COMPUTER METHODS AND PROGRAMS IN BIOMEDICINE 94 (2009) 96–107

journal homepage: www.intl.elsevierhealth.com/journals/cmpb

IA-FEMesh: An open-source, interactive, multiblock approach to anatomic finite element model development

Nicole M. Grosland [a,b,c,*], *Kiran H. Shivanna* [c], *Vincent A. Magnotta* [d,c], *Nicole A. Kallemeyn* [a,c], *Nicole A. DeVries* [a,c], *Srinivas C. Tadepalli* [a,c], *Curtis Lisle* [e]

[a] *Department of Biomedical Engineering, The University of Iowa, Iowa City, IA, United States*
[b] *Department of Orthopaedics and Rehabilitation, The University of Iowa, Iowa City, IA, United States*
[c] *Center for Computer Aided Design, The University of Iowa, Iowa City, IA, United States*
[d] *Department of Radiology, The University of Iowa, Iowa City, IA, United States*
[e] *KnowledgeVis, LLC, Maitland, FL, United States*

ARTICLE INFO

Article history:
Received 10 July 2008
Received in revised form
15 October 2008
Accepted 2 December 2008

Keywords:
Finite element method
Mesh generation
Multiblock mesh
Anatomic models
Open-source software

ABSTRACT

Finite element (FE) analysis is a valuable tool in musculoskeletal research. The demands associated with mesh development, however, often prove daunting. In an effort to facilitate anatomic FE model development we have developed an open-source software toolkit (IA-FEMesh). IA-FEMesh employs a multiblock meshing scheme aimed at hexahedral mesh generation. An emphasis has been placed on making the tools interactive, in an effort to create a user friendly environment. The goal is to provide an efficient and reliable method for model development, visualization, and mesh quality evaluation. While these tools have been developed, initially, in the context of skeletal structures they can be applied to countless applications.

1. Introduction

Since its inception in the orthopaedic literature in 1972 [1], the finite element (FE) method has been widely used to evaluate the mechanical behavior of biological tissues such as bone, ligaments, and articular cartilage. Musculoskeletal FE applications initially followed that of traditional engineering mechanics. That is, to assess the probability of structural failure, given the applied loads and constituent properties of the structural material [2]. Over the years, the scope of orthopaedic FE applications has broadened substantially. Among other things, the FE method has been used to assess: fracture risk; the optimality of bone structure; the processes of bone remodeling; prosthetic design issues; the mechanics of soft hydrated tissues; the mechanics of tissues down to the microstructural and cellular levels [3]. Unfortunately, a major drawback that has precluded the routine use of this method has been the prohibitive amount of manual labor required to generate the three-dimensional models necessary to properly characterize the complete stress field in biological structures. The majority of analyses reported in the literature refer to a single, or ‘average’, bony geometry, although in many cases the anthropometric variability of bone size and shape should not be neglected. Furthermore, mesh refinements and convergence checks prove challenging for this type of mesh. As a result, compromises may include sub-

* *Corresponding author at*: 1418 Seamans Center for the Engineering Arts and Sciences, Department of Biomedical Engineering, The University of Iowa, Iowa City, IA 52242, United States. Tel.: +1 319 335 6425; fax: +1 319 335 5631.
E-mail address: nicole-grosland@uiowa.edu (N.M. Grosland).
0169-2607/$ – see front matter
doi:10.1016/j.cmpb.2008.12.003

optimal mesh refinement, homogeneously modeled regions of (heterogeneous) bone [4], or simplifying assumptions of symmetry [5–11]. These limitations are ever more prevalent when detailed anatomic models are considered.

The finite element method requires that the physical domain in which the problem is posed be discretized completely. Furthermore, the accuracy of the numerical solution relies heavily on the nature of the mesh used to represent the physical domain. As the problem sizes have increased and the structural geometries have become more complex, mesh generation algorithms have had to adapt to accommodate these challenges. There are two broad types of mesh generation schemes—routines for structured and unstructured meshes [12–14], both of which are widely used in technical and industrial applications. The techniques for generating structured grids are based on rules for geometrical grid-subdivisions and mapping techniques; producing triangular or quadrilateral elements in two-dimensional analyses, and tetrahedral and hexahedral elements in three-dimensions. Structured grids, as the name implies, have a regular topology where the neighborhood relation between all points is captured with a two- or three-dimensional array. By incrementing or decrementing the array index the point neighbors can be directly accessed. For example, if the nodes can be ordered into a regular array (i, j, k), with the assumption that the nodes (i, j, k) and $(i, j, k+1)$, etc. are neighbors, then the grid is described as structured [15]. If the nodes cannot be arranged in such a form, the grid is unstructured. Unstructured grid generation relies on an explicit definition of the connections between nodes to form elements, in addition to the coordinates of the nodes themselves. Although largely synonymous with tetrahedral grids, unstructured grids may alternatively be composed of hexahedral elements (without directional structure) [16]. Hexahedral elements are preferred for many applications. A mathematical argument in favor of the hexahedral element is that the volume defined by one element must be represented by at least five tetrahedral elements, which in turn yields a system matrix that is computationally more expensive, in particular if higher order elements are used. In contrast to the favorable numerical quality of hexahedral meshes, mesh generation is a difficult task.

Structured grid generators are commonly used when strict elemental alignment is mandated by the analysis code or when necessary to capture physical phenomenon. Structured meshing algorithms generally involve complex iterative smoothing techniques that attempt to align elements with boundaries or physical domains. Where non-trivial boundaries are required, "block-structured" techniques can be employed which allow the user to break the domain up into topological blocks. These multiblock grids are a powerful extension of the structured mesh. Structured meshing techniques are applied to a series of interconnected sub-grids or 'blocks'. While the individual blocks remain structured, the blocks fit together in an unstructured manner. As a result, the advantages of structured and unstructured meshes are harnessed. The multiblock technique affords geometric flexibility while retaining computational efficiency.

Mesh generation is a necessary step in any finite element analysis. Inevitably, it constitutes the bulk of the setup time for a problem. This is especially true for anatomic modeling. This is attributed in part to the fact that the quality of the computed solution is highly dependent on the quality of the mesh. Moreover, it stems from that fact that few meshing strategies, or more specifically pre-processing packages, have been designed with an emphasis on anatomic model development. Toward this objective, we have established an open-source finite element pre-processing environment called IA-FEMesh (Iowa FE Mesh) to accelerate the development and sharing of anatomic finite element models (http://www.ccad.uiowa.edu/mimx/IA-FEMesh/).

This document describes the grid generation and refinement techniques used to create a multiblock structure for meshing, and more importantly the novel editing operations introduced for affording the user additional control over the resulting mesh. In addition to the mesh generation routines, algorithms have been included for improving the ensuing mesh, while a mesh quality viewer has been incorporated for displaying a number of quality metrics inherent to the resultant mesh. For ease of illustration, the modeling practices are demonstrated by meshing the proximal phalanx bone of the index finger.

2. IA-FEMesh overview

Anatomic models initiating with an image dataset (i.e., CT/MR) are often processed to yield a 3D triangulated surface representation of the structure(s) of interest. IA-FEMesh assumes these surfaces form the foundation for the structural geometry, while a series of building blocks are used to establish the mesh. Consequently, each FE model initiates with a triangulated surface representation (STL or VTK format). For example, anatomic surfaces may be generated directly from a segmented image dataset, while implantable devices may be created via a CAD software package of choice, converted to an STL or VTK file format, and imported directly into IA-FEMesh.

Thereafter, a building block, or series of blocks, is constructed, assigned a desired mesh density and projected onto the surface representation. The operator has the option of creating a surface mesh composed of triangular or quadrilateral elements, or a volumetric mesh composed solely of hexahedral elements. Once the mesh is established, material properties and loading/boundary condition assignments may be made. Thereafter, the model may be exported in ABAQUS (SIMULIA, Dassault Systèmes, Providence, RI [17]) file format for analysis.

Central to the multiblock meshing technique is the block structure definition. Toward facilitating these definitions, we have developed an interactive building block technique, demonstrated in Fig. 1. As illustrated, a single block may be sufficient (e.g., a phalanx bone of the hand). In such cases, the building block is automatically defined at the request of the operator, the dimensions of which are established directly from the bounds of the surface of interest (Fig. 1a). Subsequent interactive manipulations may be performed on the building block to provide control over the resultant nodal projections (Fig. 1b and c). Each building block is composed of mesh seeding arranged in rows, columns, and layers; the corresponding level of seed refinement is specified by the user (Fig. 1d). The mesh seeds of the building block are then projected (via clos-