17

Image-Based Modeling for Bioengineering Problems

Adrienne M. Madison and Mark A. Haidekker

CONTENTS

ABSTRACT Computational modeling uses a numerical approach to simulate material behavior under defined spatial constraints and load conditions. For this purpose, a system is decomposed into a large number of interacting volume elements (called finite elements), which can be described by a set of partial differential equations. Numerical methods are then employed to solve the equation system for the unknown quantities, such as deformation and stress. Finite-element models have been employed to observe the mechanical response behavior of biological tissue as well as implant materials. Unlike in mechanical systems, where a relatively simple geometrical description is available, patient-specific models of organs or implants are often obtained from volumetric images. Image-based finite-element models have a highly complex geometry, and the assignment of material properties to individual elements from image information is difficult. In this chapter, the four steps to obtain an image-based finite element–based material simulation (segmentation, meshing, simulation, and postprocessing) are described in detail, and strategies to overcome the specific challenges of image-based computational modeling are discussed. A focus of this chapter lies on available software to perform the four steps with aspects of the practical realization of a finite-element modeling chain.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.1 Overview

Finite-element analysis (FEA, or finite-element modeling [FEM]) is a numerical modeling technique used to solve engineering problems by obtaining approximate solutions from a large system of algebraic equations. The underlying principle requires the entire geometry of interest to be subdivided into a large number of small regions or *elements* that are considered homogeneous. These small elements have a variety of possible shape patterns (e.g., triangular, rectangular, tetrahedral, hexagonal, and cubic), depending on the number of geometric dimensions present, and they share their edges and vertices with neighboring elements. The vertices (referred to as *nodes*) are rigidly connected; therefore, each element has boundary conditions imposed upon it by its neighbors (internal boundary conditions) and by the environment (external boundary conditions). Internal boundary conditions are obtained from the balance of forces and from the equal displacement of a node that is common to multiple elements. External boundary conditions, such as pressure, forces, or spatial constraints, can be prescribed to the model as a whole. The actual finite-element *model* is represented by a series of partial differential equations that describe material properties, conservation of mass and energy, the laws of motion, and deformation of an element by forces acting upon its nodes. Ultimately, solutions are found for each region by the calculation of unknowns within the volume by only taking into account the regions located next to them.

Originally developed to analyze the twisting behavior of cylindrical objects in the early 1940s [1], FEM has been intensely used in biomedical contexts to examine, simulate, and predict the material behavior and nonlinear biomechanical properties of soft tissues [2,3], organs [4,5], bone [6,7], and joints [8,9]. It is also highly useful in applications of modeling, testing, and verification of medical device designs, such as vascular implants and stents [10–12], dental implants [13], and most recently in accident analysis and prevention [14]. The number of reported studies that utilize FEM and related computational numerical methods in the advancement of biomedical and clinical research, development, diagnosis,

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

and treatment applications has constantly increased since 1980 and currently surpasses 10,000 [15].

Classical finite-element models are based on geometric descriptions of the object under examination: A motor, for example, can be composed of cylinders, pistons, rods, the crank shaft, and other interacting parts that can be represented by their exact geometrical description. In biomedical applications, two challenges appear. First, a tissue or organ can have a fairly complex geometry that necessitates an equally complex analytical description, sometimes with approximations made to simplify the model. Second, to obtain a patient-specific description, the geometry is often extracted from volumetric images, such as computed tomography (CT) or magnetic resonance (MR) images. FEM of biomedical structures involves four main steps (Figure 17.1) and begins with a volumetric image. In the first step, the object of interest is *segmented*, that is, separated from image elements not related to the object of interest. At this point, the object is still represented by individual voxels. The *meshing* step follows, in which the volumetric arrangement of voxels is parametrized (i.e., approximated by an analytical description of the surface with straight-line or curved segments), and during which boundary conditions and material properties are applied. The third step involves the actual finite-element simulation of the time-variable behavior of the created model under the established parameters. Once this step is completed, the output of the simulation process is visualized or postprocessed in the final step.

The objective of this chapter is twofold: (i) to explore the basic principles, theories, and techniques behind the execution of each component within this modeling chain, and (ii) to introduce the reader to software options along with trends and developments in which these methods and approaches could be better used in biomedical applications.

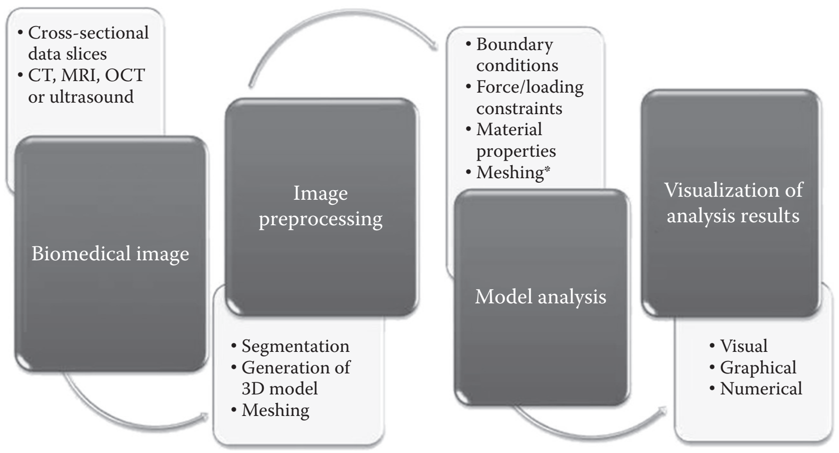

FIGURE 17.1
Flow process diagram for FEM of anatomical geometry extracted from medical imaging data. With a volumetric image as starting point, the first crucial step is the extraction of the geometry of the object of interest. For this purpose, a segmentation step is followed by the meshing step, in which an analytical description of the geometry is obtained. For this geometry, the constituent mechanical partial differential equations are then solved (i.e., the actual modeling step). The (*) symbol in the analysis step indicates that meshing can take place either during the preprocessing step or during the actual model analysis step. Lastly, the modeling results are visualized or further examined with respect to the property of interest, such as deformation, stress, or failure.

17.2 Segmentation

The preprocessing phase, which consists of the segmentation and meshing steps, is the most crucial and challenging component within the biomedical modeling chain. The segmentation process divides an image into meaningful regions in an attempt to delineate and extract objects or regions of interest (these are often referred to as *features* in the image). No uniform solution exists for the segmentation problem. The segmentation strategy strongly depends on the imaging modality, the object itself, and the varying image interpretations among different modalities. For example, CT allows for a relatively easy intensity-based segmentation of bone or the lungs because of the excellent bone-tissue and air-tissue contrast. Conversely, MR imaging can provide excellent contrast between tissues but is not usually used for imaging bone because of the low water content [16]. Frequently, image voxels are classified according to visual characteristics (e.g., intensity, texture, or color). Prerequisite is a form of contrast between healthy and abnormal tissue, or between the tissue of interest and adjoining regions—the latter are often referred to as *background* in a generalized sense.

A second goal of the segmentation process is the assignment of tissue properties, most often in inhomogeneous tissue regions. An assumed relationship between image intensities and material types is frequently the guiding standard in medical imaging segmentation application [5,17–21]. For example, the CT number of a bone voxel is related to its mineral content, and it can be argued that mineral content and stiffness are related [22,23]. Therefore, some empirical material properties are assigned to the voxel based on its CT number.

The segmentation step can be performed manually, with limited manual assistance, or in an automated fashion. In a manual segmentation step, a trained observer (such as the radiologist) delineates the boundary of the object. This process is often performed in two dimensions on a slice-by-slice basis. Manual segmentation suffers from variability owing to subjective influences. Computer-aided segmentation or fully automated segmentation is desirable to reduce intra- or interobserver variability [24,25] and to accelerate the segmentation process. A brief overview of some commonly used segmentation approaches is given in Sections 17.2.1 through 17.2.3. For a more comprehensive overview or for the underlying mathematical theories, the reader is referred to the pertinent literature [24,26–29].

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.2.1 First-Generation Processes

First-generation processes solely rely on information available within an individual image (i.e., its voxel values) and therefore are purely image based. Manual segmentation largely relies on the same contrast properties that enable these first-generation methods.

Automatic multistep segmentation approaches (discussed later) usually incorporate a combination of these first-generation methods or use them as a foundation for higher-level steps. First-generation processes can be partitioned into the following categories:

- *Intensity thresholding* (Figure 17.2b)
  - Intensity-based segmentation method that uses contrasting intensity levels to identify distinctive regions.
  - Each image pixel is compared to a threshold value such that all values
    - Above this range are labeled as *feature*
    - Equal to or below the indicated value are classified as *background*

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

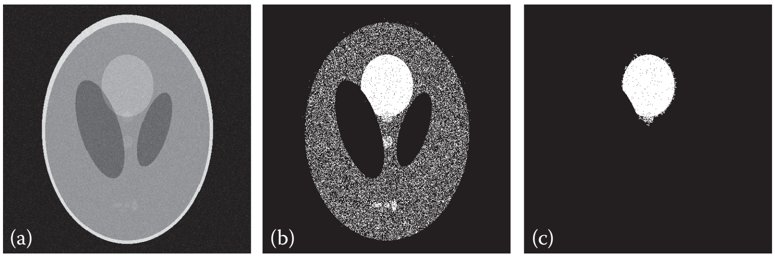

FIGURE 17.2
Intensity-based segmentation in a noisy variant of the Shepp–Logan head phantom. The additive noise causes major overlap of the image values between the *gray matter* and the large central *tumor* (a). Pure intensity-based thresholding (white pixels are those from (a) which lie above the threshold) cannot separate the tumor region from the gray matter region (b). When connectivity is considered, for example, with the region-growing algorithm, the segmented pixels are constrained to a connected region, and unconnected pixels within the thresholded intensity range are excluded (c).

- The ideal image case would contain pixels with nonoverlapping intensity values between feature and background.
- Often, a suitable transformation (e.g., local texture filters) can provide additional contrast.
- Multidimensional thresholding is an alternative option when a voxel is described by multiple criteria, such as intensity and one or more local neighborhood properties (e.g., smoothness or proximity to a gradient). When a voxel is associated with multiple orthogonal criteria, these are referred to as *feature vector.*

- *Region-based methods* (Figure 17.2c)
    - Extension of the intensity-based thresholding method under the assumption that the object forms a contiguous region.
    - Assumption of connectivity allows the separation of disjoint image features even if these have similar intensity.
- *Edge/boundary based* (Figure 17.3)
    - Relies on the identification of intensity gradients or sharp transitions of intensity levels.

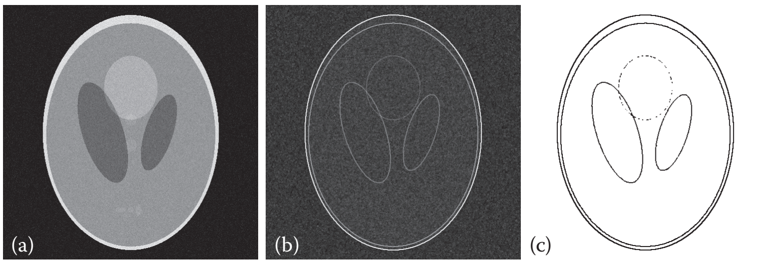

FIGURE 17.3
Edge-based segmentation. The source image is the head phantom with additive noise used in Figure 17.2 (a). The application of an edge enhancement filter (Sobel operator) converts gradients into higher-intensity pixels (b). As a side effect, the noise component is amplified. After thresholding and removal of isolated noise pixels, the edges remain; however, low-contrast features do not necessarily have a closed boundary, and the features with the lowest contrast have disappeared altogether (c).

- The anticipated result is that all of the pixels located near the gradient boundaries have higher intensity values than those lying outside or inside the object.
- The resulting image highlights the object surface rather than the volume; intensity-based thresholding is possible (Figure 17.3c).

17.2.2 Second-Generation Processes

Second-generation processes rely on the same intensity or contrast information that first-generation processes use but attempt to describe the image feature at a higher level of abstraction. Often, some form of numerical optimization or minimization is involved (purely discrete algorithms), or a physical model is approximated by numerical methods. Second-generation methods are purely image-based derivations of first-generation methods that occasionally use information gathered directly from initial first-generation segmentation results. The need for second-generation methods can be illustrated with the edge-based segmentation in Figure 17.3c, where it is desirable to obtain a closed contour for the large *tumor* in the same fashion as for the skull and the ventricles. Second-generation processes can be subdivided as follows:

- *Continuous Model Discretizations*: Based on a physical model, such as elastic contraction or viscous flow, but with an external, image-based driving force that lets the model converge on an image feature.
  - *Snakes*—Snakes are models of elastic rubber bands that are subjected to a stretching and a bending force. The external force is the negative image gradient. In the snake concept, a closed path contracts until an equilibrium is reached between the curvature-based internal force responsible for the contour's smoothness and the external force of the intensity gradient [30]. Figure 17.4 highlights the iterative progression of this process.
  - *Level Set Active Contours*—Level set active contours are implicit methods (in contrast to the explicit numerical formulations underlying the snakes). Level set–based active contours are derived from a numerical method designed to track an evolving contour deforming at a rate of speed based on curvature and gradient [31]. The model ceases deformation once the speed reaches an

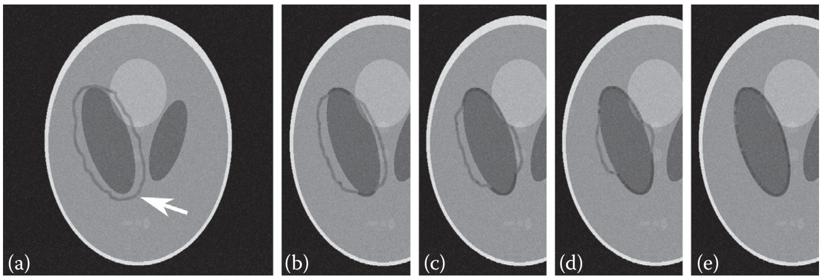

FIGURE 17.4
Semisupervised segmentation with active contours (snakes). A hand-drawn region (arrow) near one of the *ventricles* of a noisy Shepp–Logan head phantom serves as the starting point for the snake (a). The snake algorithm causes the snake to contract, but vertices are attracted by edges. Beginning contraction after five iterations (b); some convergence can be seen after 10 iterations (c); most of the snake has locked onto the edges after 20 iterations (d); final convergence of the snake (e). The vertices of the snake coincide with the edge of the ventricle, and the curve defined by the vertices serves as parametrization of the contour.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

artificial preset speed assigned to desired boundary areas. Their main benefit over snakes is their ability to change the topology [32].

- *Live Wire*—The live wire formulation is a semiassisted method where a cost function is established from representative boundary points identified by the user. The live wire algorithm then connects those points along a minimum-cost path [33].

- *Purely Discrete Algorithms*:
  - *Clustering*—Automatic assignment of pixels to one of several classes (e.g., feature and background) based on minimum-distance criteria for a vector of descriptive metrics
    - *k*-means—A pixel belongs to exactly one class, with which it has the smallest distance to the class centroid [34].
    - Fuzzy *c*-means—A pixel initially belongs to a fuzzy extent to multiple classes, and final class membership is decided after the optimization process [35].
  - *Graph Based* (see below)
    - Watershed segmentation
    - Fuzzy connectedness

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

In graph-based methods [28,36], the nodes of the graph correspond to the voxels. The graph itself is the set of all paths that connect a voxel inside the feature with a background voxel. Some criteria can be found where a path traverses a discontinuity (such as an edge) that defines the object boundary. At this point, the graph is cut. The remaining connected regions show a higher voxel homogeneity with respect to the segmentation criteria than the uncut graph. It is possible to express the links of the graph as a cost function similar to the one that drives the live wire, and training of the cost function is performed with some user-selected seed points.

The watershed and fuzzy-connectedness algorithms are based on the graph-cut principle. Watershed segmentation demarcates the boundaries of regions on a topological surface using *watershed* lines [37]. The topological surface is built on the interpretation of pixel intensity as elevation. The object of interest is assumed to be the set of pixels that becomes flooded, and the low-elevation regions that become flooded first serve to cut disjunct features. Thus, watershed segmentation is used to separate overlapping features that are connected in a first-generation segmented image. Fuzzy connectedness [38] assigns a continuous voxel similarity to pairs of voxels, and the graph (in the sense introduced above) is cut at a certain level of dissimilarity.

Statistical pattern recognition and neural networks are related image-based segmentation methods that are often collaborative [39]. In statistical pattern recognition, a model is used to assign image pixels to a known set of classes. Neural networks can be trained to segment images based on pixel values manually designated to classes. Neural networks in particular rely on high-order feature vectors that include the voxel intensity and a number of neighborhood criteria that can be generated through texture description methods (e.g., Laws' texture energy [40]).

17.2.3 Third-Generation Processes

Third-generation methods of geometry extraction are model-based recognition procedures that require extensive a priori information. A simplified explanation to third-generation processes can be given when we consider Figure 17.3. The edge contours are known to be elliptical (this is the a priori knowledge). The image can now be searched for voxels that belong to an ellipse, for example, with the Hough transform [41]. In this fashion, the fragments of the central ellipse in Figure 17.3c would be recognized as belonging to one ellipse, and the segmentation would yield the completed ellipse rather than the fragments.

More generally, third-generation processes are based on the construction of a spatial statistical model for the object of interest that consists of prominent shape details and any potential variations from a set of training images. During analysis, image data are scanned to locate regions that resemble the model. *Active shape models* identify objects within an image set that are recognized as members of the same class by using landmark points [42]. Image patches, such as object intensity or texture, are incorporated to add region-based features in *active appearance models* [43]. In *atlas-based segmentation*, a composite image is constructed from segmented images of many subjects. Descriptive information besides the geometry and shape, such as intensity properties, object labeling, and relationship definitions, is also stored with the atlas. An image set is first matched to the atlas template via three-dimensional (3D) mapping, followed by the atlas using the accompanying descriptive properties for statistical pattern recognition.

Hybrid segmentation methods merge the complementary strengths of separate methods with an aim to create more complex methods that increase accuracy and precision while simultaneously overcoming some of the individual limitations. Many of the recent advances in segmentation have contributed to the varying combinations of the aforementioned techniques. Some comprehensive surveys and reviews are based on method type [44], medical application [45,46], image type and method [47], medical application and image type [48,49], or, more specifically, medical application, image, type, and method [50].

Almost all of the above segmentation methods are supervised, and some level of user interaction is required throughout the process. Hybrid methods incorporating unsupervised processes can significantly reduce the level of operator assistance. Rule-based systems attempt to achieve effective, unsupervised segmentations. In this multistep process, first- and second-generation methods extract desired image features, which are interpreted and labeled by third-generation knowledge-based approaches. The segmentation is carried out according to an explicit predefined set of rules [51]. Neural networks and fuzzy-connected filters are commonly implemented in rule-based systems and have been used in brain tumor extraction from MR images [52–54], breast cancer lumps [55], and skin cancer lesions [56].

At the end of the segmentation process, the image has been subdivided into the background and one or more features (objects) that will be analyzed in the finite-element simulation. At this point, the voxels are merely labeled as belonging to the background or to the image feature (or features if multiple materials are present). In the next step, the shape of the objects formed by those voxels needs to be described analytically such that the proper set of governing equations can be set up.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.3 Meshing

In conventional engineering FEM applications, the meshing step is relatively straightforward, because an analytical description of the object usually exists. Notably, solid modeling software often allows automatic subdivision of a 3D model into nodes and elements and assigning material properties to the elements. The resulting finite-element shapes in these 3D models can be extensions of either a quadrilateral in the form of a *hexahedron* (8 vertices, 12 edges, 6 faces) or a triangle evolved into a *tetrahedron* (4 vertices, 6 edges, 4 faces).* The mesh is of fundamental importance, because it directly gives rise to the set of partial differential equations that are used in the FEM solver (i.e., the simulation step) to determine the unknown quantities of the simulation. The process of discretizing a known geometrical shape can be seen as a *top–down approach* to generating a finite-element model.

Conversely, the irregular shape of biomedical objects—especially when obtained from volumetric images—requires a piecewise analytical approximation of the shape. Usually, a large number of points on the surface is determined, and these points are connected with high-order curves, such as splines. The process of approximating a shape with discrete nodes can be seen as a *bottom–up approach*. The element shape, mesh arrangement, and algorithm selections to generate the most accurate mesh for evaluating biomechanical behavior are widely dependent on the analysis emphasis, material type, complexity of geometry, method of FEA, and other application characteristics. Adjustment or modification of one application parameter could potentially require the use of an entirely different meshing approach.

Moreover, in biomedical applications, the material properties are often unknown or uncertain, and a rigorous match between elements and their associated material properties cannot be obtained in a straightforward manner. For these reasons, *the overall meshing process based on patient-specific medical image data remains the most crucial bottleneck in the entire FEA chain*. However, a large body of literature has emerged in recent years where improvements and modifications are presented for tailored and application-specific mesh generation, and this section provides an overview of some of the most relevant approaches.

Image-based mesh generation begins with the segmented object, which is the outcome of any of the processes described in Section 17.2. To provide the necessary input for FEA, three steps are taken:

- *Surface mesh construction*: The shape of the object is discretized by a large number of nodes that are placed as close as possible to the surface of the segmented object. Ideally, these nodes are either evenly spaced or become denser in regions of higher curvature.
- *Volume mesh construction*: The areas between the surface boundaries can be meshed in a manner similar to the analytical top-down approach; that is, the object's volume is subdivided into a large number of small hexahedral or tetrahedral elements. Volume meshing usually begins at the nodes of the meshed surface. Similar to conventional mesh generation of computer-aided design (CAD)–based models, the elements are rigidly connected at the nodes and provide the boundary conditions for neighboring elements.
- *Application of material properties and external boundary conditions*: Material properties must be defined for all volume elements. External forces or spatial constraints can be applied to some of the nodes.

* In FEM terminology, the reference is the number of facets. A hexahedral element has six sides, and a cuboid would be a hexahedral element. A *hexagonal* prism, in FEM terminology, would count as octahedral.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.3.1 Surface Mesh Construction

A 3D image volume is the yield, or end product, of the segmentation process. Each voxel within this volumetric scalar field is assigned a value to represent pixels that are a part of either the background or any of one or more segmented objects (including their associated material types) within the identified geometry. Two general mechanisms have been implemented to fit surfaces to data [57]: isosurface extraction and adaptive contours. Not coincidentally are these related to the first- and second-order processes in the segmentation step.

Isosurface extraction uses piecewise linear interpolation algorithms such as the marching cubes algorithm [58] or level sets, to subdivide (tessellate) the surface into triangular surface facets. For the isosurface extraction, the volume of interest is embedded into a regular, cuboid mesh. The assumption is that the object has higher voxel values than the background. Each node of the mesh is now checked whether it is above the isosurface value (i.e., lies within the object) or below the isosurface value (i.e., is part of the background). Edges that connect nodes inside and outside the object intersect the surface, and the intersection point is determined by linear interpolation. The intersection points then form the nodes of the surface mesh. Edges are shared by multiple cubes, and consequently, surface facets are connected. Whenever the intersection of any cuboid with the isosurface renders quadrilaterals or higher-order polygons, these are further subdivided into triangles, because triangles are always planar, and thus have a defined surface normal. Figure 17.5 demonstrates how the algorithm constructs triangular elements.

The alternative approach to surface reconstruction of a segmented volume is a 3D adaptation of the local energy-minimizing snake algorithm [59]. A net wraps around, and conforms to, a specific volumetric feature based on internal energy and image force calculation between the binary surface and the parametric surface in an iterative manner. The points on the surface and other parameters, for example the number of nodes, must be preassigned to determine the net's initial geometry. The definition of such parameters

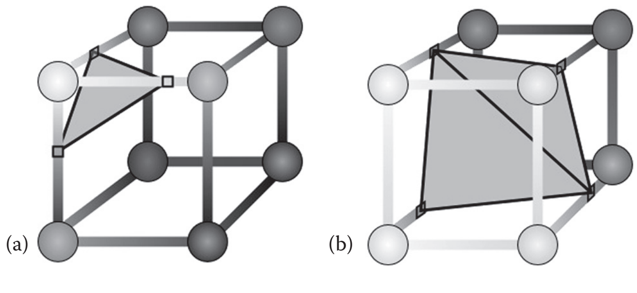

FIGURE 17.5
Two examples of an isosurface intersecting a cube. In both cases, the image value of the voxel centers (spheres) is known, and the image values between voxel centers are obtained by linear interpolation (thick lines with gradients). Interpolation provides the location of the desired isosurface value between two adjoining pixels, and this location serves as one vertex of the tessellated surface (small squares). (a) Only three edges cross the isosurface value, and all other edges lie below the isosurface value. The resulting element is a triangle. (b) The four front voxels have values above the isosurface value; the four back pixels lie below. Consequently, the vertices lie on the diagonal edges and form a quadrilateral. Since tessellation requires planar (triangular) elements, the quadrilateral is subdivided once.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

contributes to the model's smoothness. A smoothed approximation of the surface by the model nodes is the output from this process, and nodes are connected to form triangles.

17.3.2 Volume Mesh Construction

The surface mesh can now be transformed into a FEM *volume mesh* by filling the interior region with the preferred element shape and arrangement. Mesh types can be broadly classified into *structured* or *unstructured* meshes according to their geometrical pattern. *Structured* mesh arrangements consist of an evenly spaced, fixed, uniform amount of nodes and elements and are composed entirely of hexahedral elements that are a result of grid-based surface construction algorithms. This template-based connectivity pattern was the initial attempt at meshing automation, and it produces meshes that approximate curved surfaces in a stair-step fashion. The disadvantage of a structured mesh is the inability to adapt to the shape curvature: the node density is the same in regular regions, where fewer nodes would yield an adequate discretization, and in irregular regions, where the node density may not be sufficient to describe the surface with adequate detail.

Unstructured meshes contain either all tetrahedral elements or a combination of hexahedral, tetrahedral, and wedge-shaped elements generated automatically in arbitrary configurations. The relationship between the number of vertices, edges, faces, and regions is unknown beforehand. Unstructured meshes are better capable of adhering to curved boundaries while being adaptable to general complex shapes with less user intervention. Figure 17.6 demonstrates the difference between structured and unstructured meshes.

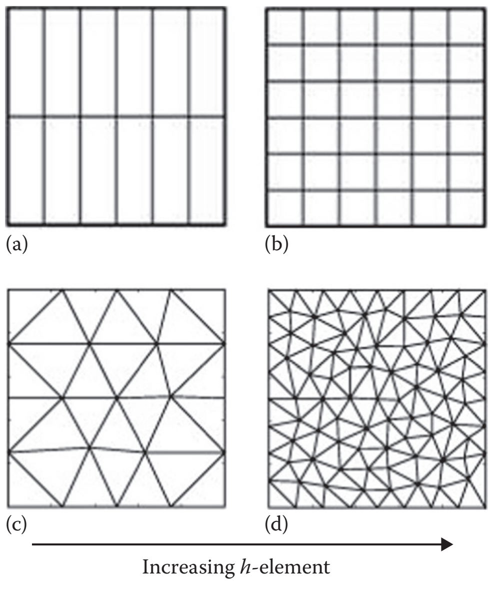

FIGURE 17.6
Comparison of mesh types. (a and b) Structured mesh with quadrilateral elements (corresponding to hexahedral elements in 3D); (c and d) unstructured mesh with triangular elements (corresponding to tetrahedral elements in 3D). The effects of increasing the *h*-element constraint can also be observed. The finer mesh patterns in (b) and (d) with smaller element size and increased element quantities are derived from the original patterns in (a) and (c) by subdivision of the original elements.

The following algorithms have been established for the generation of unstructured mesh types:

Tetrahedral-based mesh

- *Octree* [60,61]: Recursively partitions a cube containing the geometry into eight octants until a preferred resolution is reached. Nonuniform tetrahedral elements can be formed between the intersections of these cubes.
- *Advancing Front* [62,63]: Begins at a boundary and then progresses toward empty space within a region.
- *Delaunay* [64]: Creates nearly equilateral triangles (which are the faces of tetrahedrons) based on satisfaction of an *empty circle* criterion. For this method, each edge of a triangle lies near the edge of a circle, and vertices are prevented from being located within the circle's circumference. This approach minimizes the occurrence of *skinny* triangles that have very small angles or form sharp, jagged edges (see Figure 17.7).

Hexahedral-based mesh

- *Plastering* [65] is a hexahedral adaptation of *advancing front* mechanism. It begins with an all-quadrilateral mesh that projects faces into a volume, thereby creating hexahedral elements in an iterative manner.
- *Medial Surface* [66]: Decomposes a volume into hexahedral elements.
- *Grid Based* [67]: Imposes a 3D grid of hexahedral elements within volume interiors.
- *Whisker-Weaving* [68]: Builds a dual or spatial twist continuum with bisecting intersecting surfaces that fit hexagonal elements derived from quads into the volume based on connectivity followed by the subsequent determination of nodal locations.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

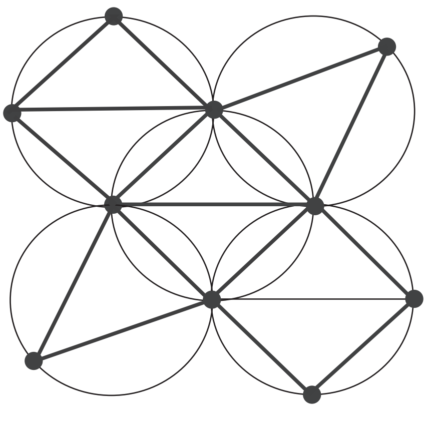

FIGURE 17.7
Triangular/tetrahedral mesh construction based on Delaunay algorithm. All points or vertices of each triangle lie near the edge of a circle, because the *empty circle* criterion prevents vertices from being placed within the circle or sphere's circumference.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.3.3 Mesh Quality, Optimization, and Adaptive Refinement

Meshes must be feature preserving in the sense that they must be as close as possible to the original surfaces and be capable of handling complex topology [69,70]. Ideally, they also possess the adaptivity to increase node density in areas of high interest (e.g., areas of concentrated stress or high curvature) to limit the surface discretization error, while balancing the number of mesh elements. A mesh composed of nearly equilateral triangles is desirable, and vertices need to be seamlessly aligned with neighboring vertices.

A frequently employed strategy is to generate an initial mesh that is then refined until convergence is reached, that is, until further mesh refinement changes the simulation results by less than a predefined margin. The method to generate the initial mesh is referred to as *primal contouring method*, and the marching cubes algorithm is one example for a primal contouring method. The resulting initial mesh approximates the implicit geometry but generally has undesirable features that need to be suppressed. Common observations in meshed objects from biomedical images include the following [71,72]:

- A uniform stair-step surface appearance that does not fully align with the natural surface curvature
- Badly shaped triangles (e.g., triangles with very large and small angles that cause sharp, jagged edges)
- Excessive amounts of irrelevant nodal and faceted data
- The nonpreservation of sharp features within the data

Methods for subsequent refinement are referred to as *dual contouring methods* [73]. Dual contouring methods were derived to improve the quality of meshes with respect to the accurate reproduction of sharp features [70]. In the basic dual contouring algorithm, the region to be meshed is divided into overlapping cubic grids. The surface is then analyzed at vertices of the grids and classified as either being inside or outside the mesh. Cubes consisting of inside and outside combinations of vertices are denoted as containing surface portions. The dual contouring process creates a vertex pair per cube that straddles the surface, and the connection of each to the neighboring vertex pair shapes the final mesh.

Additional refinement is based either on local remeshing through swapping of edges or faces or on local refinement through insertion or deletion of vertices [74]. In fact, the methods described in Section 17.3.2 for the generation of unstructured meshes can be used for mesh refinement as well. The criteria that the octree, Delaunay, and advancing front algorithms act upon are responsible for the swapping of edges/faces and insertion or deletion of vertex points that lead to the final positions and number of the vertices of the mesh. It is at these locations at which the error is minimal between the planes and normals at the edge intersections [75–77].

In addition to dual methods, mesh smoothing can be applied. Smoothing occurs when vertices are relocated following some regularity criterion, but relocation does not affect the topology of the mesh. For example, several sweeps of a spatial Laplacian smoothing operator relaxes each adjustable vertex to the arithmetic average of adjacent vertices [78]. Alternatively, the smoothing could be incorporated into an optimization algorithm to allow adjustment of either individual or independent sets of vertices in parallel [79]. The amount of smoothing a vertex is subjected to could be controlled by hierarchical order of classification. Vertices are categorized by increasing rank levels, and those within the highest class are most affected by the smoothing operations.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

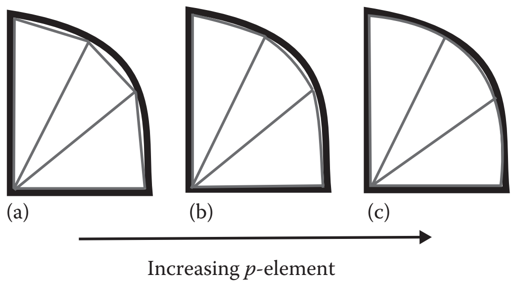

FIGURE 17.8
Observation of increasing p-elements in triangular/tetrahedral mesh. First-order linear elements do not capture the curved boundaries of the model (a). The mesh improves its adaption to the curvature of the model when the order of the p-element is increased to quadratic (b). Higher-order cubic element allows even better optimization as the mesh more accurately adapts to the curved regions of the model (c).

In mesh quality analysis, there are two types of element classifications that are applicable to both hexahedral and tetrahedral shapes. The low-order linear or quadratic *h-element* corresponds to the step size needed to converge the biomechanical behavior solutions or minimize the error obtained for the actual analysis. A smaller mesh size h represents a finer mesh consisting of a larger number of elements in the model. This is how a finer mesh in areas of high interest (e.g., areas of stress concentration) can be constructed. The *p-element* is a metric for the optimization of the mesh at a higher, polynomial, order: in p-element refinement, edges are no longer straight lines, but polynomial curves. The increased polynomial order p of the element allows the element edges to adapt to curved boundaries more accurately and thereby leads to minimization of discretization error and solution convergence during the analysis of the model. In contrast to h-elements, fewer elements are required to obtain convergence, and the surface discretization error can be reduced without increasing the number of elements within the mesh. Figure 17.6 demonstrates how a mesh can be refined by h-elements, while optimization using p-elements is illustrated in Figure 17.8.

17.3.4 Tetrahedral versus Hexahedral Elements in Biomechanics

The preference of tetrahedral or hexahedral elements in FEM continues to be an unsolved topic of debate. Each has advantages and disadvantages. Tetrahedral elements are widely selected because of their geometrical flexibility, and they produce acceptable displacement behavior [80]. Hexahedral elements are well suited to model stresses and strains in tissues, and hexahedral elements are assumed to lead to more accurate simulation results. On the other hand, inaccurate approximation of the actual shape can be observed at jagged edge areas [81]. In addition, hexahedral meshes are more challenging to generate than tetrahedral meshes.

Meshes with tetrahedral elements can match simulation accuracy of hexahedral meshes when more and smaller elements are used (i.e., increased h-element). Four to 10 times the amount of elements are needed in tetrahedral meshes in order to achieve comparable levels of accuracy provided by hexahedral elements [82–84]. Tetrahedral elements of lower order are unduly stiff and lock in instances of modeling extensive stress deformations or materials that are nearly incompressible. They are also more prone to a failure of the inversion problem during analysis [85]. Many of these differences can be observed and justified when the examples in Figures 17.6 and 17.8 are considered. One possible solution to this dilemma is to create hybrid meshes that are constructed with an outer surface of all-tetrahedral elements and an inner volume of all-hexahedral elements [6,86].

17.4 Simulation

Once the mesh is generated, the actual analysis can be performed. The analysis consists of the assignment of material properties and boundary conditions, followed by numerically solving the constituent partial differential equations. Overall, this step represents a numerical *simulation* of the material behavior under the given boundary conditions.

It is debatable whether the assignment of material properties and boundary conditions is part of the meshing process or part of the simulation process. In meshes obtained from medical images, material properties are often extracted from the image. However, because material properties and boundary conditions can easily be changed for simulation purposes, they are covered in this section.

17.4.1 Boundary and Loading Constraints

Some of the nodes and elements of the mesh must be supplemented with additional information that is strongly dependent on the analysis application. Accurate simulation may require certain regions of the model to be loaded with prescribed displacements, nodal forces, pressure forces, body forces, or surface tractions, velocity, or flux. Such conditions occurring on the boundary are referred to as nonessential or *Neumann boundary conditions.* Specifications also need to be made in reference to how the model interacts with its surroundings. These are the essential or *Dirichlet boundary conditions* and refer to the rotation, support, or fixation of the region.

17.4.2 Material Properties

Material characteristic values (e.g., elastic modulus) are then assigned to the model in order to numerically distinguish it as bone, soft tissue, muscle, fluids, or a combination of materials from an engineering aspect in terms of strength, elasticity, durability, conductivity, and porosity. Material properties can be

- *Homogeneous*: Same at all locations
- *Nonhomogeneous*: Location dependent
- *Isotropic*: Same in any direction
- *Anisotropic*: Direction dependent
- *Orthotropic*: Symmetric with respect to x–y, y–z, or x–z planes

Examples for isotropic tissues are those in the cardiovascular system and neurological tissues. Typical anisotropic tissues are ligaments, tendons, and cartilage. Both cortical and trabecular bone exhibit orthotropic behavior. Soft tissue also exhibits *creep*, that is, its stress–strain behavior is time dependent. FEM solvers need to be capable of incorporating these different types of material descriptions for the simulation of biological tissues.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.4.3 Governing Principle

In the basic theory of FEM, each subdivision or element is associated with a finite number of degrees of freedom, which are the unknown function values at the nodal points that the element is composed of. The constraints, mechanical properties, and loads applied at the nodes are condensed into discrete differential equations corresponding to the element's response. A node can be shared by several elements; therefore, the deflection at the shared node is representative of deflection of the sharing elements at the node location. Shape functions use interpolation to approximate deflection values occurring at all nonnodal points within an element. The assembly of these individual element equations leads to the generation of the global equations representing the entire meshed region. Both the individual and global equations are always expressed in the form

$$\{F\} = [K]\{u\}, \tag{17.1}$$

where

$\{F\}$ is the column matrix of the applied external force or nodal loads.

$[K]$ is a stiffness matrix containing the geometric and material information that determines the element's (or the mesh's) resistance to deformation when subjected to loading. Specifically, this matrix is

Symmetric, as a result of equal and opposite forces to ensure equilibrium.

Singular, since the equation has not been solved. Boundary conditions must be included at this point to alleviate this.

$\{u\}$ is the column matrix of nodal displacements (i.e., the degrees of freedom) resulting from the application of load $\{F\}$. These values are interdependent since the body is continuous and elastic.

Equation 17.1 is solved for the nodal displacements $\{u\}$. Once this information is obtained, the stresses (σ) can be determined from kinematic relationships, and the strain (ε) is calculated through material or constitutive relationships in solid mechanics analysis. The general form of Equation 17.1 governs many physical phenomena in which the parameters $\{u\}$ and $\{F\}$ have different physical significance. It should also be noted in applications in which there is no measure of deformation that the matrix $[K]$ is still referred to as the *stiffness matrix* to maintain uniformity in nomenclature and definition. Table 17.1 lists these physical phenomena and the interpretation of $\{u\}$ and $\{F\}$ for some applications.

TABLE 17.1

Interpretation of the Displacement Matrix $\{u\}$ and Load Matrix $\{F\}$ in Different Physical Problems That Obey the General Form of Equation 17.1

Analysis/Application	$\{u\}$	$\{F\}$
Acoustic fluid	Displacement potential	Particle velocity
Electrostatics	Electric potential	Charge density
General flow	Velocity	Flux
Heat conduction	Temperature	Heat flux
Potential flow	Pressure gradient	Velocity potential
Magnetostatics	Magnetic potential	Magnetic intensity
Structures and solid mechanics	Displacement	Mechanical load

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

It is possible to combine physical phenomena and subject them to a joint simulation. One example is blood flow in the vasculature. Fluid flow exerts shear stress on the tissue at the interface and can cause viscoelastic deformation. Deformation can also be caused by fluid pressure, such as blood pressure. Such models are referred to as *biphasic models*. The generation of biphasic models, especially those with nonlinear properties, can lead to models of appreciable complexity; however, these models still follow the general outline presented in Section 17.4.4.

17.4.4 Approximate Solution Determination

FEA solvers commonly obtain the approximate solution to Equation 17.1 via an indirect formulation method based on the minimum total potential energy (TPE). The TPE (Π) for a model composed of n elements and m nodes is the difference between the total strain energy (Λ) and the work done by the external forces ($F{\cdot}u$) and is given by

$$\Pi = \sum_{e=1}^{n} \Lambda^{(e)} - \sum_{i=1}^{m} F_i u_i, \tag{17.2}$$

where the strain energy per unit volume for linear elastic materials is

$$\Lambda^{(e)} = \frac{1}{2}\int \sigma \in \mathrm{d}V. \tag{17.3}$$

This is ideal for models with solid materials where the strain energy of the system can be calculated. During deformation, the work done by the applied load F is stored as elastic or strain energy. The model is considered to be at equilibrium when the TPE is minimal with respect to displacement u. Therefore the solution is found by

$$\frac{\partial \Pi}{\partial u_i} = \frac{\partial}{\partial u_i}\sum_{e=1}^{n} \Lambda^{(e)} - \frac{\partial}{\partial u_i}\sum_{i=1}^{m} F_i u_i = 0 \tag{17.4}$$

in accordance to the law of conservation of energy. Each finite element will have its own TPE.

For simplicity, we will use the discrete connected system of springs in Figure 17.9 to illustrate the FEA procedure. There are four springs, three nodes, and two external loads present in the system. The TPE is given by

$$\Pi = \frac{1}{2}k_1\delta_1^2 + \frac{1}{2}k_2\delta_2^2 + \frac{1}{2}k_3\delta_3^2 + \frac{1}{2}k_4\delta_4^2 - F_1u_1 - F_3u_3, \tag{17.5}$$

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

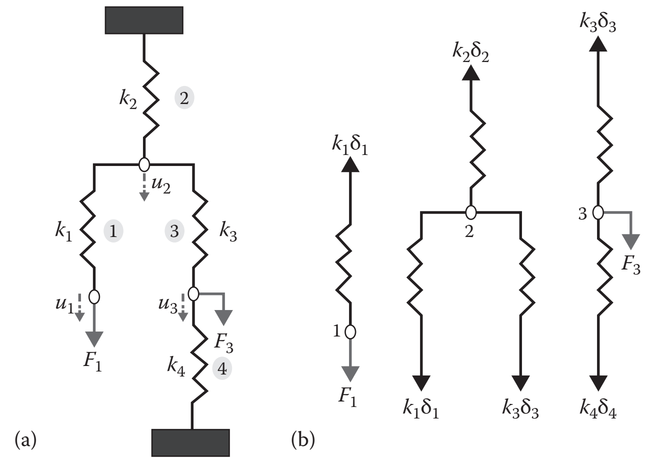

FIGURE 17.9
Discrete spring system used to derive FEA theory (a). The system is composed of four springs and is subjected to two external loads F_1 and F_3 as well as three displacements u_1, u_2, and u_3. Free-body diagram representing each spring component of the system's mechanical behavior (b).

where δ_1, δ_2, δ_3, and δ_4 correspond to the amount each spring stretches. These can be rewritten as

$$\begin{aligned} \delta_1 &= u_1 - u_2 \\ \delta_2 &= u_2 \\ \delta_3 &= u_3 - u_2 \\ \delta_4 &= -u_3. \end{aligned} \tag{17.6}$$

Now, substitution into Equation 17.5 yields

$$\Pi = \frac{1}{2}k_1(u_1 - u_2)^2 + \frac{1}{2}k_2u_2^2 + \frac{1}{2}k_3(u_3 - u_2)^2 + \frac{1}{2}k_4u_3^2 - F_1u_1 - F_3u_3 \tag{17.7}$$

such that u_1, u_2, and u_3 are the displacements of each node.

In order for equilibrium to be reached, the TPE needs to be minimized with respect to the three displacements as explained in Equation 17.4. Therefore, each spring's mechanical behavior can be defined by a set of differential equations:

$$\begin{aligned} \frac{\partial \Pi}{\partial u_1} &= k_1(u_1 - u_2) - F_1 = 0 \\ \frac{\partial \Pi}{\partial u_2} &= -k_1(u_1 - u_2) + k_2u_2 - k_3(u_3 - u_2) = 0 \\ \frac{\partial \Pi}{\partial u_3} &= k_3(u_3 - u_2) + k_4u_3 - F_3 = 0 \end{aligned} \tag{17.8}$$

These differential equations are converted into algebraic equations, followed by each k value being arranged into a stiffness matrix using the general form:

$$K^{(e)} = \begin{matrix} k_e & -k_e \\ -k_e & k_e \end{matrix}, \tag{17.9}$$

where e represents the four spring systems (k_1, k_2, k_3, and k_4). Each of these system matrices can be combined such that a global stiffness matrix for the overall system is obtained and expressed as

$$K(G) = \begin{matrix} k_1 & -k_1 & 0 & 0 \\ -k_1 & k_1+k_2+k_3 & -k_2-k_3 & 0 \\ 0 & -k_2-k_3 & k_2+k_3+k_4 & -k_4 \\ 0 & 0 & -k_4 & k_4 \end{matrix}. \tag{17.10}$$

Based on the algebraic equation from each node,

$$\begin{aligned} &k_1\delta_1 = F_1 \\ &k_2\delta_2 - k_1\delta_1 - k_3\delta_3 = 0 \\ &k_3\delta_3 - k_4\delta_4 = F_3, \end{aligned} \tag{17.11}$$

and the boundary conditions of our system, the k_2 components in the third column as well as the fourth row and column of the matrix can be deleted owing to the spatial fixation of the nodes at those locations. Now, the algebraic form can be rewritten in the general form as presented in Equation 17.1

$$\begin{matrix} F_1 \\ 0 \\ F_3 \end{matrix} = \begin{matrix} k_1 & -k_1 & 0 \\ -k_1 & k_1+k_2+k_3 & -k_3 \\ 0 & -k_3 & k_3+k_4 \end{matrix} \begin{matrix} u_1 \\ u_2 \\ u_3 \end{matrix} \tag{17.12}$$

and solved by inversion of the matrix [K].

In order to observe the mechanical behavior of the spring system, let us assume that each spring has a uniform cross-section area A and length l when subjected to a force F. The average stress (σ) in a spring is given by

$$\sigma = \frac{F}{A} \tag{17.13}$$

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

and the average normal strain (ε) is the change in length Δ*l* per unit original length of the spring

$$\varepsilon = \frac{\Delta l}{l}. \tag{17.14}$$

Hooke's law relates stress and strain in elastic regions through the equation

$$\sigma = E\varepsilon, \tag{17.15}$$

where *E* is the modulus of elasticity of the spring's material. Rearranging and substituting Equations 17.14, 17.13, and 17.15 to isolate *F* results in

$$F = \frac{AE}{l}\ \Delta l, \tag{17.16}$$

which bears resemblance to the equation of a linear spring. Therefore, we can conclude in this case that each spring has a stiffness *k* of

$$k_e = \frac{AE}{l}. \tag{17.17}$$

As described above, solving for the displacement values {*u*} allows the stresses and strains to be determined. The stress for each spring is calculated by

$$\sigma = \frac{F}{A} = \frac{k_e(u_{i+1} - u_i)}{A} = \frac{\frac{AE}{l}(u_{i+1} - u_i)}{A} = E\ \frac{u_{i+1} - u_i}{l} \tag{17.18}$$

and the strain can be determined from Equation 17.15 or from the displacement of each spring at the nodes,

$$\varepsilon = \frac{u_{i+1} - u_i}{l}\ . \tag{17.19}$$

Although we presented a simplified example, the FEM simulation obeys the exact same principle, because each node of the volumetric mesh is examined in the same fashion as the nodes in Figure 17.9, with the spring constants determined by the neighboring nodes and the material properties in the respective element. It should be noted that in more complex two-dimensional (2D) and 3D models, the directional components of stress and strain are considered (stress and strain tensors) and that the stiffness *k* is dependent on the area of the model and the analysis application (i.e., solid mechanics, heat transfer, etc.).

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.4.5 Nonlinear Analysis

Biomaterials often exhibit nonlinear behavior. A deviation from linear behavior (i.e., Hooke's law) not only occurs for large geometry changes that lead to fracture but also includes plastic deformation and creep. These situations result in nonlinear and even time-dependent σ–ε and ε–*u* relationships, where occurrences of stiffening, hardening, softening, or buckling are possible. In such cases, the stiffness matrix [*K*] becomes time dependent. The solution is usually obtained by incremental or iterative methods, such as the implicit Newton–Raphson approach. In this approach, a load is applied in discrete time steps that advance the simulation from t to $t + \Delta t$. This technique addresses material nonlinearity by solving for the state at time $t + \Delta t$ when displacements at u^t are known. The load increment is applied such that the state is updated to $t + \Delta t$, and $u^{t+\Delta t}$ is the main variable. The principles of the Newton–Raphson approach can be integrated into the TPE method and represented as

$$\Phi(u^{t+\Delta t}) = \int \sigma \in (u^{t+\Delta t})\mathrm{d}V - F = 0, \tag{17.20}$$

where Φ represents the sum of internal and external residual forces and Φ = 0 is a set of nonlinear equations in {*u*}. Given that $\Phi(u^{t+\Delta t}) = 0$, for an *i*th iteration:

$$u_{i+1}^{t+\Delta t} = u_i^{t+\Delta t} - \frac{\partial}{\partial u}\Phi\left(u_i^{t+\Delta t}\right)^{-1} \Phi\left(u_i^{t+\Delta t}\right) \tag{17.21}$$

can be rearranged such that

$$\delta u_{i+1} = u_{i+1}^{t+\Delta t} - u_i^{t+\Delta t} = -\frac{\partial}{\partial u}\Phi\left(u_i^{t+\Delta t}\right)^{-1} \Phi\left(u_i^{t+\Delta t}\right), \tag{17.22}$$

where

$$[K_\mathrm{T}] = \frac{\partial}{\partial u}\Phi\left(u_i^{t+\Delta t}\right)^{-1} \tag{17.23}$$

is the total tangential stiffness matrix for which holds

$$K_\mathrm{T}\left(u_i^{t+\Delta t}\right) = K^a\left(u_i^{t+\Delta t}\right) + K^b\left(u_i^{t+\Delta t}\right) + K^c\left(u_i^{t+\Delta t}\right), \tag{17.24}$$

where

$[K]^a$ is a linear matrix that includes small strain and displacement model components.
$[K]^b$ houses linear and quadratic strain terms and is considered the large displacement or stiffness matrix.
$[K]^c$ is a nonlinear matrix composed of quadratic strains and nonlinearities associated with the material.

Substitution yields

$$\delta u_{i+1} = -K_{\mathrm{T}}\left(u_i^{t+\Delta t}\right)^{-1} \Phi\left(u_i^{t+\Delta t}\right) \tag{17.25}$$

and therefore

$$-\left\{\Phi\left(u_i^{t+\Delta t}\right)\right\} = K_{\mathrm{T}}\left(u_i^{t+\Delta t}\right) \left\{\delta u_{i+1}\right\} \tag{17.26}$$

is in the general form presented in Equation 17.1. This must be solved for each iteration i to advance the simulation by Δt and obtain the change in incremental displacements

$$u_i^{t+\Delta t} - u_i^t = \Delta u_i = \sum_{k=1}^{i} \delta u_k, \tag{17.27}$$

whereby K_{T} and Φ are different for each iteration. A quadratic convergence rate toward a tolerance threshold ε_{T} is achieved when

$$\left|\left\{\Phi\left(u_{i+1}^{t+\Delta t}\right)\right\}\right| < \varepsilon_{\mathrm{T}}. \tag{17.28}$$

Because Φ needs an accurate stress value (σ) to obtain a current approximation of $u^{t+\Delta t}$, the updated Lagrangian approach is implemented as the solution process moves from $t \rightarrow \Delta t$ to iteratively follow elements of the model along their configuration. All derivatives and integrals are determined with respect to spatial coordinates.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.5 Postprocessing: Output and Visualization

The solutions to equations evaluated in the analysis are stored as raw data. These solutions represent behavioral effects, such as stress, strain, pressure, or geometrical distortion attributed to the simulated forces. The postprocessing step can be either quantitative or qualitative, where the latter most often involves some form of *visualization* of the simulation results. The postprocessing step is the most application-specific step in the entire analysis chain, and only a general overview can be given.

Quantitative analysis would involve the examination of forces, stresses, or deformation in an automated fashion: computer software could, for example, provide a histogram of the stress magnitude for the examined volume. The histogram then displays the probability or absolute number of elements that experience a certain stress. Similarly, a threshold function would provide the simplest means to obtain a yes/no decision if material failure can occur.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

Qualitative analysis usually involves the examination of the simulation results by a trained observer. For this purpose, simulation results are usually presented in graphical, visual form. Most frequently, the simulation results (e.g., the stress magnitude) is superimposed over the original geometry. For 2D simulations, visualization is fairly straightforward, but in three dimensions, some form of projection is required, where the object of interest is displayed from a camera perspective. The visualization software allows the user to position the object in its virtual space: rotate it, move it, or zoom in or out. In this fashion, the observer can focus on critical parts of the geometry.

There is an almost limitless number of visualizations, and in many ways, a good visual representation of the result contains a considerable artistic element. In fact, it is recommended that a *visualization strategy* [87] is developed beforehand, which is determined by the key conclusions of the simulation. However, there are a number of commonly used tools in the visualization process. These include the following:

- *False-coloring*: Magnitude can be displayed in grayscale, but more frequently, the magnitude is mapped to colors, because the human eye is more sensitive to differences in hues than to differences in gray shades.
- *Contours*: Similar to contour lines in topographical maps, contour lines that connect locations of equal value can be introduced. Contour lines facilitate the assessment of gradients as well as the shape of regions of interest.
- *Glyphs*: Vector fields (e.g., forces) can be visualized with glyphs. Glyphs are small 3D objects, such as arrows, that are placed along a vector field. Their size and direction indicate magnitude and direction of the vector field.
- *Partial transparency*: Most often, only the outer surface of a 3D object is visible. Scalar fields can be represented by an isosurface. However, some software programs make it possible to introduce transparency to a solid volume, and the amount of transparency can be made magnitude dependent. In this fashion, parts of a volume can be made *see-through*, allowing the visualization of different elements inside the volume.
- *Clip planes*: A clip plane allows cutting away parts of a 3D object, and the interior becomes visible at the cut surface.
- *Animation*: The time-dependent evolution of the simulation was introduced in Section 17.4. Visualizations can include animations to illustrate the time-dependent behavior of, for example, forces, stresses, and deformation.

In one specific case (GiD visualization software, CIMNE, Barcelona, Spain), deformation can be exaggerated. A likely approach is to take the spatial coordinates of each node at two time points t and $t + \Delta t$ and to compute the displacement vector. In the visualization, each node is then again displaced by a multiple of its displacement vector. Since material deformation often occurs on a small scale, this exaggeration can make deformation behavior particularly well visible.

Some of the visualization techniques are demonstrated in Figure 17.10 with the help of Paraview software. A more complex visualization, created with OpenDX, is shown in Figure 17.11, where the stress magnitude is represented by glyphs and the pressure is imprinted in false colors on an oblique clip plane. The geometry is superimposed in a transparent gray shade.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

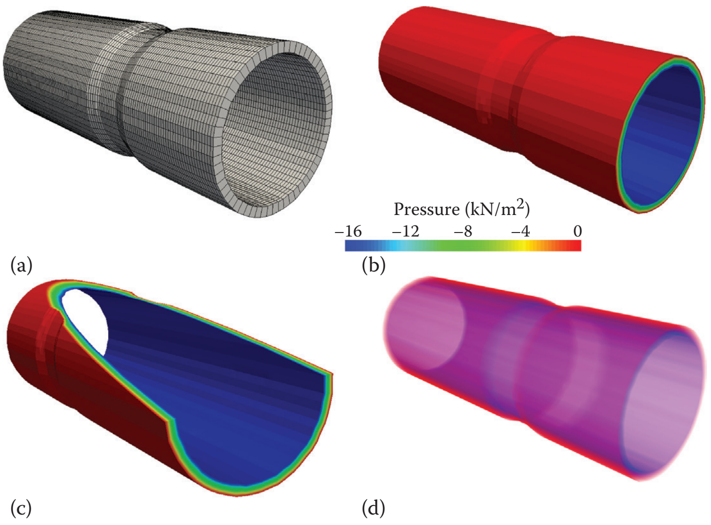

FIGURE 17.10
Visualization examples of a blood vessel phantom model. (a) The meshed model of the tubular phantom. (b) View of the phantom after FE simulation with color-coded pressure values superimposed (surface view). (c) View of the phantom with color-coded pressure, but after application of an oblique clip plane. This view better reveals the pressure distribution along the phantom and highlights a thinner section. (d) Rendering of the phantom with partial transparency.

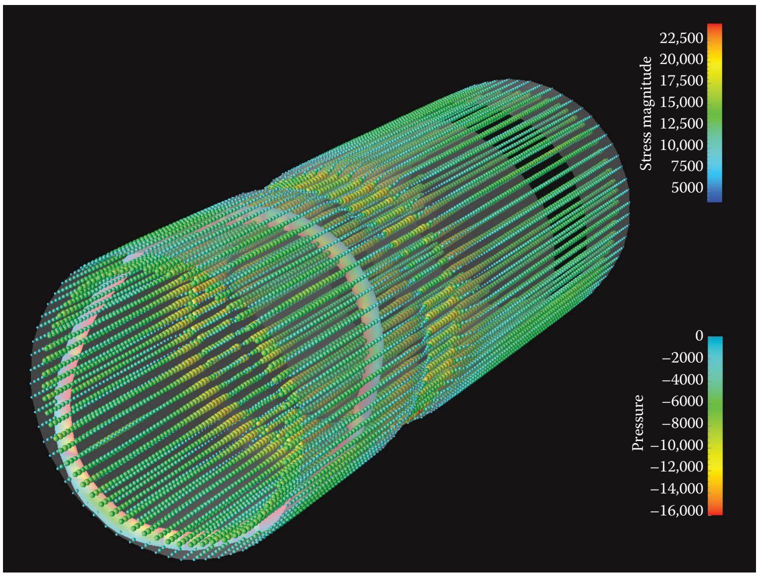

FIGURE 17.11
More complex visualization example of the phantom in Figure 17.10, created with OpenDX. The stress magnitude is represented by glyphs, and it can be seen that the thinner section of the phantom experiences larger stress. The pressure is superimposed in false colors over an oblique ring. The geometry of the object is made visible by overlaying a partly transparent gray surface.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

17.6 Software

17.6.1 Commercial Software Packages

There are a large number of software options available for use in image segmentation, mesh generation, FEA, and visualization applications. Specifically, review of biomechanical FEA literature reveals the following frequently used commercial options:

Image Segmentation/Meshing:

- MIMICS (Materialise, Leuven, Belgium)
- ScanIP+FE (Simpleware Ltd., Exeter, UK)
- 3D-DOCTOR (Able Software Corp., Lexington, Massachusetts)

FEA/Visualization:

- Abaqus (Simulia [Dassault Systèmes], Vélizy-Villacoublay Cedex, France)
- Algor (Autodesk)
- ANSYS (ANSYS, Inc., Canonsburg, Pennsylvania)
- COMSOL (COMSOL, Inc.)

Software companies usually try to offer monolithic solutions. For example, many combined FEM and visualization packages also come with a model builder tool, which allows the assembly of models from relatively simple geometrical elements. In addition, these packages come with several import options to read either surface models or solid models that have been created with other packages. This collection of features makes these packages ideal for the analysis of classical load and stress problems in mechanical engineering. One advantage of the all-in-one approach of monolithic software is the ability of the individual steps to seamlessly interface with each other. Unlike solutions where multiple software modules need to use a common file format for data interchange, a monolithic package handles its data transfer internally and usually invisible to the user.

The creation of finite-element models from biomedical images is more challenging for the monolithic packages. Segmentation and surface parametrization is usually not included, and an external program, such as MIMICS, is needed to perform the first steps in the analysis chain. Although the segmentation and meshing software is aimed at a broad range of applications, it may fail in some special cases, in part because no universal segmentation and meshing approach exists. This shortcoming has led many research groups to develop application-specific modules (e.g., Refs. [69,88–90]). Meshing-related studies often do not address the issue of segmentation. In fact, segmentation is usually considered to be a separate step, although one preprocessing toolkit [21] specializes in combining segmentation and meshing of 3D models.

Analysis of the pertinent literature reveals that a focus of the software development lies on the meshing algorithm, because the transition from a voxel-based image to an analytical description of the object (i.e., the mesh) remains the most critical step in the entire FEM chain.

17.6.2 Image-Based Modeling with Open-Source Software

Frequently, research groups not only describe their methods but also release the actual software source code for use by other interested parties. In fact, a strong community-driven effort to develop and make available free, open-source software has evolved over the

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

last several years, from which interested researchers can benefit immensely. The philosophy of the open-source community not only places an emphasis on free-of-charge availability of such software but more importantly encourages users to share software source code. As a consequence of this openness—and the associated freedoms to inspect and modify the software code—an Internet-connected community continuously works on improving the software and adding new features. Furthermore, the same community is generally available through Internet forums to respond to user questions. Free software, therefore, can be seen as a parallel model to the commercial software model, which is peer based, and which has analogous characteristics of software distribution and user support through the Internet. Since the software is free of license fees, a researcher can download and install software to try without further risk whether the software is suitable for the specific application.

In many cases, wide distribution and continuous improvement have led to the evolution of high-value software. Popular free, open-source software is available not only as program code. Rather, some members of the community create installation-ready software packages that can be conveniently installed without programming experience.* Specifically for medical image processing, segmentation and feature extraction, meshing, FEA, and visualization, there exist several options that make a fully open-source FEM toolchain feasible. Since this field of development currently exhibits a strong momentum, it is covered in detail in Sections 17.6.3 and 17.6.4.

Some examples for free open-source software packages that fulfill some of the functionality laid out in Figure 17.1 are as follows:

Image Processing and Segmentation:

- ImageJ (US National Institutes of Health, Bethesda, Maryland)
- OsiriX [91]
- Crystal Image [27]
- Seg3D (Scientific Computing and Imaging Institute, University of Utah)

Combined Segmentation/Meshing Applications:

- BiMECH [21]
- IA-MESH [92]
- Works presented in Refs. [71,72,93,94]

Meshing:

- BioMesh3D (Scientific Computing and Imaging Institute, University of Utah)
- Cleaver [95]
- Gmsh (http://www.geuz.org/gmsh/)
- MeshLab (http://meshlab.sourceforge.net/)
- Works presented in Refs. [6,69,85,88,96,97]

FEA:

- FeBio [98]
- FreeFEM (http://www.freefem.org/)

* One example is the repositories of the popular *ubuntu* operating system, which contain more than 20,000 installation-ready software packages ranging from office suites to scientific applications, including mesh handling and FEM. The repositories are linked to the operating system in such a fashion that the installation of new software is no more effort than a few mouse clicks.

- Elmer (CSC, Espoo, Finland)
- Tochnog (http://tochnog.sourceforge.net/)

Visualization:

- Paraview (Kitware, Inc., Clifton Park, New York)
- OpenDX (http://www.opendx.org/)

Open-source software for image-based FEM has not yet reached the mainstream of the FEM community, and a careful discussion of the advantages and limitations of a free-software approach is needed. Software modules that find their way into the open-source community are often incorporated to bridge a gap or overcome challenges encountered within the execution of the flow process outlined in Figure 17.1. As such, some open-source packages are limited to special cases. However, frequently those packages are *absorbed* by the wider community and integrated into a larger system of software modules.

Before integrating open-source software into the research process, the following actions can help determine the best overall approach:

- Identify what role(s) the needed software should address.
- Assess capabilities of the software options currently accessible within the research facility. First, site licenses reduce per-user fees for commercial software. Second, available software can be evaluated for suitability for a specific purpose.
- Assess capabilities of commercially available software. If the license fees of a commercial software package are not an obstacle, monolithic software can offer a streamlined solution that promises the least investment of time by the user.
- If an open-source package is being considered, a literature search is helpful to determine if similar solutions have been reported and whether a method is applicable to desired research objectives.

The most frequent contributions to open-source software related to biomechanical FEM analysis are in the form of meshing modules or toolkits that discretize the geometries obtained from segmentation of medical imaging data.* Suitable representations of biomedical objects often lead to excessive numbers of nodes, elements, vertices, and faces that tend to overwhelm the computational specifications of commercial meshing and FE software, which are usually designed to handle CAD-based analytical geometries. As previously mentioned, commercial meshing software is also often somewhat limited in mesh functionalities and capabilities. The other major factor contributing to the recent surge in published mesh generation methods and algorithms can be attributed to the diverse field of biomedical applications, including bone, soft tissue (muscles, organs, blood vessels, implants), the medical image source, and the properties of the mesh (e.g., allowable discretization error, mesh complexity, and the inclusion of material properties). Moreover, reproducibility is a key factor: since open-source software allows the user to examine the actual *inner workings* of the program, a very high level of transparency is inherent in the open-source model. For scientific applications, transparency (and, by association, reproducibility) is of prime concern. Commercial *black-box* software does not provide this advantage: it is well possible that slightly different numerical implementations can lead to

* A comprehensive list can be found at http://www.robertschneiders.de/meshgeneration/software.html, accessed August 24, 2013.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

divergent results between different closed-source software packages, and even between different versions of the same package.

17.6.3 Completely Open Source–Based FEA

Some meshing modules have been developed with the sole aim of being interfaced or used in conjunction with a specific commercial FE software. To our knowledge, there are only two reported instances in which the entire analysis toolchain has been implemented exclusively with open-source software. The first instance examines patient-specific data in order to identify optimal locations for placement of implantable cardiac defibrillators [99,100] and the second study proposes a process to evaluate the biomechanical behavior of autologous tissue-engineered blood vessels [97]. The work presented by Jolley et al. [99,100] features customization through the addition and modification of visualization, mesh refinement, and finite-element calculation parameters within currently available open-source software (notably the software offered by the Scientific Computing and Imaging Institute, University of Utah). Our toolchain [97] revolves around the development of a hexagonal mesh-extraction module capable of adaptive mesh refinement; assignment of material properties, loading, and boundary conditions; and automatic generation of input file data needed to conduct the actual analysis performed with stand-alone FE and visualization software. Both studies compare the results obtained to those provided by commercial counterparts for proof-of-principle validation.

17.6.4 Limitations of Open-Source Software

The vast assortment of currently available open-source software options for image processing, meshing, analysis, and visualization may give rise to the misconception that construction of a FEM toolchain can be done with ease. Fortunately, the continuing adoption of open-source software into scientific computing in general leads to significant streamlining of the data analysis process. Through the organization of community-based libraries, such as the Insight Segmentation and Registration (ITK), Finite Element (FETK), and Visualization (VTK) toolkits, there exists often a central *hub* or headquarters in which developers share, debug, update, modify, and provide cross-platform support for related open-source software. To provide one example, the Scientific Computing and Imaging Institute at the University of Utah (http://www.sci.utah.edu) has provided a comprehensive package of interrelated modules for image-based FEA. Notable elements of this package are Seg3D for volumetric image segmentation, BioMesh3D as a mesh generator for segmented biomedical images, and SciRUN, termed *problem solving environment*, which allows operations with significant manual interaction, such as manual delineation and segmentation. The same research group provides FEBio for solving nonlinear FEA problems specifically in biomedical applications [98], PreView and PostView for pre- and postprocessing, ImageVis3D for volume rendering and visualization, and AtlasWerks for medical image atlas generation.

Not all software is as streamlined as this suite of packages. Frequently, use of open-source software requires some familiarity with computer programming. In some cases, the software is provided in source-code form *only*, and the user has to compile the software (i.e., convert it into an executable application). The compilation process requires that development tools are installed and available. Furthermore, such software often depends on libraries (such as the ITK and VTK libraries mentioned above) that have to be installed separately.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

The main challenge that the creation of a completely open-source FEM toolchain poses is the interfacing between different software applications. Since the need to interface different software programs is a known problem, some file formats are likely to emerge as universal mesh data exchange formats. For example, the STL format (STL stands for *stereo lithography*) is widely used to describe arbitrarily complex surfaces, and it is probably best known as the input format for rapid prototyping. A file format known as STEP (Standard for the Exchange of Product Data) has become an ISO standard exchange format for representing 3D data. IGES (Initial Graphics Exchange Specification) is a NIST (National Institute of Standards and Technology)-supported file format that allows the exchange of information among CAD software. DXF (Drawing Exchange Format) is an alternative file format, created by AutoDesk, to exchange drawing information between CAD programs. STL and DXF files contain geometry, but not material properties or boundary conditions. STEP, which can serve as input for FreeFEM, is a format under development, but the flexible format definition would allow a STEP file to carry all information needed for FEM analysis.

In some instances, special files are needed. For example, Tochnog is a versatile solver for a large number of PDE-based physical problems, including stress/strain, heat transfer, and fluid dynamics (cf. Table 17.1). Tochnog also contains mechanical models for time-dependent, nonlinear materials, which makes it very attractive for modeling of biological tissue. Tochnog uses a nonstandard plaintext input file that contains the spatial coordinates of all mesh vertices, the list of elements and their associated vertices, material groups and the assignment of elements to material groups, boundary conditions, and control instructions. Although Tochnog output is directly readable by Paraview and OpenDX, special software is needed to generate the input file from an existing mesh. It is therefore almost inevitable that Tochnog users need to write a small translation program that reads a mesh in its existing format (e.g., the widely used STL file format) and writes a Tochnog input file.

At this point of software availability, however, the creating of small filter programs to link individual applications is a small price to be paid for the advantage of source code availability. Commercial software is distributed in black-box form, and user support only reaches the extent of the existing software as provided. Given the complexity of biological geometry, the software often becomes overwhelmed and freezes or stops processing. The authors have experienced this effect with two different commercial packages that attempt to create their own volumetric mesh from a given surface mesh, yet fail with complex geometries, such as the volumetric CT image of dentures, or even an inhomogeneous tubular phantom [97]. Because of the black-box nature of commercial software, the failure reason cannot be determined beyond the help available from customer support. Conversely, an open-source package would allow the identification of the point of failure and either remedy the problem (e.g., by increasing the memory limits) or adjust the input file to create a workaround that avoids the point of failure.

17.7 The Open-Source Toolchain—A Case Study

17.7.1 Problem Statement and Underlying Volumetric Image

The skeleton of a rhinoceros (*Diceros bicornis*) was donated to the School of Veterinary Medicine at the University of Georgia. A photo of the skeleton as it is currently on display is shown in Figure 17.12. In the process of preparation, the left hind foot was damaged, and

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

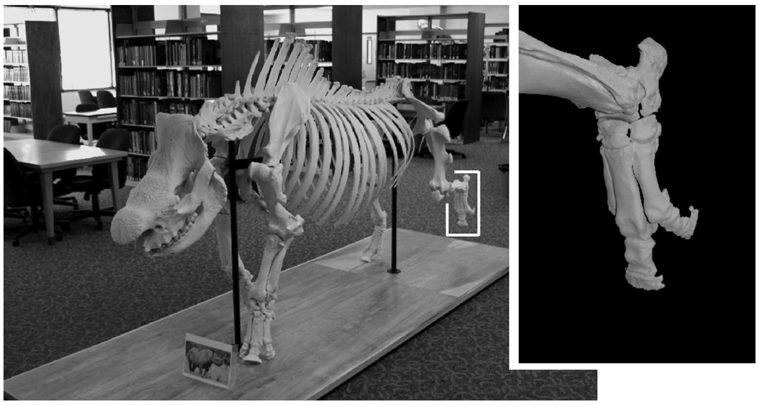

FIGURE 17.12
Photo of the rhinoceros skeleton on display at the University of Georgia School of Veterinary Medicine. The inset shows a close-up photo of the left hind foot.

a repair was attempted by means of inserting metal rods to fixate the bone fragments. A CT scan was performed to verify the placement of these rods. The volumetric CT image was provided to the authors courtesy of Dr. Steven D. Holladay. In this section, we demonstrate the application of the individual steps described in Sections 17.2 through 17.5 to obtain the stress and strain distribution under a defined load.

17.7.2 Segmentation

The CT image (one slice is shown in Figure 17.13a) allows segmentation with purely first-order methods because of the large contrast between bone and the surrounding air and the similarly large contrast between metal and bone. A region-growing process yields the

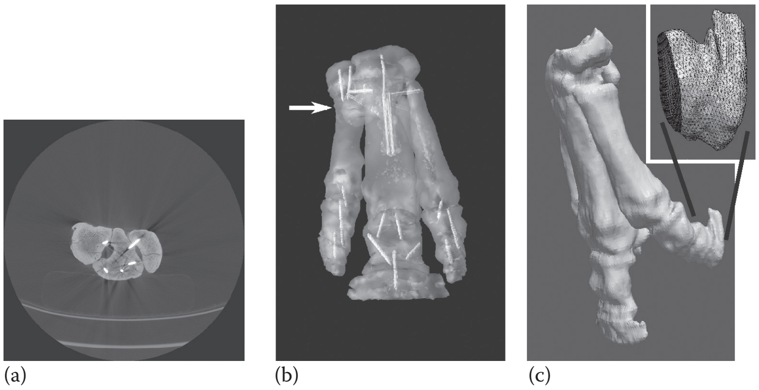

FIGURE 17.13
CT imaging results of the rhinoceros foot. (a) Raw cross-sectional CT image; one slice is shown at about the height indicated by a white arrow in (b). Bright white areas indicate metal. Below the bone, the slightly curved patient tray can be seen. The image shows dark streak artifacts that are partly attributed to beam hardening. (b) Projection view of the segmented foot where the metal regions are shown in white and the bone has been assigned a transparent color to make the embedded metal rods visible. (c) Projection view of the meshed volume, rotated to match the observer position of the inset in Figure 17.12. The inset shows a magnified section with the mesh edges highlighted.

bone area and eliminates the patient tray and the foam cushion on which the foot rested during imaging. Morphological operations were used to fill small discontinuities caused by the streak artifacts. In a second step, purely intensity-based thresholding was used to separate bone and metal. We used Crystal Image to perform these steps. A rendered view of the segmented foot, created with OpenDX, is shown in Figure 17.13b.

17.7.3 Meshing

Meshing was performed with Biomesh3D and Cleaver. A rendered image of the mesh, visualized with Paraview, can be seen in Figure 17.13c. In preparation for meshing, the volumetric image was rescaled to one-third of its original size, because Cleaver does not offer any mesh scaling option: the number of voxels in the image directly determines the number of nodes in the mesh. The bone region in the unscaled image (voxel size, $0.5 \times 0.5 \times 1$ mm) occupies 3.5 million voxels. The number of bone voxels in the scaled image was approximately 132,000. The volume mesh produced by Cleaver was composed of 1,045,415 tetrahedral elements and 216,627 vertices. With this number of elements, a simulation run with Tochnog requires almost 16 GB of memory, and the available memory is the main limiting factor for initial mesh refinement. For example, scaling the image by 1/2 instead of 1/3 increases the number of nodes by a factor of 3.5, and memory requirement increases from 16 to 64 GB. These numbers again highlight the key role of the meshing software in image-based FEM. Objects with known analytical geometry can be meshed with much fewer elements, because the optimum model, notably with *p*-refinement, is known. Conversely, in image-based FEM, the optimum model is unknown, and the meshing software needs to decide to what extent *p*- and *h*-refinement can be used. In the simplest case (i.e., Cleaver), no refinement takes place. Biomesh3D allows some control over the level of detail in the mesh, but a Biomesh3D run can take several days to complete.

17.7.4 Simulation

Consistent with the limitations discussed in Section 17.6.4, we wrote a simple script that translated the Cleaver output file into a Tochnog input file. The script was also responsible for scaling the vertex coordinates by the voxel size and supplementing the loading and boundary conditions. In this example, we simulated a situation where the isolated foot rested on a surface (simulated by fixation of the nodes in space), and where a force was applied to the middle toe.

Given its skeletal form, the rhino foot was modeled as an elastic, isotropic, nonlinear solid composed of cortical bone and the titanium rods used to join and fixate the bone fragments. The strain energy (TPE) method for material deformation containing both distortional (deviatoric) and volumetric components of the deformation gradient is defined in Tochnog by

$$\rho v_i = \frac{\partial \sigma_{ij}}{\partial x_{ij}} + (1-\beta T)\rho g_i - d\frac{\partial v_i}{\partial x_i} + \mathrm{f}_i, \tag{17.29}$$

where

ρ is the density of the material,
v_i is the material velocity in the i direction,
σ_{ij} is the material stress matrix,
x is the space coordinate,

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

β is the material expansion volume, or measurement of the volume change of a solid in response to a change in pressure or stress,
T is the temperature of the material (assumed to be constant in our case),
g_i is gravity,
d is the damping coefficient of material (negligible in our case), and
f_i is the pressure tensor source (in our case, a force is applied to the middle toe).

17.7.4.1 Boundary and Loading Conditions

The following conditions were assigned to the meshed model (Figure 17.14a):

Boundary:

- Nodes corresponding to lower portions of each outer toe were fixed in the x, y, and z directions to mimic their attachment to a surface.

Loading:

- A constant load of 9.8 N (≈1 kg or 2.2 lb) was applied to the nodes near the lower portion of the middle toe. In order for the force to be evenly distributed, each of the 19,777 nodes in the region was assigned a force of -4.9×10^{-4} N. This force was applied in the y direction and assigned a negative value to simulate a load directed downward onto the toe based on the rhino foot's orientation.

17.7.4.2 Material Properties

The material property input parameters and units for both materials are provided in Table 17.2.

17.7.5 Visualization

The results of the FE simulation are presented in Figure 17.14. The behavior exhibited in the rhino foot subjected to an applied load in Figure 17.14a is similar to a diver on a diving board. The board is fixated at the edge of a pool in the manner the foot is secured to the tray. The middle toe and the free edge of the diving board are comparable. As the diver walks to the end of the platform, the board is likely under minimal amounts of stress. However, once the diver jumps and lands back on the diving board in preparation for the dive, the largest stresses are observed at the point of impact, resulting in deformation of the board (strain) to accommodate the intensity of the stress (Figure 17.14b and c). This partially explains why the observed changes in stress and strain behavior are restricted to the boundary between areas of the middle toe subjected to the load and areas that are not.

In the case of the rhino foot, the bone fragment's fixation by the titanium rods near the edge of the middle toe also contributes to the stress and strain behavior observed. Additionally, the stress and strain values reported correlate with the known mechanical properties of bone and metal. Such materials exhibit high strength as reflected by their large elastic modulus values and are able to withstand large amounts of stress (in this case, 10^9 Pa) with only minimal deformation or strain (10^{-4}). Movement throughout the middle toe in response to the boundary conditions, applied load, and mechanical properties of the materials can be seen in Figure 17.14d. Although the values are relatively small $\left(10^{-4}\,\frac{\text{m}}{\text{s}}\right)$,

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

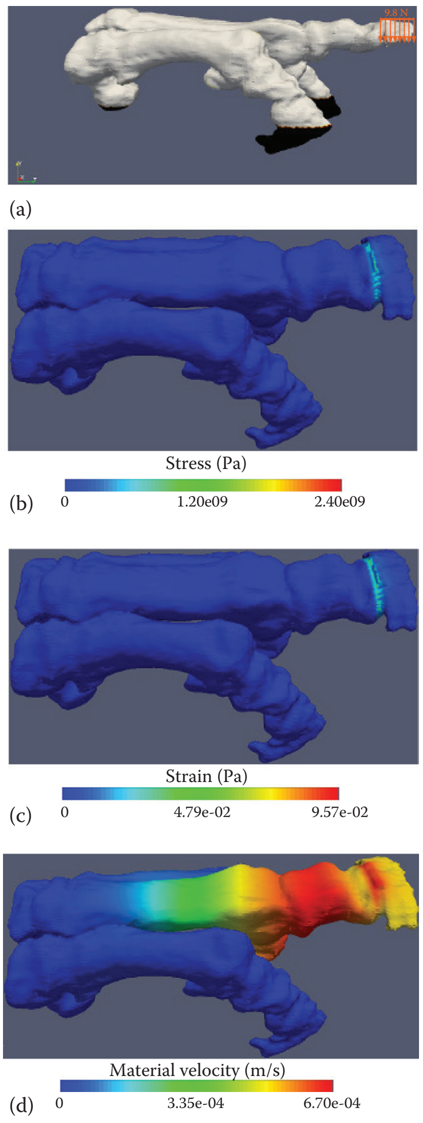

FIGURE 17.14
Visualization results of the rhino foot simulation. Boundary and loading conditions applied to the rhino foot mesh (a). All nodes of each outer toe's lower portion (black) are fixed in the *x*, *y*, and *z* directions. The intermediate colors represent the boundary between fixated and nonfixated nodes. A load of 9.8 N was evenly distributed over the 19,777 nodes near the end of the lower toe (orange arrows). Stress and stain distribution behavior are nearly identical and higher levels are restricted to a thin region where the bone fragment near the end of the middle toe is held in place by a titanium rod (b and c). Although the model is subjected to large amounts of stress, the strain or deformation values observed are very small in comparison. The material velocity is representative of the movement of the center toe in response to the applied load and boundary conditions (d).

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

TABLE 17.2

Material Property Parameters Used in FE Simulation

Property (Units)	Cortical Bone	Titanium Rods	Reference
Density, ρ $\frac{\text{kg}}{\text{m}^3}$	3.315×10^5	4.430×10^3	[101,102]
Poisson ratio	3.00×10^{-1}	3.42×10^{-1}	[102,103]
Elastic modulus, *E* (GPa)	18.6	113.8	[102,104]

the most movement is observed in the areas of applied force and where there is a connection of bone fragments via the titanium rods. In Figure 17.14, the effect is most prominent in the material deformation, which is represented as velocity (Figure 17.14d), that is, the spatial change of the nodes over one simulated time step Δt. Once again, the deformation is consistent with the diving-board analogy.

A side benefit of the meshing process is also illustrated in Figure 17.12. At some point, it was decided that the damaged bones of the rhinoceros foot were beyond repair and at risk of coming apart in the lively environment of the library where the skeleton is exhibited. We decided to use the meshed model (Figure 17.13c) to extract the surface and used a 3D printer to create an exact representation of the foot. The left hind foot shown in Figure 17.12 is actually the 3D-printed construct, and only very astute observers actually recognize the difference.

17.8 Future Perspective

The need for robust, completely automatic methods for image segmentation and mesh generation of geometric models obtained from medical imaging data continues to drive the research within this realm. As the capabilities of medical imaging coupled with FEA continue to expand past engineering and computational mathematics-based laboratories into hospitals and research groups composed of medical professionals, engineers, and statisticians, techniques that are accurate and reproducible, require minimal user intervention, and can generate models within a short time frame are in high demand. The evolving focus areas of computer-aided diagnosis and computer-aided visualization and analysis (CAVA) have catalyzed biomedical FEA applications and thereby are largely responsible for the formulation of such multidisciplinary teams. Computer-aided diagnosis is applied in the detection and diagnosis of disease, lesions, and abnormalities while CAVA involves the study and development of computerized methods, such as the ones discussed in this chapter, to catalyze new strategies, education, and training [29,105–107]. To provide one example, the biomechanical response behavior obtained from FEA could assist in clinical research trials to measure changes in condition attributed to drug and radiation therapies.

The creation of registration, template, and atlas-based databases; rapid prototyping; and computational fluid dynamics are a few CAVA-related developments that can further facilitate the growth of computer-aided diagnosis. Integration of a novel meshing algorithm that *morphs* itself around a presegmented medical image based on a template or atlas of the desired geometry provides a convenient way to evaluate clinical data for multiple patients at varying stages of disease and treatment [108–113]. Rapid prototyping allows the creation of physical models manufactured from 3D data obtained from medical images and is ideal for anatomical instruction/education, presurgical planning, and the design, verification, and manufacturing of medical implants and prosthetics [114–116]. Lastly, the adaptation of computational fluid dynamics, a numerical simulation technique designed for the analysis of fluids and of biphasic solid and fluidic models, to accommodate nonlinear and complex flow patterns provides real-time simulations of blood flow and cerebrospinal fluid through ventricles in the vascular and neurological cavities [117,118].

References

1. R. Courant, Variational methods for the solution of problems of equilibrium and vibrations, *Bulletin of the American Mathematical Society* 49 (1) (1943) 1–23.
2. D. J. Hawkes, D. Barratt, J. M. Blackall, C. Chan, P. J. Edwards, K. Rhode, G. P. Penney, J. McClelland, D. L. G. Hill, Tissue deformation and shape models in image-guided interventions: A discussion paper, *Medical Image Analysis* 9 (2) (2005) 163–175.
3. T. Hopp, M. Dietzel, P. Baltzer, P. Kreisel, W. Kaiser, H. Gemmeke, N. Ruiter, Automatic multimodal 2D/3D breast image registration using biomechanical FEM models and intensity-based optimization, *Medical Image Analysis* 17 (2) (2013) 209–218.
4. K. Miller, A. Wittek, G. Joldes, A. Horton, T. Dutta-Roy, J. Berger, L. Morriss, Modelling brain deformations for computer-integrated neurosurgery, *International Journal for Numerical Methods in Biomedical Engineering* 26 (1) (2010) 117–138.
5. T. A. Sundaram, J. C. Gee, Towards a model of lung biomechanics: Pulmonary kinematics via registration of serial lung images, *Medical Image Analysis* 9 (6) (2005) 524–537.
6. J. Teo, C. Chui, Z. Wang, S. Ong, C. Yan, S. Wang, H. Wong, S. Teoh, Heterogeneous meshing and biomechanical modeling of human spine, *Medical Engineering & Physics* 29 (2) (2007) 277–290.
7. W. Parr, U. Chamoli, A. Jones, W. Walsh, S. Wroe, Finite element micro-modelling of a human ankle bone reveals the importance of the trabecular network to mechanical performance: New methods for the generation and comparison of 3D models, *Journal of Biomechanics* 46 (1) (2013) 200–205.
8. M. Mononen, M. Mikkola, P. Julkunen, R. Ojala, M. Nieminen, J. Jurvelin, R. Korhonen, Effect of superficial collagen patterns and fibrillation of femoral articular cartilage on knee joint mechanics—A 3d finite element analysis, *Journal of Biomechanics* 45 (3) (2012) 579–587.
9. M. Bajuri, M. R. A. Kadir, M. M. Raman, T. Kamarul, Mechanical and functional assessment of the wrist affected by rheumatoid arthritis: A finite element analysis, *Medical Engineering & Physics* 34 (9) (2012) 1294–1302.
10. C. Lally, F. Dolan, P. J. Prendergast, Cardiovascular stent design and vessel stresses: A finite element analysis, *Journal of Biomechanics* 38 (8) (2005) 1574–1581.
11. P. J. Prendergast, C. Lally, S. Daly, A. J. Reid, T. C. Lee, D. Quinn, F. Dolan, Analysis of prolapse in cardiovascular stents: A constitutive equation for vascular tissue and finite-element modelling, *Journal of Biomechanical Engineering* 125 (5) (2003) 692–699.
12. P. Mortier, G. Holzapfel, M. De Beule, D. Van Loo, Y. Taeymans, P. Segers, P. Verdonck, B. Verhegghe, A novel simulation strategy for stent insertion and deployment in curved coronary bifurcations: Comparison of three drug-eluting stents, *Annals of Biomedical Engineering* 38 (1) (2010) 88–99.
13. H. Huang, J. Hsu, L. Fuh, M. Tu, C. Ko, Y. Shen, Bone stress and interfacial sliding analysis of implant designs on an immediately loaded maxillary implant: A non-linear finite element study, *Journal of Dentistry* 36 (6) (2008) 409–417.
14. M. Ghajari, S. Peldschus, U. Galvanetto, L. Iannucci, Effects of the presence of the body in helmet oblique impacts, *Accident Analysis and Prevention* 50 (2013) 263–271.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

15. A. Erdemir, T. M. Guess, J. Halloran, S. C. Tadepalli, T. M. Morrison, Considerations for reporting finite element analysis studies in biomechanics, *Journal of Biomechanics* 45 (4) (2012) 625–633.
16. M. A. Haidekker, *Medical Imaging Technology, SpringerBriefs Series in Physics*, Springer, New York, 2013.
17. L. Allard, G. Cloutier, L. Durand, 3D power Doppler ultrasound imaging of an in vitro arterial stenosis, *Acoustical Imaging* 23 (1997) 267–272.
18. Z. Guo, A. Fenster, Three-dimensional power doppler imaging: A phantom study to quantify vessel stenosis, *Ultrasound in Medicine and Biology* 22 (8) (1996) 1059–1069.
19. Z. Guo, L.-G. Durand, L. Allard, G. Cloutier, A. Fenster, In vitro evaluation of multiple arterial stenoses using three-dimensional power doppler angiography, *Journal of Vascular Surgery* 27 (4) (1998) 681–688.
20. G. Cloutier, Z. Qin, D. Garcia, G. Soulez, V. Oliva, L.-G. Durand, Assessment of arterial stenosis in a flow model with power Doppler angiography: Accuracy and observations on blood echogenicity, *Ultrasound in Medicine and Biology* 26 (9) (2000) 1489–1501.
21. C.-K. Chui, Z. Wang, J. Zhang, J. S.-K. Ong, L. Bian, J. C.-M. Teo, C.-H. Yan, S.-H. Ong, S.-C. Wang, H.-K. Wong, S.-H. Teoh, A component-oriented software toolkit for patient-specific finite element model generation, *Advances in Engineering Software* 40 (3) (2009) 184–192.
22. R. Andresen, H. J. Werner, H. C. Schober, Contribution of the cortical shell of vertebrae to mechanical behaviour of the lumbar vertebrae with implications for predicting fracture risk, *British Journal of Radiology* 71 (847) (1998) 759–765.
23. C.-S. Kuo, H.-T. Hu, R.-M. Lin, K.-Y. Huang, P.-C. Lin, Z.-C. Zhong, M.-L. Hseih, Biomechanical analysis of the lumbar spine on facet joint force and intradiscal pressure—A finite element study, *BMC Musculoskeletal Disorders* 11 (1) (2010) 1–13.
24. D. Withey, Z. Koles, Medical image segmentation: Methods and software, in: *Noninvasive Functional Source Imaging of the Brain and Heart and the International Conference on Functional Biomedical Imaging, 2007. NFSI-ICFBI 2007. Joint Meeting of the 6th International Symposium on*, IEEE, 2007, pp. 140–143.
25. M. Gao, J. Huang, X. Huang, S. Zhang, D. Metaxas, Simplified labeling process for medical image segmentation, in: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2012*, 2012, pp. 387–394.
26. G. Dougherty, *Digital Image Processing for Medical Applications*, Vol. 1, Cambridge University Press, Cambridge, United Kingdom, 2009.
27. M. Haidekker, *Advanced Biomedical Image Analysis*, John Wiley & Sons, Hoboken, NJ, 2011.
28. K. Ciesielski, J. Udupa, A framework for comparing different image segmentation methods and its use in studying equivalences between level set and fuzzy connectedness frameworks, *Computer Vision and Image Understanding* 115 (6) (2011) 721–734.
29. X. Chen, J. K. Udupa, A. Alavi, D. A. Torigian, GC-ASM: Synergistic integration of graph-cut and active shape model strategies for medical image segmentation, *Computer Vision and Image Understanding* 117 (5) (2013) 513–524.
30. M. Kass, A. Witkin, D. Terzopoulos, Snakes: Active contour models, *International Journal of Computer Vision* 1 (4) (1988) 321–331.
31. R. Malladi, J. Sethian, B. Vemuri, Shape modeling with front propagation: A level set approach, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 17 (2) (1995) 158–175.
32. S. Osher, R. P. Fedkiw, Level set methods: An overview and some recent results, *Journal of Computational Physics* 169 (2) (2001) 463–502.
33. A. X. Falcão, J. K. Udupa, S. Samarasekera, S. Sharma, B. E. Hirsch, R. D. A. Lotufo, User-steered image segmentation paradigms: Live wire and live lane, *Graphical Models and Image Processing* 60 (4) (1998) 233–260.
34. S. Lloyd, Least squares quantization in PCM, *Information Theory, IEEE Transactions on* 28 (2) (1982) 129–137.
35. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Kluwer Academic Publishers, Norwell, MA, 1981.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

36. Y. Boykov, G. Funka-Lea, Graph cuts and efficient ND image segmentation, *International Journal of Computer Vision* 70 (2) (2006) 109–131.
37. L. Vincent, P. Soille, Watersheds in digital spaces: An efficient algorithm based on immersion simulations, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13 (6) (1991) 583–598.
38. J. K. Udupa, S. Samarasekera, Fuzzy connectedness and object definition: Theory, algorithms, and applications in image segmentation, *Graphical Models and Image Processing* 58 (3) (1996) 246–261.
39. A. K. Jain, R. P. W. Duin, J. Mao, Statistical pattern recognition: A review, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 22 (1) (2000) 4–37.
40. K. I. Laws, Texture energy measures, in: *Proc. DARPA Image Understanding Workshop*, 1979, pp. 47–51.
41. R. K. Yip, P. K. Tam, D. N. Leung, Modification of Hough transform for circles and ellipses detection using a 2-dimensional array, *Pattern Recognition* 25 (9) (1992) 1007–1022.
42. T. Cootes, C. Taylor, D. Cooper, J. Graham, Active shape models—Their training and application, *Computer Vision and Image Understanding* 61 (1) (1995) 38–59.
43. T. Cootes, G. Edwards, C. Taylor, Active appearance models, in: *Computer VisionECCV98*, 1998, pp. 484–498.
44. T. Heimann, H.-P. Meinzer, Statistical shape models for 3D medical image segmentation: A review, *Medical Image Analysis* 13 (4) (2009) 543–563.
45. D. Lesage, E. D. Angelini, I. Bloch, G. Funka-Lea, A review of 3D vessel lumen segmentation techniques: Models, features and extraction schemes, *Medical Image Analysis* 13 (6) (2009) 819–845.
46. S. Ghose, A. Oliver, R. Martí, X. Lladó, J. C. Vilanova, J. Freixenet, J. Mitra, D. Sidibé, F. Meriaudeau, A survey of prostate segmentation methodologies in ultrasound, magnetic resonance and computed tomography images, *Computer Methods and Programs in Biomedicine* 108 (1) (2012) 262–287.
47. Z. Dokur, T. Ölmez, Segmentation of ultrasound images by using a hybrid neural network, *Pattern Recognition Letters* 23 (14) (2002) 1825–1836.
48. C. Petitjean, J.-N. Dacher, A review of segmentation methods in short axis cardiac MR images, *Medical Image Analysis* 15 (2) (2011) 169–184.
49. M. M. Fraz, P. Remagnino, A. Hoppe, B. Uyyanonvara, A. R. Rudnicka, C. G. Owen, S. A. Barman, Blood vessel segmentation methodologies in retinal images—A survey, *Computer Methods and Programs in Biomedicine* 108 (1) (2012) 407–433.
50. L. Szilágyi, S. M. Szilágyi, B. Benyó, Z. Benyó, Intensity inhomogeneity compensation and segmentation of mr brain images using hybrid c-means clustering models, *Biomedical Signal Processing and Control* 6 (1) (2011) 3–12.
51. A. P. Dhawan, S. Juvvadi, Knowledge-based analysis and understanding of medical images, *Computer Methods and Programs in Biomedicine* 33 (4) (1990) 221–239.
52. F. Masulli, A. Schenone, A fuzzy clustering based segmentation system as support to diagnosis in medical imaging, *Artificial Intelligence in Medicine* 16 (2) (1999) 129–147.
53. M. Clark, L. Hall, D. Goldgof, R. Velthuizen, F. Murtagh, M. Silbiger, Automatic tumor segmentation using knowledge-based techniques, *Medical Imaging, IEEE Transactions on* 17 (2) (1998) 187–201.
54. D. Zhang, S. Chen, A novel kernelized fuzzy c-means algorithm with application in medical image segmentation, *Artificial Intelligence in Medicine* 32 (1) (2004) 37–50.
55. A. Papadopoulos, D. Fotiadis, A. Likas, An automatic microcalcification detection system based on a hybrid neural network classifier, *Artificial Intelligence in Medicine* 25 (2) (2002) 149–167.
56. F. Xie, A. C. Bovik, Automatic segmentation of dermoscopy images using self-generating neural networks seeded by genetic algorithm, *Pattern Recognition* 46 (3) (2013) 1012–1019.
57. S. Gibson, Constrained elastic surface nets: Generating smooth surfaces from binary segmented data, in: *Medical Image Computing and Computer-Assisted Intervention MICCAI98*, 1998, pp. 888–898.
58. W. Lorensen, H. Cline, Marching cubes: A high resolution 3D surface construction algorithm, *Computer Graphics* 21 (4) (1987) 163–169.

59. I. Takanashi, S. Muraki, A. Doi, A. Kaufman, 3D active net for volume extraction, in: *Proc. SPIE*, Vol. 3298, 1998, pp. 184–193.
60. M. A. Yerry, M. S. Shephard, Automatic three-dimensional mesh generation by the modified-octree technique, *International Journal for Numerical Methods in Engineering* 20 (11) (1984) 1965–1990.
61. M. S. Shephard, M. K. Georges, Automatic three-dimensional mesh generation by the finite octree technique, *International Journal for Numerical Methods in Engineering* 32 (4) (1991) 709–749.
62. S. Lo, Volume discretization into tetrahedra—I. Verification and orientation of boundary surfaces, *Computers & Structures* 39 (5) (1991) 493–500.
63. S. Lo, Volume discretization into tetrahedra—II. 3d triangulation by advancing front approach, *Computers & Structures* 39 (5) (1991) 501–511.
64. B. Delaunay, Sur la sphère vide, *Izvestia Akademii Nauk SSSR: Otdelenie Matematicheskikh i Estestvennykh Nauk* 7 (1934) 793–800.
65. T. D. Blacker, R. J. Meyers, Seams and wedges in plastering: A 3-d hexahedral mesh generation algorithm, *Engineering with Computers* 9 (2) (1993) 83–93.
66. T. Li, R. McKeag, C. Armstrong, Hexahedral meshing using midpoint subdivision and integer programming, *Computer Methods in Applied Mechanics and Engineering* 124 (1) (1995) 171–193.
67. R. Schneiders, A grid-based algorithm for the generation of hexahedral element meshes, *Engineering with Computers* 12 (3) (1996) 168–177.
68. T. J. Tautges, T. Blacker, S. A. Mitchell, The whisker weaving algorithm: A connectivity-based method for constructing all-hexahedral finite element meshes, *International Journal for Numerical Methods in Engineering* 39 (19) (1996) 3327–3349.
69. Z. Yu, M. J. Holst, J. Andrew McCammon, High-fidelity geometric modeling for biomedical applications, *Finite Elements in Analysis and Design* 44 (11) (2008) 715–723.
70. S. Azernikov, A. Miropolsky, A. Fischer, Surface reconstruction of freeform objects based on multiresolution volumetric method, in: *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications*, ACM, 2003, pp. 115–126.
71. Y. Zhang, C. Bajaj, B.-S. Sohn, 3d finite element meshing from imaging data, *Computer Methods in Applied Mechanics and Engineering* 194 (48) (2005) 5083–5106.
72. N. Ribeiro, P. Fernandes, D. Lopes, J. Folgado, P. Fernandes, 3-d solid and finite element modeling of biomechanical structures-a software pipeline, in: J. Ambròsio (Ed.), *Proceedings of the 7th EUROMECH Solid Mechanics Conference*, Lisbon, Portugal, 2009.
73. G. B. Dantzig, L. R. Ford, D. R. Fulkerson, A primal-dual algorithm for linear programs, *Linear Inequalities and Related Systems* (38) (1956) 171–181.
74. L. A. Freitag, C. Ollivier-Gooch, Tetrahedral mesh improvement using swapping and smoothing, *International Journal for Numerical Methods in Engineering* 40 (21) (1997) 3979–4002.
75. T. Ju, F. Losasso, S. Schaefer, J. Warren, Dual contouring of hermite data, *ACM Transactions on Graphics (TOG)* 21 (3) (2002) 339–346.
76. S. Schaefer, J. Warren, Dual marching cubes: Primal contouring of dual grids, *Computer Graphics Forum* 24 (2005) 195–201.
77. S. Schaefer, T. Ju, J. Warren, Manifold dual contouring, *Visualization and Computer Graphics, IEEE Transactions on* 13 (3) (2007) 610–619.
78. L. R. Herrmann, Laplacian-isoparametric grid generation scheme, *Journal of the Engineering Mechanics Division* 102 (5) (1976) 749–907.
79. L. Freitag, M. Jones, P. Plassmann, A parallel algorithm for mesh smoothing, *SIAM Journal on Scientific Computing* 20 (6) (1999) 2023–2040.
80. M. Puso, J. Solberg, A stabilized nodally integrated tetrahedral, *International Journal for Numerical Methods in Engineering* 67 (6) (2006) 841–867.
81. D. L. Camacho, R. H. Hopper, G. M. Lin, B. S. Myers, An improved method for finite element mesh generation of geometrically complex structures with application to the skullbase, *Journal of Biomechanics* 30 (10) (1997) 1067–1070.
82. A. Cifuentes, A. Kalbag, A performance study of tetrahedral and hexahedral elements in 3-D finite element structural analysis, *Finite Elements in Analysis and Design* 12 (3) (1992) 313–318.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

83. V. I. Weingarten, The controversy over hex or tet meshing, *Machine Design* 66 (8) (1994) 74–76.
84. A. Ramos, J. Simoes, Tetrahedral versus hexahedral finite elements in numerical modelling of the proximal femur, *Medical Engineering & Physics* 28 (9) (2006) 916–924.
85. K. H. Shivanna, S. C. Tadepalli, N. M. Grosland, Feature-based multiblock finite element mesh generation, *Computer-Aided Design* 42 (12) (2010) 1108–1116.
86. Z. Wang, J. Teo, C. Chui, S. Ong, C. Yan, S. Wang, H. Wong, S. Teoh, Computational biomechanical modelling of the lumbar spine using marching-cubes surface smoothened finite element voxel meshing, *Computer Methods and Programs in Biomedicine* 80 (1) (2005) 25–35.
87. D. Thompson, J. Braun, R. Ford, *OpenDX—Paths to Visualization*, Visualization and Imagery Solutions, Inc., Missoula, MT, 2001.
88. G.-H. Kwon, S.-W. Chae, K.-J. Lee, Automatic generation of tetrahedral meshes from medical images, *Computers and Structures* 81 (2003) 765–775.
89. W. Lee, T.-S. Kim, M. Cho, Y. Ahn, S. Lee, Methods and evaluations of MRI content-adaptive finite element mesh generation for bioelectromagnetic problems, *Physics in Medicine and Biology* 51 (2006) 6173–6186.
90. P. Perré, Meshpore: A software able to apply image-based meshing techniques to anisotropic and heterogeneous porous media, *Drying Technology* 23 (2005) 1993–2006.
91. A. Rosset, L. Spadola, O. Ratib, Osirix: An open-source software for navigating in multidimensional dicom images, *Journal of Digital Imaging* 17 (3) (2004) 205–216.
92. N. M. Grosland, K. H. Shivanna, V. A. Magnotta, N. A. Kallemeyn, N. A. DeVries, S. C. Tadepalli, C. Lisle, IA-FEMesh: An open-source, interactive, multiblock approach to anatomic finite element model development, *Computer Methods and Programs in Biomedicine* 94 (1) (2009) 96–107.
93. L. Antiga, M. Piccinelli, L. Botti, B. Ene-Iordache, A. Remuzzi, D. A. Steinman, An image-based modeling framework for patient-specific computational hemodynamics, *Medical and Biological Engineering and Computing* 46 (11) (2008) 1097–1112.
94. Q. Fang, D. A. Boas, Tetrahedral mesh generation from volumetric binary and gray-scale images, in: *Proceedings of the Sixth IEEE International Conference on Symposium on Biomedical Imaging: From Nano to Macro*, IEEE Press, 2009, pp. 1142–1145.
95. J. R. Bronson, J. A. Levine, R. T. Whitaker, Lattice cleaving: Conforming tetrahedral meshes of multimaterial domains with bounded quality, in: *Proceedings of the 21st International Meshing Roundtable*, Springer, 2013, pp. 191–209.
96. S. K. Boyd, R. Müller, Smooth surface meshing for automated finite element model generation from 3D image data, *Journal of Biomechanics* 39 (7) (2006) 1287–1295.
97. A. Madison, M. A. Haidekker, A completely open-source finite element modeling chain for tubular tissue-engineered constructs, *International Journal of Computer Science and Application (IJCSA)* 1 (2) (2012) 44–55.
98. S. A. Maas, B. J. Ellis, G. A. Ateshian, J. A. Weiss, FEBio: Finite elements for biomechanics, *Journal of Biomechanical Engineering* 134 (1) (2012) 011005-01–011005–10.
99. M. Jolley et al. Open-source environment for interactive finite element modeling of optimal ICD electrode placement. in: *Functional Imaging and Modeling of the Heart*, Springer, Berlin, Heidelberg, 2007, pp. 373–382.
100. M. Jolley, J. Stinstra, S. Pieper, R. MacLeod, D. H. Brooks, F. Cecchin, J. K. Triedman, A computer modeling tool for comparing novel ICD electrode orientations in children and adults, *Heart Rhythm* 5 (4) (2008) 565–572.
101. R. Andresen, M. Haidekker, S. Radmer, D. Banzer, CT determination of bone mineral density and structural investigations on the axial skeleton for estimating the osteoporosis-related fracture risk by means of a risk score, *British Journal of Radiology* 72 (858) (1999) 569–578.
102. R. F. Boyer, E. Collings, *Materials Properties Handbook: Titanium Alloys*, ASM International, Materials Park, OH, 1994.
103. D. C. Wirtz, N. Schiffers, T. Pandorf, K. Radermacher, D. Weichert, R. Forst, Critical evaluation of known bone material properties to realize anisotropic FE-simulation of the proximal femur, *Journal of Biomechanics* 33 (10) (2000) 1325–1330.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015

104. M. Cuppone, B. Seedhom, E. Berry, A. Ostell, The longitudinal Youngs modulus of cortical bone in the midshaft of human femur and its correlation with CT scanning data, *Calcified Tissue International* 74 (3) (2004) 302–309.
105. J. K. Udupa, V. R. LeBlanc, Y. Zhuge, C. Imielinska, H. Schmidt, L. M. Currie, B. E. Hirsch, J. Woodburn, A framework for evaluating image segmentation algorithms, *Computerized Medical Imaging and Graphics* 30 (2) (2006) 75–87.
106. K. Doi, Computer-aided diagnosis in medical imaging: Historical review, current status and future potential, *Computerized Medical Imaging and Graphics* 31 (4–5) (2007) 198–211.
107. H. Kobatake, Future cad in multi-dimensional medical images: Project on multi-organ, multi-disease cad system, *Computerized Medical Imaging and Graphics* 31 (4) (2007) 258–266.
108. L. Baghdadi, D. A. Steinman, H. M. Ladak, Template-based finite-element mesh generation from medical images, *Computer Methods and Programs in Biomedicine* 77 (1) (2005) 11–21.
109. I. A. Sigal, M. R. Hardisty, C. M. Whyne, Mesh-morphing algorithms for specimen-specific finite element modeling, *Journal of Biomechanics* 41 (7) (2008) 1381–1389.
110. M. A. Baldwin, J. E. Langenderfer, P. J. Rullkoetter, P. J. Laz, Development of subject-specific and statistical shape models of the knee using an efficient segmentation and mesh-morphing approach, *Computer Methods and Programs in Biomedicine* 97 (3) (2010) 232–240.
111. M. Bucki, C. Lobos, Y. Payan, A fast and robust patient specific finite element mesh registration technique: Application to 60 clinical cases, *Medical Image Analysis* 14 (3) (2010) 303–317.
112. G. D. Santis, M. D. Beule, K. V. Canneyt, P. Segers, P. Verdonck, B. Verhegghe, Full-hexahedral structured meshing for image-based computational vascular modeling, *Medical Engineering & Physics* 33 (10) (2011) 1318–1325.
113. C. Lederman, A. Joshi, I. Dinov, L. Vese, A. Toga, J. D. V. Horn, The generation of tetrahedral mesh models for neuroanatomical MRI, *NeuroImage* 55 (1) (2011) 153–164.
114. R. Petzold, H.-F. Zeilhofer, W. Kalender, Rapid prototyping technology in medicinebasics and applications, *Computerized Medical Imaging and Graphics* 23 (5) (1999) 277–284.
115. R. Bibb, J. Winder, A review of the issues surrounding three-dimensional computed tomography for medical modelling using rapid prototyping techniques, *Radiography* 16 (1) (2010) 78–83.
116. T. Boehler, D. van Straaten, S. Wirtz, H.-O. Peitgen, A robust and extendible framework for medical image registration focused on rapid clinical application deployment, *Computers in Biology and Medicine* 41 (6) (2011) 340–349.
117. K. B. Chandran, S. C. Vigmostad, Patient-specific bicuspid valve dynamics: Overview of methods and challenges, *Journal of Biomechanics* 46 (2) (2013) 208–216.
118. V. Kurtcuoglu, D. Poulikakos, Y. Ventikos, Computational modeling of the mechanical behavior of the cerebrospinal fluid system, *Transactions of the ASME-K-Journal of Biomechanical Engineering* 127 (2) (2005) 264–269.

Downloaded by [Adrienne Madison] at 16:34 24 April 2015