A Fast Algorithm for Generating Large Tetrahedral 3D Finite Element Meshes from Magnetic Resonance Tomograms

Ulrich Hartmann, Frithjof Kruggel
Max-Planck-Institute of Cognitive Neuroscience
Inselstraße 22-26, 04103 Leipzig, Germany
{hartmann, kruggel}@cns.mpg.de

Abstract

This paper addresses the problem of generating three-dimensional (3D) finite element (FE) meshes from medical voxel datasets. With our background in cognitive neuroscience, we deal with brain MR tomograms of up to 256^3 voxels which contain a multitude of incompletely definable, complex-shaped objects. We describe an algorithm that allows the fast and stable creation of very large 3D meshes with well-defined geometric properties. The task of generating anisotropic meshes consisting of up to one million tetrahedra is fulfilled within minutes on a standard workstation. As the angles of the tetrahedra have a direct influence on the stability of the finite element analysis, special care has been taken to assess the element quality. Our algorithm is based on the idea of an image-based spatial decomposition of the problem domain yielding smaller subproblems that can efficiently be handled. Our primary purpose is to set up mechanical and electro-magnetical finite element models of the brain. However, our FE meshes could also be useful in other types of finite element analyses or as deformable volume models for shape descriptions and shape comparisons.

1. Introduction

The finite element (FE) method for the numerical solution of partial differential equations (PDEs) has a large field of applications in its classical domain, the engineering sciences [19, 7]. Recently, this method gained interest in the medical field. The finite element method is used for

- locating electro-magnetical sources of the brain [31],
- modelling irradiation in tumor therapy [24],
- simulating the mechanical system answer of the head under impact [18, 11] and
- for surgery planning and the prediction of human facial shape after craniofacial surgery [16].

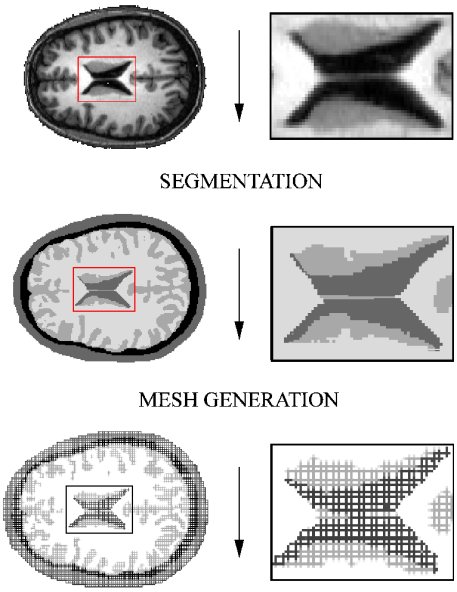

Figure 1. *From MRI data to finite elements: By the means of image processing a segmented image is obtained which can be used as starting point for the mesh generation process.*

Usually FE models for medical problems are based on tomographic examinations from magnetic resonance (MR) or X-ray scanners. A proper segmentation of the medical images yields objects (i.e. bone, brain, ventricles) as homogeneous regions, to which tissue properties (i.e. electrical conductivity or mechanical elasticity) are assigned. Then, a finite element description of those objects has to be introduced by a mesh generation procedure (see Figure 1). By considering the connectivity of the elements and their

tissue properties, an equation system is set up and solved by appropriate numerical techniques. The solution of an FE calculation yields a physical property depending on the underlying PDE. The spatial diplacements or the electrical potentials at the mesh nodes are typical results of a finite element analysis.

The problem of 3D mesh generation is a current research topic within the field of computational geometry. All the solutions proposed can be roughly subdivided into two classes.

The most common approach to create a 3D mesh starts with a boundary representation of a polyhedral object [2, 14, 15, 21, 22]. During the mesh generation, so-called "Steiner points" are placed within the interior of the object. It has been shown that in three and higher order dimensions it is generally impossible to tesselate a polyhedral object without introducing Steiner points [30]. A Delaunay tesselation is then applied to generate the finite elements [6, 3]. Often a post-processing step is required to either remove degenerate elements or reconfigure the mesh, so that the elements satisfy predefined quality criteria (for definition, see [20, 10]).

A second approach is based on a recursive spatial decomposition in the geometrical domain, in which a 3D region is subdivided into cubes or octants [28, 29, 33, 25, 23, 32]. This subdivision may result in an isotropic or anisotropic mesh, imposed by object boundaries or resolution limits as stopping criteria [26, 27].

Object descriptions for both types of algorithms are hard to generate from medical datasets, so the application of these mesh generators poses a number of problems:

- Tomographic examinations yield voxel datasets, which first have to be segmented into objects.
- Imperfections in the imaging process (non-linear behaviour of the scanner, motion artifacts, partial-volume effect, noise, etc.) will lead to segmentation errors and likewise to an incompletely defined object.
- Segmentation errors lead to incompletely defined or roughly approximated object boundaries.
- Shapes of anatomical objects are generally complex and highly non-convex. This will (i) complicate geometry-based algorithms and (ii) result in a large number of elements generated.
- Multiple objects of different tissue types are present in a tomogram and must be handled simultaneously.

From several commercial and non-commercial mesh generators available to us none was able to successfully build a tetrahedral mesh from a MR tomogram of the brain. The following drawbacks arise with the approaches mentioned above:

- Boundary-oriented approaches require a polyhedral definition of an object that serves as input for the mesh generator. Since typically hundreds of connected components are found in a segmented brain tomogram, which results in a complex mesh generation procedure.
- Given the highly complex structures in the brain, a few of the larger objects exhibit $10^4 - 10^5$ boundary vertices, which lead to numerical stability problems in the Delaunay tesselation algorithms.
- On incompletely defined boundaries, the subdivision approach generates a huge number of small elements or even fails completely.
- The subdivision approach has considerable requirements in terms of memory and computational resources.

Because of these disadvantages the cited approaches are not suitable for the generation of FE meshes based on medical datasets. The algorithm we describe in this article will take advantage of the discretization of space inherent in a tomogram. A well-defined subset $\mathcal{M}$ of the set $\mathcal{A}$ of all image lattice points will represent the nodes of the whole FE mesh. By processing subsets of $\mathcal{M}$ we decompose the problem into small subproblems that can efficiently be handled. The precise mathematical definitions will be given in the following chapter and the above idea will be described in more detail in chapter 3, where we will present our algorithm and discuss its behaviour and properties. Chapter 4 shows FE meshes of neuroanatomical structures produced with our algorithm and focuses on possible applications and further developments of our new technique.

2 Mathematical Definitions

Three-dimensional medical images can be thought of as a set of addresses with each address being labeled with a defined gray value. The set of addresses which is often referred to as the image lattice can formally be written as:

$$\begin{aligned} \mathcal{A} = \{(s,r,c) \mid \quad & 0 \leq s < nslices, \\ & 0 \leq r < nrows, \\ & 0 \leq c < ncolumns\}, \end{aligned} \tag{1}$$

where *nslices, nrows, ncolumns* denote the number of slices, rows and columns of the 3D image. Every address can be interpreted as a point in the Euclidean space (depicted as black circles in Figure 2). These points may likewise be referred to as nodal points or nodes. The assignment of labels to the elements of $\mathcal{A}$ can be formulated as a mapping

$$f_{image} : \mathcal{A} \rightarrow \mathcal{G}, \tag{2}$$

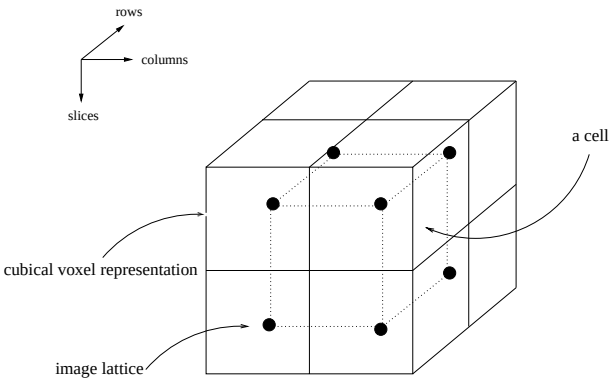

Figure 2. *This image explains the terms image lattice, voxel representation and cell that are used throughout the text.*

where $\mathcal{G}$ for MRI images is normally chosen as $\mathcal{G}$ = {0, 1, ..., 255}. Within this framework a voxel v can be defined as a an element of the set $\mathcal{A}$ with a label l $\in \mathcal{G}$ given by (2). Voxels are in almost any case visualized as intensity coded cubes defined in the Euclidean space (see Figure 2). If we apply a function $f_{segment}$ to the set of voxels $\mathcal{V}$ that classifies its elements v depending on their intensity l and then assign a unique label u to every member of a class we obtain a segmented image L. Mathematically written, the voxels of the original image are subject of the mapping

$$f_{segment} : \mathcal{V} \rightarrow \mathcal{U}, \tag{3}$$

where $\mathcal{U}$ = {0, 1, ..., u_{max} } denotes the set of all assigned labels u.

An important role in our algorithm plays the concept of a mathematical entity that we will call a *cell* in the remainder of this paper. We define a cell c as a cubus whose corner points are a subset of $\mathcal{A}$, i.e. a cell is delimited by eight image lattice points. A cell may also have image lattice points on its faces and edges. The set of image lattice points belonging to the ith cell will be denoted as $\mathcal{N}_i$ and the j-th member of $\mathcal{N}_i$ will be referred to as the cellular node n_i^j. The set of all cells is called $\mathcal{C}$ and its number of members w will be dependent on the image structure and the degree of mesh anisotropy chosen by the user. $\mathcal{C}$ is defined as

$$\mathcal{C} = \{\mathcal{N}_1 \cup \mathcal{N}_2 \cup ... \cup \mathcal{N}_w\}. \tag{4}$$

The edge length e of a cell must satisfy the constraints $0 < e < 256$. A simple eight-noded cell will be called a brick in the following. Now we can mathematically describe the set of nodes $\mathcal{M}$ that constitute a mesh:

$$\mathcal{M} = \mathcal{C} \cap \mathcal{A} \tag{5}$$

The elements of $\mathcal{M}$ are numbered with the labels k_1 to k_n with n = card($\mathcal{M}$).

Finally we want to give a formal description of a *mesh*:

A mesh is represented by the combination of a set $\mathcal{M}$ with a set of elements $\mathcal{E}$. A member of $\mathcal{E}$ is described by several integral numbers, in the case of a tetrahedron five numbers suffice:

$$k_l \quad k_h \quad k_v \quad k_w \quad m \tag{6}$$

The first four numbers are the labels of the elements of $\mathcal{M}$ that define the corner points of a tetrahedron. The last number is a unique label m assigned to the element during the mesh generation process that can be interpreted as a material number.

3 The Algorithm

3.1 Introduction

The basic idea of our new algorithm is to simplify the task of meshing by introducing cells of different size into the MRI dataset and then handle the cells according to the number of cellular nodes n_i^j. Without loss of generality we will assume a 3D segmented MRI dataset L as input for our mesh generator. Voxels with the label u=0 will be interpreted as image background and the creation of background elements is suppressed. In our problem domain, the labels $u \neq 0$ correspond to different objects (i.e. white matter, bone, ventricles) and/or regions with specific material properties (i.e. electrical conductivity, stiffness). Our algorithm can be subdivided into three steps:

1. Isotropic subdivision of the input image into bricks.
2. Collection of bricks to build up cells.
3. Cellular mesh generation phase.

3.2 Subsampling and collection

Figure 3 visualizes the result of every step for a segmented skull-brain dataset with an isotropic resolution of 1 mm. In this example the original image is first decomposed into bricks (for definition see preceding chapter). To preserve relatively fine skull structures the user-specified subsampling factor e_{min} defining the edge length of the bricks must not be too high. Typically we choose a value of 2 (mm). Each of these bricks has already been assigned its specific "material" label. As the i-th brick is defined by eight cellular nodes n_i^j (j=1, ..., 8) we can define its label by (i) converting the brick nodes into voxels by the means of the mapping function (eq. 2) and (ii) finding its class label u with of the help of the segmentation function (eq. 3). The most frequently found label within a brick will become its material label m. In the case of two labels being equally

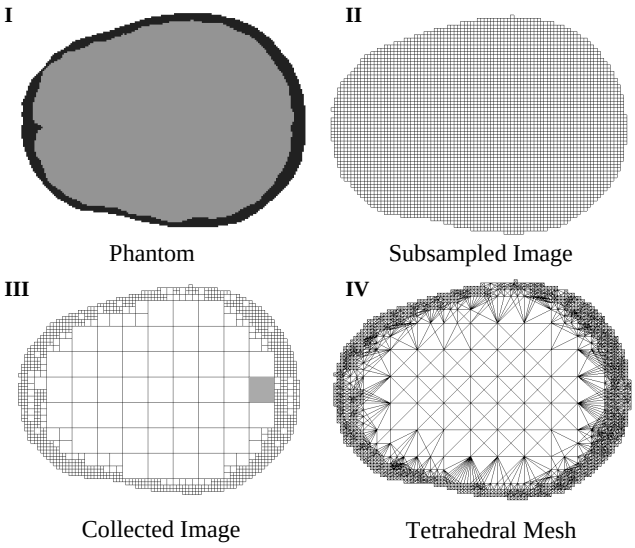

Figure 3. *This image shows the three phases of our algorithm. In a first step the phantom (I) representing the skull-brain system is subdivided into bricks of identical edge length (II). The result of the subsampling phase is an anisotropic decomposition of the image into cells (III). This cell image serves as starting point for the mesh generation (image IV).*

frequent a random function decides to which class the brick under consideration belongs. The creation of bricks with material labels of zero is suppressed.

The subsampling is followed by the collection step. By recursively traversing the subsampled image, bricks of identical material label are collected to form cells with larger edge lengths. A cell inherits the material label of its bricks. The maximal edge length e_{max} of all cells is given by a user-specified parameter, the so-called collection factor. The quotient of the collection factor and the subsampling factor will be referred to as the anisotropy factor. In the shown example this factor was set to be eight. One can clearly see in image III that cells of different edge lengths have been produced. Within the homogeneous region cells with e_{max} = 16 have been created whilst in the complex-shaped skull region most of the cells have the edge length e_{min} = 2. In the transition region between brain and skull cells with $e_{min} < e < e_{max}$ have been formed. As we know from chapter 2, the i-th cell is represented by a set $\mathcal{N}_i$. We find the number of cellular nodes (i.e. card($\mathcal{N}_i$)) by looking at the neighbours of the cell of interest. An examination of the gray-marked cell c_{gray} in image III reveals that card($\mathcal{N}_{gray}$) = 39, i.e. 39 cellular nodes n^j_{gray} are necessary to maintain the neighbourhood relationships. The cardinals of the sets ($\mathcal{N}_i$) range from 8 to 150 (i=1, ... card($\mathcal{C}$)). As mentioned above our algorithm locally processes these sets according to their number of elements. We will discuss this point in detail with the help of a simple mathematical phantom. Before doing this it should be noted that the subsampled as well as the collected image could be interpreted as FE meshes if the cells are considered as finite elements. For calculations with an anisotropic mesh (see image III) a more complicated formulation of the finite element method has to be implemented. We are not pursuing this case within this article. The result of the last step, the mesh generation, is depicted as image IV. In the complex-shaped region we see a much higher resolution as in the homogeneous one. This is reasonable when using FEM for mechanical applications as stress and deformations are more variable at boundary regions and geometrically complicated areas.

3.3 Generation of anisotropic tetrahedral meshes

Figure 4 shows a simple mathematical phantom consisting of 128 x 128 x 32 voxels (card(A) = 524288). The value of the anisotropy factor was chosen to be two. During the collection phase the algorithm detects the small structure and creates cells of appropriate edge lengths in this region (see image in the right upper corner of Figure 4). Execpt for three (marked with a "D"), all cells are bricks. The bricks are processed by a geometrical subdividion into five tetrahedra according to Figure 5. The remaining cell sets $\mathcal{N}_i$ are containing either 13 or 9 elements. These "complex" cells are tesselated by a Delaunay algorithm. To produce tetrahedra with faces compatible with those of neighbouring cells, we selected an incremental tesselation algorithm. We have found the Clarkson algorithm [4] to be modifiable for our purpose. Since card($\mathcal{N}_i$) for i=1, ... card($\mathcal{C}_i$) is relatively small, tesselation is numerically stable and fast. With the determination of the intersection $\mathcal{M}$ (card($\mathcal{M}$) = 69) and of the elements of $\mathcal{E}$ (card($\mathcal{E}$) = 147) the generation of the phantom mesh is accomplished. The material number m of a certain tetrahedron is determined by the label m of its "mother cell". Figure 8 shows an outline of an implementation of our algorithm.

3.3.1 Quality of tetrahedra

For the sake of stability of any FE simulation carried out with tetrahedral elements the tetrahedra should satisfy certain quality criteria. As a rule of thumb, angles below 10 degrees are more likely to cause stability problems when solving the equation system. For the tetrahedra that are result of a brick decomposition (see Figure 5) this requirement is fulfilled in any case. We only find 3 different angles (all greater than 10 degrees) and 3 different edge lengths. In the "Delaunay case" the situation is more difficult. The creation of degenerate tetrahedra (all its nodal points being coplanar)

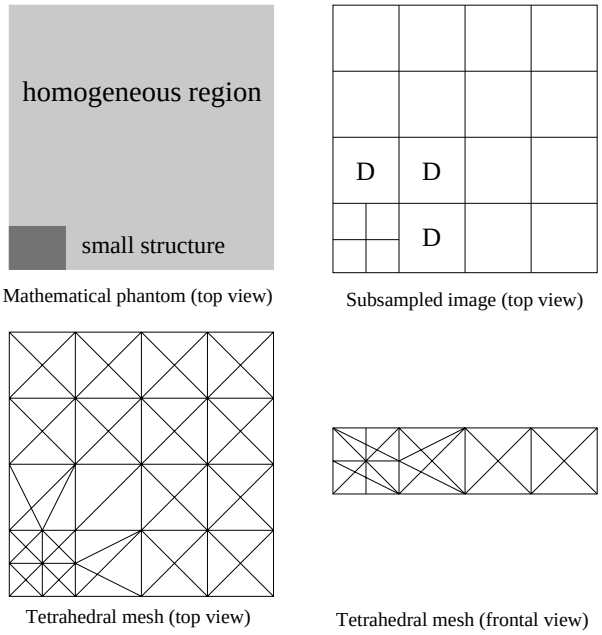

Figure 4. *This set of images explains the behaviour of the algorithm in the presence of a small structure. The image in the upper right shows the result of the subsampling phase. Cells with larger edge lengths have been built in the homogeneous region and those with smaller edges in the area of the structure. Dependent on the number of nodes the bricks are either Delaunay-triangulated (marked with a "D") or simply subdivided into five tetrahedra (see Figure 5). The results of the triangulation are depicted in the lower row.*

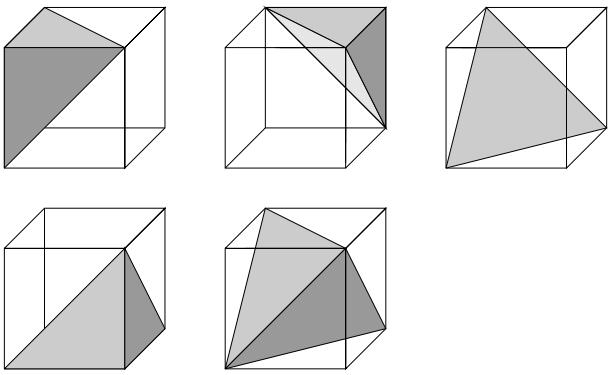

Figure 5. *Subdivision of a brick into five tetrahedra.*

has to be suppressed by fixing the order by which the n_i^j are handed to the Delaunay algorithm. As all cellular nodes are lying on a convex polyhedron (a cube) all angles in the cells are greater than 10 degrees if we limit the anisotropy factor to the value eight. It it obvious that the anisotropy factor influences the angles of the tetrahedra. In the worst case the smallest edge of a tetrahedron has the value e_{min} and largest is of the size e_{max}. If the ratio of these values (i.e. the anisotropy factor) is chosen too high very small angles might occur. To avoid the construction of such "minor quality" tetrahedra a Steiner point could be introduced in the center of a cell. This possibility has not yet been implemented because for our applications an anisotropy factor of eight has proven to be sufficiently high to remarkably reduce the number of nodes (see Table 1).

3.3.2 Numbering of mesh nodes

An additional positive feature of our mesh generator is the "natural" numbering of the nodes. The way the nodes are numbered determines the bandwidth of the FE matrices. As a large bandwidth of an FE matrix may result in numerical instabilities during the FE analysis, meshes often have to be postprocessed by renumbering algorithms [8, 5]. These algorithms aim at minimizing the difference D between the largest and the smallest node numbers in an element. The relation between D and the bandwidth BW of an FE matrix is given by the formula

$$BW = \max_e [D^{(e)}] + 1. \tag{7}$$

Time- and memory-consuming renumbering of nodes is unnecessary for meshes produced by our algorithm. As our mesh generator creates the elements during a row-by-row and slice-by-slice traversal through the MRI dataset, the maximum difference D_{max} is equal to the number of nodes that are produced for generating the FE elements of a single MR slice.

4 Results and Outlook

The algorithm has been implemented in the C++ language as a module in the BRIAN environment [17]. The algorithm has been tested with a number of MR datasets which were preprocessed in the following way:

- Acquisition of a T1-weighted MR brain dataset, GRE sequence, 128 slices, in-plane resolution of 0.91mm, interslice distance 1.4mm.
- Trilinear interpolation to an isotropic resolution of 1mm.
- (Optional) noise filtering with a Lee filter.
- K-means classification using 5 classes.

Individual objects (e.g. the white matter, ventricles, etc.) were extracted by thresholding the k-means classification, labeling connected components, and selecting a specific object by its label. Figure 7 shows a mesh of a segmented anatomical object, the ventricles. To gain quantitative knowledge about the properties of our algorithm, we

generated meshes from a labeled 192x192x200 MR brain dataset containing five different labels in 12 objects. Tab. 1 lists a compilation of the results obtained by choosing different spatial resolutions and varying anisotropy factors.

The test cases prove the evident relation between card($\mathcal{E}$) and and the anisotropy factor e_{max}/e_{min}. Obviously, the number of created nodes also depends on the size of the homogeneous regions in the input image. Because we do not expect homogeneous regions in our datasets to be much larger than 16^3 voxels, it is not useful to set e_{max} to greater values than 16. Experiments with e_{min}=1 have not been carried out, because the fivefold subdivision of such cells should yield elements smaller than the spatial resolution of the MRI scanner. For our datasets we have found anisotropy factors of four and eight to be most useful. As listed in Tab. 1, this setting cuts down the number of nodes (which determines the memory requirements) by up to 80 % relative to the number of the corresponding isotropic case. Expectedly, the node reduction effect becomes smaller with increasing e_{min} and with decreasing e_{max}.

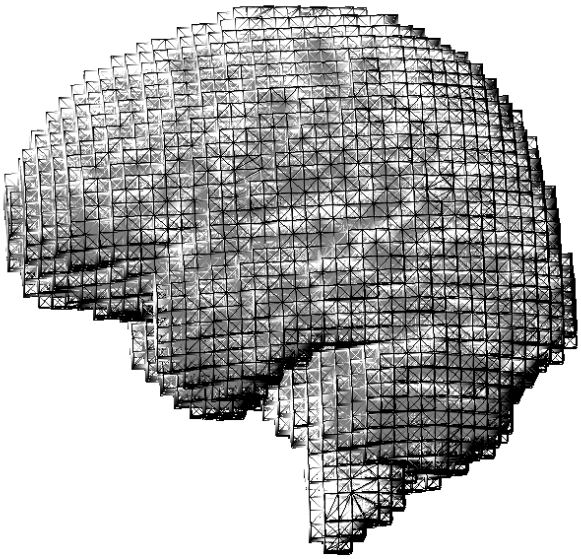

Figure 6. *A mesh of the human brain (41000 mesh nodes and 188000 elements). The regular subdivision of the bricks is visible especially at the boundary. Some Delaunay tesselated cells can be seen in the region of the brain stem.*

The execution times to generate these meshes are also listed in Table 1. Highly resolved meshes are created within less than four minutes, coarser meshes do not even take more than a few seconds. Figure 6 shows a brain mesh consisiting of about 41000 nodes as a typical output of our mesh generator. Before we are going to discuss possible further developments of the algorithm we would like to compile the characteristics of our mesh generator:

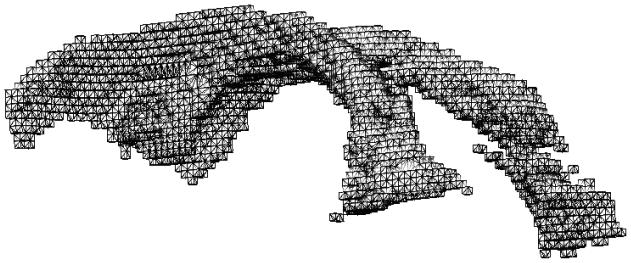

Figure 7. *A mesh of a neuroanatomical object. The large inner cavities, the ventricles, are depicted as a tetrahedral mesh consisiting of 1579 nodes and 3006 elements.*

- It is fast: producing a mesh consisting of around 100000 tetrahedra takes half a minute.
- Tetrahedra produced have a high quality with regard to FE analyses.
- The Delaunay triangulation is stable for node numbers of 10 to 150.
- Node renumbering is not necessary.

Though the mesh generator is fully applicable in its current state, further improvements can be thought of.

- The implementation of local refinement strategies could be desirable when using our mesh generator for calculations aiming at locating electro-magnetical sources of the brain.
- The extraction of object boundaries could be useful for visualization purposes.

In this paper we have described an algorithm to generate 3D meshes from MR tomograms. These meshes are targeted towards applications in the finite element method of solving PDEs. Previous approaches required the formation of boundaries and well-defined polyhedral objects, which are hard to generate from medical image datasets. Moreover, we have found stability problems of these algorithms when applied to the highly non-convex objects in the brain.

By spatial decomposition of the voxel dataset into cells, we could derive a fast and numerically stable algorithm for the generation of anisotropic tetrahedral grids. By collecting homogeneous regions in larger cells (and thus building an anisotropic grid) we were able to cut down the number of mesh nodes by up to 80 %.

Using this mesh generator, we were able to set up finite element models with the generic scanner resolution of 1mm, which were solved on our parallel computer in a few minutes [1, 9]. By limiting the spatial resolution to 2mm, these models fit into the memory of conventional workstations.

Our primary goal is to study the consequences of focal brain damages by mechanical FE models [13, 12] and the source localization of neural brain activity by electromagnetical FE models. However, other types of PDEs may be handled the same way: growth processes in the brain, atlas transformations, or irradiation in tumor therapy.

Finally, FE analysis is becoming a tool in the biomedical engineering and the computational neurosciences.

References

[1] R. Beck, B. Erdmann, and R. Roitzsch. Kaskade 3.0 - an object-oriented adaptive finite element code. *Technical Report TR 95-4, Konrad-Zuse-Zentrum $f\ddot{u}r$ Informationstechnik Berlin*, 1995.

[2] E. Boender. Reliable delaunay-based mesh generation and mesh improvement. *Communications in Numerical Methods in Engineering*, 10:773–783, 1994.

[3] P. Cignoni, C. Montani, and R. Scopigno. A merge-first divide and conquer algorithm for e^d delaunay triangulations. *Technical Report 92-16*, Instituo CNUCE-C.N.R, Pisa, Italy, 1992.

[4] K. Clarkson, K. Mehlhorn, and R. Seidel. Four results on randomized incremental constructions. *Computational Geometry: Theory and Applications 3*, pages 185–212, 1992.

[5] E. Cuthill and J. McKee. Reducing the bandwidth of sparse symmetric matrices. *Proceedings of the 24th National Conference of the Association for Computing Machinery*, pages 157–172, 1969.

[6] H. . Edelsbrunner. *Algorithms in combinatorial geometry*. Springer Verlag, Berlin, Heidelberg, 1987.

[7] C. Fletcher. *Computational techniques in fluid dynamics*. Springer Verlag, Berlin, Heidelberg, 1991.

[8] N. Gibbs, W. Poole, and P. Stockmeyer. An algorithm for reducing the bandwidth and profile of a sparse matrix. *SIAM Journal of Numerical Analysis*, 3:236–249, 1976.

[9] J. Gobat and D. Atkinson. Felt: User's guide and reference manual. *Computer Science Technical Report CS94-376, University of California, San Diego*, 1994.

[10] N. Golias and T. Tsiboukis. An approach to refining three-dimensional tetrahedral meshes based on delaunay transformations. *International Journal of Numerical Methods in Engineering*, 37:793–812, 1994.

[11] U. Hartmann and F. Kruggel. Ein dreidimensionales mechanisches finite-elemente-modell des menschlichen gehirns. *Proceedings of 5th Freiburg Workshop on Digital Image Processing in Medicine*, pages 219–224, 1997.

[12] U. Hartmann and F. Kruggel. Ein virtueller dummy zur simulation des sch$\ddot{a}$del-hirn-traumas. *German edition of the Scientific American*, September 1997.

[13] U. Hartmann and F. Kruggel. Transient analysis of the biomechanics of the human head with a high-resolution 3d finite element model. *Computer Methods in Biomechanics and Biomedical Engineering*, accepted, 1997.

[14] B. Joe. Construction of three-dimensional delaunay triangulations using local transformations. *Comuter Aided Geometric Design*, 8:123–142, 1991.

[15] B. Joe. Three-dimensional boundary-constrained triangulations. *Artificial Intelligence, Expert Systems, and Symbolic Computing, (Eds. Houstis, E.N., Rice, J.R.) Elsevier Publishers*, pages 215–222, 1992.

[16] R. Koch, M. Gross, F. Carls, D. von Buren, G. Fankhauser, and Y. Parish. Simulating facial surgery using finite element models. *Computer Graphics Proceedings, Annual Conference Series*, pages 412–428, 1996.

[17] F. Kruggel and G. Lohmann. Brian (brain image analysis) - a toolkit for the multimodal analysis of brain datasets. *Proceedings, Computer Assisted Tomography '96, Elsevier Amsterdam*, pages 323–328, 1996.

[18] A. Kuijpers, M. Claessens, and S. A.A.H.J. The influence of different boundary conditions on the response of the head to impact: a two-dimensional finite element study. *Journal of Neurotrauma*, 12:715–724, 1995.

[19] P. Lewis and J. Ward. *The finite element method: principles and applications*. Addison Wesley Reading, 1991.

[20] A. Liu. *Quality local refinement of tetrahedral meshes*. PhD Thesis, Department of Computing Science, University of Alberta, 1994.

[21] S. Lo. Volume discretization into tetrahedra - i. verification and orientation of boundary surfaces. *Computer and Structures*, 39:493–500, 1991.

[22] S. Lo. Volume discretization into tetrahedra - ii. 3d triangulation by advancing front approach. *Computer and Structures*, 39:501–511, 1991.

[23] L. Mitchell and S. Vavasis. Quality mesh generation in three dimensions. *Proceedings of the ACM Computational Geometry Conference*, pages 212–221, 1992.

[24] K. Paulsen, X. Jia, and J. Sullivan. Finite element computations of specific absorption rates in anatomically conforming full-body models for hyperthermia treatment analysis. *IEEE Transactions on Biomedical Engineering*, 40:933–945, 1993.

[25] R. Perucchio, M. Saxena, and A. Kela. Automatic mesh generation from solid models based on recursive spatial decompositions. *International Journal of Numerical Methods in Engineering*, 28:2469–2501, 1989.

[26] F. Schindler and R. Schneiders. Automatic geometry-adaptive generation of quadrilateral and hexahedral element meshes for fem. *5th International Conference on Numerical Grid Generation in Computational Field Simulations*, pages 689–697, 1996.

[27] L. Schmelzer. *Covariant geometry description, WIAS-Preprint No. 152.* Weierstraß-Institute Berlin, 1995.

[28] W. Schroeder and M. Shepard. A combined octree/delaunay method for fully automatic 3d mesh generation. *International Journal of Numerical Methods in Engineering*, 20:37–55, 1990.

[29] M. Shepard and M. Georges. Automatic three-dimensional mesh generation by the finite octree technique. *International Journal of Numerical Methods in Engineering*, 32:709–749, 1991.

[30] N. Weatherill and H. O. Efficient three-dimensional delaunay triangulation with automatic point creation and imposed boundary constraints. *International Journal for Numerical Methods in Engineering*, 37:1841–1861, 1994.

[31] Y. Yan, R. Nunez, and R. Hart. Finite element model of the human head: scalp potentials due to dipole sources. *Medical and Biological Engineering and Computing*, pages 475–481, 1991.

[32] M. Yerry and M. Shephard. Automatic three-dimensional mesh generation by the modified-octree technique. *International Journal of Numerical Methods in Engineering*, 20:1965–1990, 1994.

[33] M. Yuen, S. Tan, and K. Hung. A hierarchical approach to automatic finite element mesh generation. *International Journal of Numerical Methods in Engineering*, 32:501–525, 1991.

e_{min}	2	4	8	2	2	2	4	4	8
e_{max}	2	4	8	4	8	16	8	16	16
card($\mathcal{E}$)	863540	108250	13475	244325	188673	186954	45510	42228	8639
card($\mathcal{M}$)	185914	24944	3531	53989	41029	40095	10574	9516	2196
NR [%]	-	-	-	71	78	79	58	62	38
t [s]	211	31	10	190	185	181	35	32	10

Table 1. *Compilation of the number of elements and nodes produced by our algorithm for the generation of meshes with different anisotropy factors. e_{min} denotes the subsampling factor, e_{max} stands for the collection factor, NR represents the percentage by which the number of nodes is reduced compared with the corresponding isotropic case and t is the total execution time on an SGI 02 workstation.*

ANISOTROPIC TETRAHEDRAL MESH GENERATOR {

```
   SUBSAMPLING PHASE {
      formation of bricks with edge length e_min
      labeling of bricks with most frequent voxel label l
   }

   COLLECTION PHASE (determination of C) {
      merging of bricks to form cells with a maximal edge length e_max
      description of cells by a set N_i
      labeling of cells with brick label m
   }

   MESH GENERATION PHASE (determination of M and E) {
      creation of tetrahedra
      for i = 1 to i = card(C) {
         if( card(N_i) = 8) {
            creation of tetrahedral elements as shown in Figure 5
            assignment of material label m_i to the five tetrahedra
         }
         else {
            ordering of nodes to avoid creation of degenerate tetrahedra
            Delaunay tesselation of the set N_i
            assignment of material label m_i to the produced tetrahedra
         }
      }
      creation of mesh node list
      determination of the intersection M
   }
}
```

Figure 8. *Outline of our algorithm for the generation of anisotropic tetrahedral meshes.*