuration.

	EVT		Eff	Damage		Unif	Z_L
	Mono	Bi		Mono	Bi		
1	801.8	561.3	18.7	14.3	3.0	14.0	48.5
2	389.7	272.8	17.2	3.5	0.02	7.8	34.6

TABLE I

A sample comparison of the defibrillation criteria results for two different cases. Case 1 is for an anode and cathode lead system. Case 2 uses the same lead system as case 1, but it also includes an anodal subcutaneous patch electrode.

C. *The Inverse ECG/EEG Problem*

In this next section, we examine the inverse bioelectric fields problems associated with the heart and brain. Mathematically, these problems have identical formulations, so we'll focus primarily on the inverse-ECG problem here, noting the corresponding EEG problem in parentheses where appropriate. The intrinsic bioelectric source currents in the heart gives rise to electric, and thus electric potential, fields within the volume of the thorax (head) and upon the torso (scalp) surface. The voltages on the torso (scalp) surface are related to voltages upon the heart (cortical) surface via the resistive properties of the intermediary tissues of the thoracic (cranial) cavity or volume conductor. The general inverse problem in electrocardiography (electroencephalography) can be stated as follows: given a subset of electrostatic potentials measured on the surface of the torso (scalp) and the geometry and conductivity properties of the thorax (head), calculate the potential fields on, and source currents within, the heart (brain). Mathematically this can be posed as an inverse source problem in terms of the primary current sources within the heart (brain) and described mathematically by Poisson's equation for electrical conduction:

$$\nabla \cdot \sigma \nabla \Phi = -I_v \quad \text{in } \Omega \tag{12}$$

with the boundary condition:

$$\sigma \nabla \Phi \cdot \mathbf{n} = 0 \quad \text{on } \Gamma_T \tag{13}$$

where Φ are the electrostatic potentials, σ is the conductivity tensor, I_v are the cardiac (cortical) current sources per unit volume, and Γ_T and Ω represent the surface and the volume of the thorax (head), respectively. The goal of the inverse solution is to recover the magnitude and location of the cardiac (cortical) sources. Alternatively, it is possible to formulate the problem in terms of the electrostatic potentials on a surface bounding the heart (brain). This form of the inverse problem does have a unique solution. Thus, instead of solving Poisson's equation, we solve a generalized Laplace's equation with Cauchy boundary conditions:

$$\nabla \cdot \sigma \nabla \Phi = 0 \tag{14}$$

with boundary conditions

$$\Phi = \Phi_0 \quad \text{on } \Sigma \subseteq \Gamma_T \quad \text{and} \quad \sigma \nabla \Phi \cdot \mathbf{n} = 0 \quad \text{on } \Gamma_T. \tag{15}$$

While this version of the inverse problem has a unique solution [40], the problem is still mathematically ill-posed in the Hadamard sense; i.e., because the solution does not depend continuously on the data, small errors in the measurement of the voltages on the torso (scalp) can yield unbounded errors in the solution.

An accurate solution to the inverse problem in electrocardiography would provide a non-invasive procedure for the evaluation for myocardial ischemia [41], the localization of ventricular arrhythmias and the site of accessory pathways in Wolff-Parkinson-White (WPW) syndrome, and, more generally, the determination of patterns of excitation and recovery of excitability.

Similarly, an accurate solution to the inverse problem in electroencephalography would provide a non-invasive procedure for functionally mapping the cortex. Traditional efforts of non-invasive cortical mapping have used simplified geometries such as concentric spheres to approximate the complex physical domains of the scalp, skull, cerebrospinal fluid, grey matter, and white matter. While these methods yield a good first approximation to the true cortical potentials for simple cases involving a single dipole source, they are not accurate enough for many clinical applications [42]. An accurate inverse solution would also provide a non-invasive means for localizing and analyzing the activity of ectopic foci in epileptic patients [43], thereby providing neurologists reliable data on which to diagnose and treat patients.

C.1 Regularization of the ECG Inverse Problem

Traditional schemes for solving the inverse problem, (14), have involved reformulating the linear equation, $A\xi = b$, into $\xi_T = K\xi_H$, where ξ_T and ξ_H are the voltages on the torso (scalp) and heart (cerebral cortex) respectively, and K is the $T \times H$ transfer matrix of coefficients relating the measured torso voltages to the voltages measured on the heart. The inverse problem can then be stated as, $\xi_H = K^{-1}\xi_T$.

Unfortunately, K is ill-conditioned, and regularization techniques are necessary to restore continuity of the solution back onto the data. Regularization schemes work by finding an approximate vector, $\xi_{C\varepsilon}$, such that $\xi_{C\varepsilon} \rightarrow \xi_H$ as $\varepsilon \rightarrow 0$, where ε is the error between the approximate and true values. As introduced in a previous work [17], we found that it was possible to reformulate the K matrix into an expression involving sub-matrices of K such that regularization can then be applied at a *local* level.

Researchers have long noticed that the discontinuities in the inverse solution appear irregularly distributed throughout the data. Tikhonov and other regularization schemes are, in effect, filters, which restore continuity by attenuating the high (spatial) frequency components of the solution. Since regularization is usually applied globally, the results can leave some regions overly damped or smoothed while others remain poorly constrained. Our idea was then, to apply regularization only to sub-matrices that require it and to apply different amounts of regularization to different sub-matrices.

Briefly, we begin by expressing $A\xi = b$ for the Cauchy problem in (14) as

$$\begin{pmatrix} A_{TT} & A_{TV} & A_{TH} \\ A_{VT} & A_{VV} & A_{VH} \\ A_{HT} & A_{HV} & A_{HH} \end{pmatrix} \begin{pmatrix} \xi_T \\ \xi_V \\ \xi_H \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \tag{16}$$

We can then rearrange the matrix to solve directly for the cortical voltages in terms of the measured scalp voltages,

$$\xi_H = (A_{TV}A_{VV}^{-1}A_{VH} - A_{TH})^{-1}(A_{TT} - A_{TV}A_{VV}^{-1}A_{VT})\xi_T, \tag{17}$$

where the subscripts, S, V, and C stand for the nodes in the regions of the scalp, volume, and cortex, respectively.

Setting A_T equal to $A_{TT} - A_{TV}A_{VV}^{-1}A_{VT}$ and A_H equal to $A_{TV}A_{VV}^{-1}A_{VH} - A_{TH}$, yields

$$\xi_H = A_H^{-1}A_T\xi_T. \tag{18}$$

Note that we now have (possibly pseudo) inverses of two matrices, A_{VV} and A_H, as well as several other matrix operations to perform. If one estimates the condition number, κ, of the two matrices using the ratio of the maximum to minimum singular values from a singular value decomposition (SVD), one finds that the condition number varies significantly (from $\kappa \approx 200$ for A_{VV} to $\kappa \approx 1.0 \times 10^{16}$ for A_{VH} in the three-dimensional finite element model described below). Thus, one can regularize matrices differently, leaving some of the sub-matrices untouched from filtering. The overall effect of this *local regularization* is more control over the regularization process.

As is generally the case, there are trade-offs in obtaining more accurate solutions. Here, the principle of *conservation of complexity* applies. Achieving greater accuracy in the inverse solution using the local regularization method increases computational costs. In the global method, the regularization procedure and the estimation of the regularization parameter, α, is accomplished using the $T \times H$ transfer matrix, K, while the local scheme must "invert" two (or more, depending on the partition) sub-matrices which are $V \times V$ and $T \times H$ in size. It is almost always the case that $\dim\{V\} >> \dim\{T, H\}$. Thus, one needs to make compromises concerning accuracy versus CPU time.

C.2 Inverse Problem Results

As a test of the adaptive algorithm discussed in section III-A, a simulated source distribution was placed on the surface of the heart model. Direct solutions were computed for different levels of mesh refinement and compared at the outer torso boundary. Figure 3 shows the voltage at the outer boundary versus distance around a transthoracic contour. The maximum estimated error in the calculated potential was over 30% greater in the original mesh compared to the final mesh, and the maximum estimated error in the potential gradient was over 13% larger in the original mesh compared to the refined mesh. Increased accuracy does not come without a price: h-refinement increases the number of degrees of freedom, and thus, the computational costs. However, since the relative error falls off rapidly with increasing degrees of freedom and then levels out after only a few iterations of the adaptive algorithm, it is not difficult to choose a reasonable balance between accuracy and computational cost.

The application of the local regularization technique recovered the voltages to within 12.6% RMS error. Previous studies in three-dimensional problems have reported the recovery of epicardial potentials with errors in the range of 20-40%, [44], [45], [46], [47]. Figure 4 shows the inverse solution calculated using the local regularization technique compared with the recorded heart voltages as a function of position on the epicardium. The global solution tended to be smoother, not able to follow the extrema as well as the local solution could. The local solution also showed areas of local error, which suggest that a different partitioning

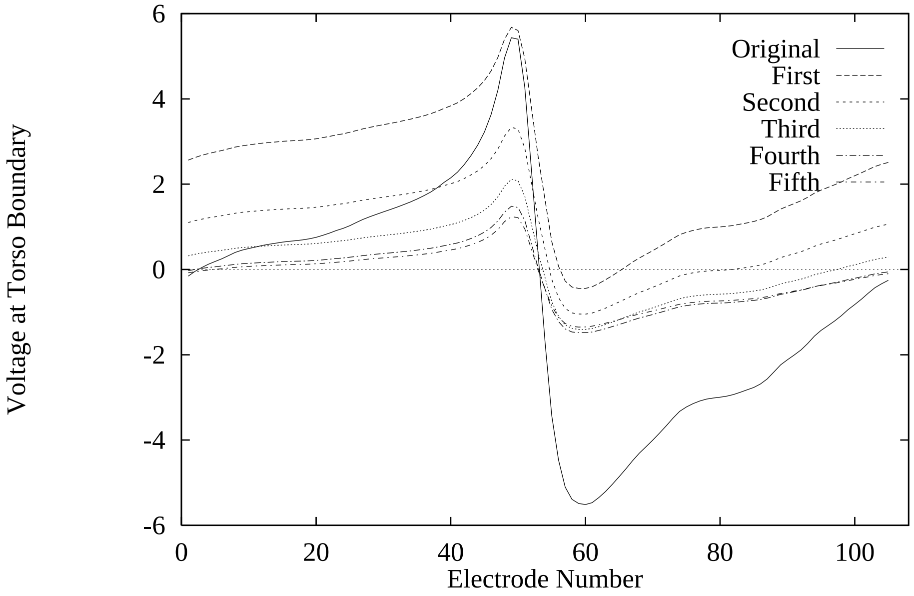

Fig. 10. Effects of Automatic Mesh Refinement

of the sub-matrices may provide even better accuracy. Currently, we are investigating the effects of added noise on a variety of experimental data as well as developing more general and computationally efficient ways to decompose the transfer matrix.

VI. Computational Steering

The current computational engineering and science modeling process is a sequential one – create, or make modifications to, a geometric model, input initial conditions and/or boundary conditions, numerically approximate solutions to the governing equation(s), store data off to disk and/or tape, visualize the results via a separate visualization package, decide upon appropriate changes, and repeat the whole process *ad nauseam*. Usually, the modifications made to the model, input parameters, numerical and computing methods, and visualization parameters are accomplished using separate in-line, command based instructions, and each change made in the model and/or input parameters requires all of the other steps to be repeated. Even for experienced scientists who may employ scripts and conversion programs to facilitate the task, the process is seldom streamlined. Ideally, scientists and engineers would be provided with a system in which all these computational components were linked, so that all aspects of the modeling and simulation process could be controlled interactively through a graphical interface within the context of a single application program. While this would be the preferred *modus operandi* for most computational scientists, it is not the current standard because it is difficult and time consuming to create such a program and

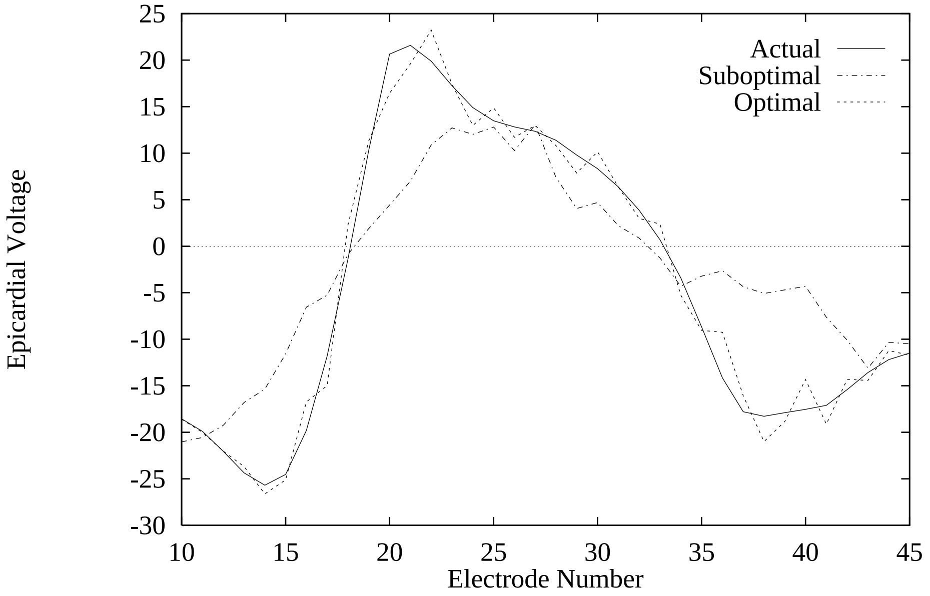

Fig. 11. Local Regularization Technique

tailor it to the particular requirements of a specific research problem.

A. SCIRun

To reduce the time the scientist spends in using this standard modeling paradigm and to provide a tool for exploration of computational science and engineering problems, we have developed SCIRun[2] [48], [49]. SCIRun is a framework in which large scale computer simulations can be composed, executed, controlled and tuned interactively. Composing the simulation is accomplished via a visual programming interface to a dataflow network. To execute the program, one specifies parameters with a graphical user interface rather than with the traditional text-based datafile. Controlling a simulation involves steering the simulation interactively as it progresses.

Computational steering has been defined as "the capacity to control the execution of long-running, resource-intensive programs" [50]. In the field of computational science, we apply this concept to link visualization with geometric design and computational phases to interactively explore (steer) a simulation in time and/or space. As knowledge is gained, it can be used to change the input conditions and/or other parameters of the simulation. Implementation of a computational steering framework requires a successful integration of the many aspects of scientific computing, including geometric modeling, numerical analysis,

[2] Pronounced "ski-run." SCIRun derives its name from the Scientific Computing and Imaging (SCI) research group at the University of Utah.

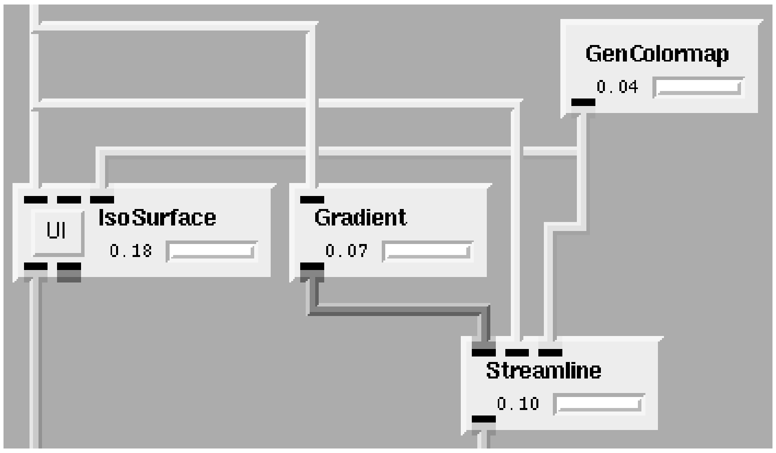

Fig. 12. Dataflow Network for a Subset of Visualization Modules

and scientific visualization, all of which need to be effectively coordinated within an efficient computing environment.

SCIRun is a framework in which large scale computer simulations can be composed, executed, controlled and tuned interactively. Composing the simulation is accomplished via a visual programming interface to a dataflow network. Software systems such as AVS from Application Visualization Systems Inc. [51], Iris Explorer from Silicon Graphics, and Visualization Data Explorer from IBM [52] have made this archetype popular for scientific visualization [53]. Our work has extended this paradigm into the realm of scientific computation. In SCIRun, the typical components of the computational paradigm – geometric modeling, numerical analysis, and scientific visualization – are integrated into a visual programming environment with the ability to interactively steer any one phase of the process and to see the effects propagate throughout the system automatically.

As a simple example we consider a group of visualization modules within a dataflow network, illustrated in Figure 5. The boxes represent computational algorithms (modules) and the lines represent data pipes between the modules. Scalar field data enters through the pipe at the top, passing into an isosurface extraction algorithm and a gradient field computation. Streamlines are computed on the resulting gradient vector field and colored according to the scalar values. Geometry objects for the isosurfaces and streamlines are passed out the bottom.

When the user changes a parameter in any of the module user interfaces, the module is re-executed, and all changes are automatically propagated to all connecting modules. The user is freed from worrying about details of data dependencies and data file formats. The user may make some changes without stopping the computation, thus "steering" the computational process. Other changes cause the computations to be cancelled and automatically re-started. The combination of manual and automatic tasks improves the efficiency of this "computational workbench."

B. Remote Steering of Large Datasets

One of the major disadvantages of most steering systems is their limited ability to manipulate the extremely large data-sets generated on large parallel supercomputing systems. Such systems are a valuable computing resource–allowing researchers to solve problems that are too large or complex to be solved on ordinary workstations. Unfortunately, these systems are often located remote sites such as National Laboratories or Supercomputing centers. This creates a fundamental problem—very large simulations tend to generate vast amounts of data (tens to hundreds of gigabytes) that simply can not be transferred across the network to a local workstation for analysis and visualization. Even using the fastest graphics workstations available can not help this situation since the real bottleneck becomes the limited network bandwidth.

Recently, we have been exploring the possibility of combining large-scale computational codes with parallel visualization and analysis software. With this approach, we generate images remotely and send them across the network instead of the raw simulation data. This changes the problem to one of finding good image compression techniques and tactics for providing an interactive user-environment that can operate on little or no simulation data. Unlike systems that rely on high performance workstations, this approach allows researchers to explore large datasets from inexpensive workstations connected via standard internet connections to supercomputing centers.

At this point, our work in this area is preliminary, but, in collaboration with Los Alamos National Laboratory, we have developed a prototype system for remote visualization and analysis of large-scale molecular dynamics simulations. With this system, it is possible to visualize and interactively manipulate datasets involving more than 100 million particles from an ordinary UNIX workstation located at a remote site. In recent tests, we have been able to interactively manipulate large datasets on the LANL CM-5 and T3D from workstations at the University of Utah. While we are only in the early stages of this work, we feel that such an approach can be applied to many problems in computational medicine. Considering that many users of a computational medicine system may not have large supercomputing systems on-site, a remote visualization and analysis system will allow these users to solve certain large-scale computational problems on a supercomputing center located elsewhere.

VII. Future Directions

One of the goals of the HPCC is to "Extend U.S. technological leadership in high performance computing and computer communications [54]". Unfortunately, it not economically feasible to put a high performance computer on the desk of every scientist in the country. Since most scientists have ready access to a (relatively) inexpensive workstation and a network, developing interactive tools designed for standard workstations would be of great benefit to a large number of researchers. We believe that the remote steering system software (or similar) that we previously described is one such approach for extending the capabilities of the HPCC computing equipment to the scientists' desks.

A major impetus of HPCC goals is to use high performance computing to solve important problems in science and engineering. A clear bottleneck to accomplishing this goal is difficulty of both using advanced architectures and managing the tremendous amount of data generated by large-scale simulations. Many of the unsolved problems in high performance computing are related to the user interface - we believe that increased interactivity, improved debugging and increased control, are essential ingredients of future high performance software. To this end, the computational steering software system we are currently developing may help application scientists better utilize high performance architectures by providing a greater degree of interaction and control than is presently possible on current supercomputing systems. Ultimately, we hope to develop a flexible steering system that allows scientists to interact with large-scale simulations by linking desktop workstations and remote supercomputing facilities.

While past computational efforts have focused on using vector supercomputers to exploit fine-grained loop-level parallelism, shared memory multiprocessors and/or systems of clustered work stations for course-grained parallelism, we believe that the future of large-scale scientific computing will lend itself ultimately to mixed collections of shared memory machines, massively parallel machines, workstation clusters, and graphics engines connected by high bandwidth networks – a *network multicomputer*. To this end we are developing algorithms and software that will be able to exploit such heterogeneous systems. In particular, we are investigating large-scale, unstructured, three-dimensional mesh generation algorithms, and are working on linking such algorithms with adaptive finite element programs, sparse system solvers, and visualization software via our computational steering software. We are also investigating more general and efficient ways to decompose sparse ill-conditioned linear systems using a combination of iterative Krylov subspace methods coupled with local regularization techniques, with the aim of controling accuracy and balancing computational costs.

While advances in computer architecture has allowed scientists to apply the computational paradigm to biophysical systems, the major bottleneck to succesfully utilizing high performance computing in medical and healthcare applications, is the lack of existing usuable (i.e. that non-hpc specialists can use) software. Such application software is essential for the computational methodology to make substantial contributions to physiologically and clinically important areas.

VIII. Acknowledgements

This work was supported in part by awards from the NIH and the NSF. The authors would like to thank K. Coles and J. D. Brederson for their helpful comments and suggestions. Furthermore, we appreciate access to facilities that are part of the NSF STC for Computer Graphics and Scientific Visualization.

IX. Author Biographies

Christopher Johnson received his Ph.D. from the University of Utah in 1989. He is currently a member of the faculty in the Department of Computer Science at the University of Utah. His research interests are in the area of scientific computing. Particular interests include inverse and imaging problems, adaptive methods for partial differential equations, numerical analysis, large scale computational problems in medicine, and scientific visualization. Professor Johnson was awarded a FIRST Award from the NIH in 1992, a National Young Investigator (NYI) Award from the NSF in 1994, and the Presidential Faculty Fellow (PFF) Award in 1995. He heads the Scientific Computing and Imaging (SCI) research group at the University of Utah.

David Beazley is a Ph.D. student in the Department of Computer Science at the University of Utah. For the past 5 years, he worked in the Theoretical Division at Los Alamos National Laboratory on materials modeling, molecular dynamics, and large-scale massively parallel processing. He is currently interested in the problem of analyzing and managing extremely large scientific datasets using inexpensive workstations and networks.

Yarden Livnat received a B.Sc. in Computer Science in 1982 from Ben Gurion University in the Negev, Israel and an M.Sc. in Computer Science from the Hebrew University, Israel in 1991. He is currently a Ph.D. candidate at the University of Utah working with the Scientific Computing and Imaging research group. His research interests include computational geometry, geometric modeling, scientific visualization and computer generated holograms.

Steve Parker is a Ph.D. candidate in the Department of Computer Science at the University of Utah. His research interests include computational steering, scientific computing, visualization, parallel programming, and graphics. In 1994 he was awarded a Computational Science Graduate Fellowship by the Department of Energy.

John Schmidt received his B.S.E degree in Biomedical and Electrical Engineering and his M.S. and Ph.D. from Duke University in 1986, 1989, and 1993, respectively. He is currently a Research Computing Scientist in the Department of Computer Science at the University of Utah. His research interests include mesh generation, adaptive methods, computational electrophysiology, and scientific visualization.

Han-Wei Shen is currently a Ph.D. candidate at the University of Utah working with the Scientific Computing and Imaging Research Group. He received his B.S. in 1988 from the National Taiwan University in Taipei, Taiwan, and his M.S. in 1992 from the State University of New York at Stony Brook. His research interests include scientific visualization, computer graphics, and parallel rendering.

David Weinstein received B.A. degrees in Computer Science and Applied Mathematics from the University of California, Berkeley in 1992. An NSF Graduate Research Fellow, he is currently a Ph.D. candidate at the University of Utah working with the Scientific Computing and Imaging Research Group. His research interests include inverse bioelectric field problems, medical imaging techniques, computer graphics and scientific visualization.

REFERENCES

[1] C.E. Miller and C.S. Henriquez. Finite element analysis of bioelectric phenomena. *Crit. Rev. in Biomed. Eng.*, 18:181–205, 1990.

[2] T.C. Pilkington, B. Loftis, J.F. Thompson, S.L-Y. Woo, T.C. Palmer, and T.F. Budinger. *High-Performance Computing in Biomedical Research*. CRC Press, Boca Raton, 1993.

[3] H.W. Shen and C.R. Johnson. Semi-automatic image segmentation: A bimodel thresholding approach. Technical report, UUCS-94-019, Dept. of Computer Science, Univ. of Utah, 1994.

[4] D. Weinstein and C.R. Johnson. Hierarchical data structures for interactive volume visualization. *IEEE Visualization '95*, 1995 (submitted).

[5] J.A. Schmidt, C.R. Johnson, J.C. Eason, and R.S. MacLeod. Applications of automatic mesh generation and adaptive methods in computational medicine. In I. Babuska, J.E. Flaherty, W.D. Henshaw, J.E. Hopcroft, J.E. Oliger, and T. Tezduyar, editors, *Modeling, Mesh Generation, and Adaptive Methods for Partial Differential Equations*, pages 367–390. Springer-Verlag, 1995.

[6] D.F. Watson. Computing the n-dimensional Delaunay tesselation with applications to Voronoi polytopes. *Computer Journal*, 24(2):167–172, 1981.

[7] J. Thompson and N.P. Weatherill. Structed and unstructed grid generation. In T.C. Pilkington, B. Loftis, J.F. Thompson, S.L-Y. Woo, T.C. Palmer, and T.F. Budinger, editors, *High-Performance Computing in Biomedical Research*, pages 63–112. CRC Press, Boca Raton, 1993.

[8] W. Wells, R. Kikinis, W. Grimson, and F. Jolesz. Statistical intensity correlation and segmentation of magnetic resonance image data. In *Proceedings of the Third Conference on Visualization in Biomedical Computing*. SPIE, 1994.

[9] H.W. Shen and C.R. Johnson. Sweeping simplices: A fast isosurface extraction algorithm for unstructured grids. In *Visualization '95*. IEEE Press, 1995 (to appear).

[10] C.R. Johnson, R.S. MacLeod, and P.R. Ershler. A computer model for the study of electrical current flow in the human thorax. *Computers in Biology and Medicine*, 22(3):305–323, 1992.

[11] C.R. Johnson, R.S. MacLeod, and M.A. Matheson. Computational medicine: Bioelectric field problems. *IEEE COMPUTER*, pages 59–67, Oct., 1993.

[12] R.S. MacLeod, C.R. Johnson, and M.A. Matheson. Visualization tools for computational electrocardiography. In *Visualization in Biomedical Computing*, pages 433–444, 1992.

[13] R.S. MacLeod, C.R. Johnson, and M.A. Matheson. Visualization of cardiac bioelectricity — a case study. In *IEEE Visualization '92*, pages 411–418, 1992.

[14] C.R. Johnson, R.S. MacLeod, and M.A. Matheson. Computer simulations reveal complexity of electrical activity in the human thorax. *Comp. in Physics*, 6(3):230–237, May/June 1992.

[15] C.R. Johnson and R.S. MacLeod. Nonuniform spatial mesh adaption using a posteriori error estimates: applications to forward and inverse problems. *Applied Numerical Mathematics*, 14:311–326, 1994.

[16] J.A. Schmidt, C.R. Johnson, J.C. Eason, and Macleod R.S. *Modeling, Mesh Generation, and Adaptive Methods for Partial Differential Equations*, chapter Applications of Automatic Mesh Generation and Adaptive Methods in Computational Medicine. Springer-Verlag, 1994.

[17] C.R. Johnson and R.S. MacLeod. Inverse solutions for electric and potential field imaging. In R.L. Barbour and M.J. Carvlin, editors, *Physiological Imaging, Spectroscopy, and Early Detection Diagnostic Methods*, volume 1887, pages 130–139. SPIE, 1993.

[18] P.G. Ciarlet and J.L Lions. *Handbook of Numerical Analysis: Finite Element Methods*, volume 1. North-Holland, Amsterdam, 1991.

[19] D. Burnett. *Finite Element Analysis- From Concepts to Applications.* Addison-Wesley, Reading, MA, 1987.

[20] R. Lewis, H. Huang, A. Usmani, and J. Cross. Finite element analysis of heat transfer and flow problems using adaptive remeshing including application to solidification problems. *International Journal of Numerical Methods in Engineering*, 32:767–781, 1991.

[21] O. Zienkiewicz and J. Zhu. A simple error estimator and adaptive procedure for practical engineering analysis. *International Journal of Numerical Methods in Engineering*, 24:337–357, 1987.

[22] O. Zienkiewicz and J. Zhu. Adaptivity and mesh generation. *International Journal of Numerical Methods in Engineering*, 32:783–810, 1991.

[23] R.S. MacLeod, C.R. Johnson, and M.A. Matheson. Visualizing bioelectric fields. *IEEE Computer Graphics and Applications*, pages 10–12, 1993.

[24] H.W. Shen and C.R. Johnson. Differential volume rendering: A fast alogrithm for scalar field animation. In *Visualization '94*, pages 180–187. IEEE Press, 1994.

[25] H. Shen and C. R. Johnson. Sweeping simplicies: A fast iso-surface extraction algorithm for unstructured grids. *Proceedings of Visualization '95*, 1995. (to appear).

[26] Y Livnat, H. Shen, and C. R. Johnson. A near optimal isosurface extraction algorithm for structured and unstructured grids. *IEEE Trans. Vis. Comp. Graphics*, 1995. (submited).

[27] W.E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics*, 21(4):163–169, July 1987.

[28] J. Wilhelms and A. Van Gelder. Octrees for faster isosurface generation. *ACM Transactions on Graphics*, 11(3):201–227, July 1992.

[29] T. Itoh and K. Koyyamada. Isosurface generation by using extrema graphs. In *Proceedings of Visualization '94*, pages 77–83. IEEE Computer Society Press, Los Alamitos, CA, 1994.

[30] M. Giles and R. Haimes. Advanced interactive visualization for CFD. *Computing Systems in Engineering*, 1(1):51–62, 1990.

[31] R. S. Gallagher. Span filter: An optimization scheme for volume visualization of large finite element models. In *Proceedings of Visualization '91*, pages 68–75. IEEE Computer Society Press, Los Alamitos, CA, 1991.

[32] P. Ghapure and C.R. Johnson. A 3d cellular automata model of the heart. In *Proceedings of 15th Annual IEEE Engineering in Medicine and Biology Society International Conference*. IEEE Press, 1993.

[33] K.-L. Ma and K. Sikorski. A distributed algorithm for the three-dimensional compressible navier-stokes equations. *Transputer Research and Applications*, 4, 1990.

[34] R Yagel and Z. Shi. Accelerating volume animation by space-leaping. In *Proceedings of Visualization '91*, pages 62–69. IEEE Computer Society Press, Los Alamitos, CA, October 1993.

[35] R Yagel and A. Kaufman. Template-based volume viewing. In *Proceedings of EUROGRAPHICS '92*, pages 153–157. Blackwell, Cambridge, England, September 1992.

[36] M.F. Cohen K.-L. Ma and J.S. Painter. Volume seeds: A volume exploration technique. *The Journal of Visualization and Computer Animation*, 2:135–140, 1991.

[37] H.W. Shen and C.R. Johnson. Differential volume rendering: A fast volume visualization technique for flow animation. In *IEEE Visualization '94*, 1994 (to appear).

[38] K.-L. Ma J.S. Painter, C.D. Hansen and M.F. Krogh. A data distributed, parallel algorithm for ray-traced volume rendering. In *Parallel Rendering Symposium*, pages 15–22. IEEE Computer Society Press, Los Alamitos, CA, 1993.

[39] J. Schmidt. *Mesh generation with applications in computational electrophysiology.* PhD thesis, Duke University, Durham, NC, 1993.

[40] Y. Yamashita. Theoretical studies on the inverse problem in electrocardiography and the uniqueness of the solution. *IEEE Trans Biomed. Eng.*, BME-29:719–725, 1982.

[41] R.S. MacLeod, C.R. Johnson, M.J. Gardner, and B.M. Horáček. Localization of ischemia during coronary angioplasty using body surface potential mapping and an electrocardiographic inverse solution. In *IEEE Computers in Cardiology*, pages 251–254. IEEE Press, 1992.

[42] Y. Yan, P.L. Nunez, and R.T. Hart. Finite-element model of the human head: scalp potentials due to dipole sources. *Medical & Biological Engineering & Computing*, 29:475–481, 1991.

[43] F.H. Lopes da Silva. A critical review of clinical applications of topographic mapping of brain potentials. *Journal of Clinical Neurophysiology*, 7(4):535–551, 1990.

[44] F.P. Colli Franzone, G. Gassaniga, L. Guerri, B. Taccardi, and C. Viganotti. Accuracy evaluation in direct and inverse electrocardiology. In P.W. Macfarlane, editor, *Progress in Electrocardiography*, pages 83–87. Pitman Medical, 1979.

[45] P. Colli Franzone, L. Guerri, S. Tentonia, C. Viganotti, S. Spaggiari, and B. Taccardi. A numerical procedure for solving the inverse problem of electrocardiography. Analysis of the time-space accuracy from *in vitro* experimental data. *Math. Biosci.*, 77:353, 1985.

[46] B.J. Messinger-Rapport and Y. Rudy. Regularization of the inverse problem in electrocardiography: A model study. *Math. Biosci.*, 89:79–118, 1988.

[47] P.C. Stanley, T.C. Pilkington, and M.N. Morrow. The effects of thoracic inhomogeneities on the relationship between epicardial and torso potentials. *IEEE Trans Biomed. Eng.*, BME-33:273–284, 1986.

[48] C.R. Johnson and S.G. Parker. A computational steering model for problems in medicine. In *Supercomputing '94*, pages 540–549. IEEE Press, 1994.

[49] S.G. Parker and C.R. Johnson. Scirun: A scientific programming environment for computational steering. In *Supercomputing '95*. IEEE Press, 1995. (submitted).

[50] W. Gu, J. Vetter, and K. Schwan. An annotated bibliography of interactive program steering. *Georgia Institute of Technology Technical Report*, 1994.

[51] C. Upson and et al. The application visualization system: A computational environment for scientific visualization. *IEEE Computer Graphics & Applications*, 9(4):30–42, July 1989.

[52] B. Lucas and et al. An architecture for a scientific visualization system. In *Proceedings of Visualization '92*, pages 107–114. IEEE Press, 1992.

[53] C. Willams, J. Rasure, and C. Hansen. The state of the art of visual languages for visualization. In *Proceedings of Visualization '92*, pages 202–209. IEEE Press, 1992.

[54] Communications The High Performance Computing, Information Technology (HPCCIT) Subcommittee of the National Science, and Technology Council. High performance computing and communications: Foundations for america's information future, 1995.