INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2006; **65**:734–751
Published online 1 September 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nme.1465

Efficient surface reconstruction from contours based on two-dimensional Delaunay triangulation

Desheng Wang*,†, Oubay Hassan‡, Kenneth Morgan§ and Nigel Weatherill¶

Civil and Computational Engineering Centre, School of Engineering, University of Wales Swansea, Singleton Park, Swansea SA2 8PP, U.K.

SUMMARY

This paper introduces an efficient method for surface reconstruction from sectional contours. The surface between neighbouring sections is reconstructed based on the consistent utilization of the two-dimensional constrained Delaunay triangulation. The triangulation is used to extract the parametric domain and to solve the problems associated with correspondence, tiling and branching in a general framework. Natural distance interpolations are performed in order to complete the mapping of the added intermediate points. Surface smoothing and remeshing are conducted to optimize the initial surface triangulations. Several examples are presented to demonstrate the effectiveness and efficiency of the proposed approach. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: surface reconstruction; correspondence; tiling; branching; Delaunay triangulation

1. INTRODUCTION

Various applications in medical imaging, computer graphics, terrain modelling and reverse engineering require a surface reconstruction from a series of planar sectional contours. These contours are derived from the intersections of the boundaries of the object with a set of parallel planes. For medical imaging, these sections can be generated from computed tomography (CT), magnetic resonance imaging (MRI) or ultrasound imaging. The reconstruction of a surface consistent with these parallel contours is a very intriguing and challenging problem, as the construction can be non-unique due to severe topology variations or sparseness of contours [1–6].

*Correspondence to: Desheng Wang, Civil and Computational Engineering Centre, School of Engineering, University of Wales Swansea, Singleton Park, Swansea SA2 8PP, U.K.
†E-mail: desheng.wang@swansea.ac.uk, desheng@ntu.edu.sg
‡E-mail: o.hassan@swansea.ac.uk
§E-mail: k.morgan@swansea.ac.uk
¶E-mail: N.P.Weatherill@swansea.ac.uk

Received 22 February 2005
Revised 26 May 2005
Accepted 22 July 2005

Most existing techniques subdivide the problem into subproblems involving the surface reconstruction between pairs of adjacent contours [1, 3–13]. The surface reconstruction has to address three fundamental issues: the solution to the problem of correspondence, which involves finding the correct connections between the contours of the adjacent sections, the problem of tiling which involves the triangulation of the strip lying between contours of adjacent sections and the branching problem which occurs when a contour in one section corresponds to more than one contour in the adjacent section [1, 11, 12, 14].

Among the various approaches discussed in the literature, one class of algorithms deals with the three problems in a systematic manner and has recently received wide attention [14–19]. The common characteristic of these algorithms is to construct the surface, between neighbouring sections, from a parametric domain which is computed from the overlap of the contours of the two adjacent sections [15–19]. The mapping is implicitly defined through some form of interpolation of the Z-co-ordinates of the neighbouring sections. In many approaches, the parametric domain is triangulated and then the resulting triangulation is mapped to three dimensions. Oliva *et al.* [3] proposed a simplified generalized Voronoi diagram for the parametric domain triangulation. Cong and Parvin [14] improved the method by using the 'equal importance criterion' for the interpolation of intermediate points, which resulted in a very smooth surface with many added points. This class of algorithms is known as surface recovery based on parametric domain triangulation.

In this paper, we apply the constrained Delaunay technique to construct the above parametric domain triangulation and address the issues of correspondence, tiling and branching in a more systematic, robust and efficient way. The correspondence problem is solved by the definition of the parametric domain and favours vertical correspondence [1, 3, 11, 12, 14–19]. Traditionally, the parametric domain is explicitly defined by applying the plane-sweep algorithm for intersection. This has a complexity of $O(n)\ln(n+s)$, where n is the number of contour edges and s is the number of intersections [20, 21]. These algorithms are time consuming and complicated to implement in the case of complex contour configurations, with a large number of vertices. In our algorithm, there is no need for the explicit computation of a parametric domain. The parametric domain computation is implicitly replaced by the classification of the two-dimensional constrained Delaunay triangulation (CDT) of the boundaries forming the contours of the two adjacent sections. The tiling problem is naturally solved by the Delaunay connection of the nodes forming the two sections. The branching problem is addressed by ensuring that the triangulated parametric domain does not contain triangles which have their nodes on the contours of the same section. In this case intermediate points are added to the edges connecting the same contour and these points are inserted into the triangulation using the Delaunay insertion procedure. The equal importance criterion is used for the interpolation of the three-dimensional co-ordinates of the added points as described in Reference [14]. For almost similar neighbouring sections, perturbation or enlargement of one of the contours is usually applied. However, in the case of partial similarity, this may result in the removal of the similarity in one part of the contour but introduces a similarity in a different part of the contour. In the present method, we replace the perturbation method by a snapping method, where, in the parametric domain, the points of one of the similar sections are projected onto the segment of the other section if the distance is smaller than a defined tolerance.

After the initial surface reconstruction, postprocessing, such as surface smoothing based on modified Laplacian smoothing and remeshing based on edge splitting/contractions, is

conducted for the optimization of the surface triangulation. This can then be used for volumetric meshing and subsequent numerical simulations. The reconstruction of various human organs from scanned images, such as bone, heart, lung, and more complex examples including hip, skull and foot, is performed to demonstrate the efficiency and robustness of the proposed method.

The remainder of the paper is organized as follows. First, in Section 2, the underlying principle of parametric domain triangulation based upon the surface recovery method is given. The details of the complete procedure are described in Section 3. Section 4 gives an overview of the surface smoothing and remeshing of the initially constructed triangulation. Finally, numerical examples are presented in Section 5.

2. SURFACE RECOVERY BASED ON PARAMETRIC DOMAIN TRIANGULATION

The input data consist of a sequence of parallel sections. Through an appropriate transformation, each section has a constant Z-co-ordinate and contains a set of contours in the form of non-intersecting closed polygons. The contours which are not contained in any other contours are called *parent polygon*s, and the remaining are called *child polygons*. As the reconstruction can be subdivided into subproblems of surface reconstruction of two neighbouring sections, in what follows we will restrict attention to a single pair of neighbouring sections. In the order of the Z-co-ordinates, the contours on the first section are called the *lower contours*, and the contours on the second section are called the *upper contours*. The problem can be stated as: *given the lower contours* $\mathbf{C}_\mathrm{l}$ (*with co-ordinate* Z_1) *and the upper contours* $\mathbf{C}_\mathrm{u}$ (*with co-ordinate* Z_2), *find a surface* $\mathbf{S}$ *such that* $\mathbf{C}_\mathrm{l}$ *and* $\mathbf{C}_\mathrm{u}$ *is the intersection of* $\mathbf{S}$ *with the plane* Z_i $(i = 1, 2)$.

Obviously, the above reconstruction problem is ill-posed and has no unique solution. Some kind of assumption or requirement of optimality is necessary. Most existing methods are based on minimization or maximization of a given entity, such as minimization of area, or favouring vertical connection [1, 6, 11, 13].

The first step in the reconstruction process is the consistent reorientation of the sectional contours. All the parent polygons are first reoriented in counter-clockwise directions and all the child polygons are reoriented clockwise. The orientation is achieved automatically using the CDT of all the polygons forming the contours of each section.

The surface to be constructed $\mathbf{S}$, is considered as a surface with an implicit parametric expression. Assuming that $\mathbf{A}$ represents the domain closed by the lower contours $\mathbf{C}_\mathrm{l}$, and $\mathbf{B}$ represents the domain closed by the upper contours $\mathbf{C}_\mathrm{u}$, then the parametric domain of $\mathbf{S}$ is defined as $\mathbf{D} = (\mathbf{A} - \mathbf{B}) + (\mathbf{B} - \mathbf{A})$. Figure 1(a) shows the surface $\mathbf{S}$ to be reconstructed between the contours $\mathbf{C}_\mathrm{l}$ and $\mathbf{C}_\mathrm{u}$, and Figure 1(b) shows the projection of the upper contours onto the lower section. The grey area $\mathbf{D}$ shown in Figure 1(c) forms the parametric domain for the surface to be constructed.

The parametric domain co-ordinates are taken to be the Cartesian X–Y-co-ordinates of the vertices forming the contours of the section. This allows a trivial mapping from the parametric domain to the physical domain to be used for the points lying on the upper or lower contours. However, for any point added in the parametric domain, some kind of interpolation, for the Z-co-ordinate, needs to be performed to obtain a complete mapping from the parametric domain to the physical domain. Various types of interpolation can be implemented for this purpose

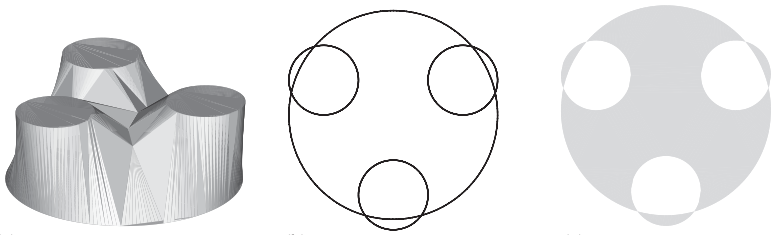

Figure 1. Parametric domain of the surface: (a) the surface to be reconstructed with one lower contour and three upper contours; (b) contours overlay; and (c) parametric domain (grey area).

such as linear or average. Interpolation based on equal importance criterion is presented in detail in Reference [14], and is preferred in the present work. For every added point p, two signed distances, $D_{\mathrm{l}}(p)$ and $D_{\mathrm{u}}(p)$, of point p to the lower and upper contours, respectively, are calculated. The sign is taken to be positive if the point is inside the contours and negative if the point is outside the contours. Denote the Z-co-ordinates of the lower and upper section by Z_1 and Z_2, and the mapping to the physical domain is then completed by the calculation of the Z-co-ordinate of the added point using the equation

$$Z(p) = \frac{D_{\mathrm{u}}(p) * Z_1 - D_{\mathrm{l}}(p) * Z_2}{D_{\mathrm{u}}(p) - D_{\mathrm{l}}(p)} \tag{1}$$

The computation of the parametric domain is not trivial [3, 15]. In nearly all the published literatures [14–19], the parametric domain is obtained using the plane-sweep technique [20, 21]. Even though the plane-sweep algorithm can be efficient and has the complexity of $\mathrm{O}(n)\ln(n+s)$ where n is the number of contour edges and s is the number of intersections, it could still be computationally extensive, when the number of edges is large and the geometry is complicated. However, in the present algorithm, this computation is avoided and the parametric domain is identified by the classification of the triangles which result from the CDT of the polygons of the two sections in the parametric domain.

For most published methods, the tiling problem is addressed by the triangulation of the parametric domain. The mapping is then used to obtain the surface triangulation of the considered sections. To obtain a valid triangulation of the parametric domain, Barequet *et al.* [16] constructed a straight skeleton of the parametric domain and the dual triangulation is chosen as 'to be as monotone as possible with respect to the parametric domain edges'. Oliva *et al.* [3], constructed the Voronoi diagram and then subdivided each cell into triangles. However, as the well-developed Delaunay triangulation [22–26] provides the optimality of nearest connection, and this in turn favours vertical connection in the reconstruction, the problems of tiling and correspondence are naturally solved.

Finally, in order to deal with the branching problem, intermediate points are added to the edges of the generated triangles in the parametric domain when vertices of the edges are on similar contours. The addition of points guarantees that no planar triangles will exist after the mapping from the parametric domain to the physical domain.

3. DELAUNAY-BASED SURFACE RECONSTRUCTION ALGORITHM

The method proposed for the reconstruction of a three-dimensional image from a set of two-dimensional sections, is completely based on the use of the two-dimensional CDT.

3.1. Orientation of the contours

Contours extracted from a CT scan or an MRI scan will not, in general, obey any rules for orientation. Hence, in order to facilitate the computation of the parametric domain used in this method, a consistent orientation needs to be adopted. This procedure should be fully automatic and should be able to handle sections which contain multi-regions and multi-layers.

This can be achieved by using a constrained triangulation of the polygons forming the contours of the section, followed by grouping of the elements into regions whose boundaries are the contour polygons.

3.1.1. Constrained Delaunay construction. Given a two-dimensional domain with the boundary discretized into vertices and edges, the procedure for constructing the CDT, involves two stages:

1. Generate the unconstrained Delaunay triangulation of all vertices.
2. Perform the constrained recovery of the boundary edges.

The key part of the first step is the so-called incremental Delaunay insertion. Two approaches can be followed to obtain an updated Delaunay triangulation T_{new} after the insertion of a new point P into an existing Delaunay triangulation T_{old} [22, 25–27]. The approach adopted in the present work uses local splitting together with repeated edge swapping until all the edges, except constrained edges, satisfy the Delaunay property [25, 26].

After the insertion of all the boundary vertices, some boundary edges of the domain may not exist in the triangulation and the constrained boundary recovery must be performed to reconstruct the missing edges. This is implemented using a random edge swapping procedure [22, 23].

3.1.2. Identification of regions and polygon orientation. The triangulated domain is subdivided into regions bounded by one or more polygons. Starting from the triangles which have one edge on the external box which contains the triangulated domain, a greedy-type algorithm is used to mark all the adjacent elements without crossing any of the polygons. All the identified elements are marked as region one. The layer of adjacent elements to the marked elements can now be used as the starting elements to identify the next region. The process continues until all elements belong to one of the regions. This procedure is illustrated in Figure 2. Figure 2(a) shows the contours forming the section under consideration. Figure 2(b) shows the CDT of the four polygons forming the contours. The green elements shown in Figure 2(c) belong to region one. Figure 2(d) shows the various regions (coloured) which resulted from the completion of the above procedure.

The polygons forming the contours are given a marker which is taken to be equal to the minimum region number to which it is attached. The orientation is completed by assigning an counter-clockwise orientation to all the polygons with an odd marker while a clockwise orientation is assigned to polygons with an even marker. In addition, for each element in the

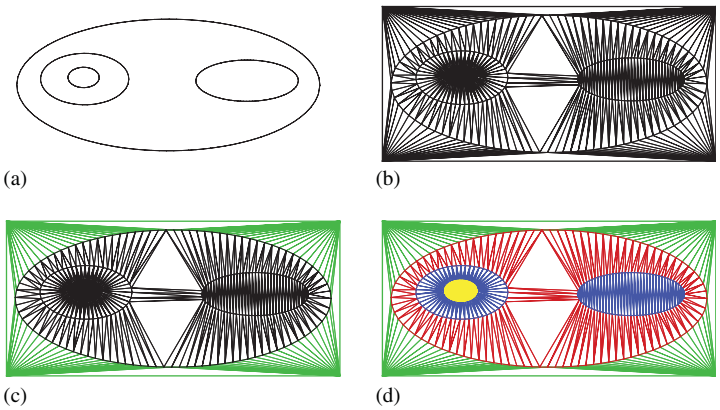

Figure 2. Polygon orientation and region identification: (a) the section formed by four contours; (b) constrained Delaunay triangulation; (c) the first region identified (coloured); and (d) all the regions are identified (coloured).

triangulation, the region to which the element belongs is stored. This information will be used later in the identification of the triangulation forming the parametric domain.

The above procedure needs to be carried out on the contours of the lower and upper sections under consideration. However, since at a later stage the Delaunay triangulation of each section is required in the reconstruction of the three-dimensional surface, the triangulation performed on the upper contours is stored and used in the reconstruction of the surface between the two adjacent sections.

3.2. Construction of the parametric domain

Traditionally, the use of a parametric domain to reconstruct the surface from two consecutive sections requires the identification of the domain, followed by its triangulation. The triangulation process is, generally, easy to obtain. However, the first step is more problematic and various complex algorithms have been used for this purpose. In the present method, we reverse this process. A single CDT of the contours of the lower and upper sections is performed first. A classification process is then carried out to directly extract the triangulated parametric domain.

3.2.1. Single constrained triangulation of the lower and upper contours. The Delaunay triangulation of the contours of the lower section which was constructed in order to establish the correct orientation of the polygons, is used as the starting point of this process. The edges of the polygons forming the lower contours are labelled as 1. The vertices of the upper contours are inserted into the CDT of the lower contours employing almost the same Delaunay insertion procedure as before. However, the edge swapping procedure, used to ensure that the final mesh is Delaunay compliant, is constrained to edges which are not part of the lower contours.

After the addition of the vertices that define the upper contours, the presence in the triangulation of all the edges forming the upper contours must be ensured. For a missing edge, the random edge swapping procedure is applied in the case where no intersection with an edge forming the lower contours exists. Figure 3 shows the recovery of edge AB which does not

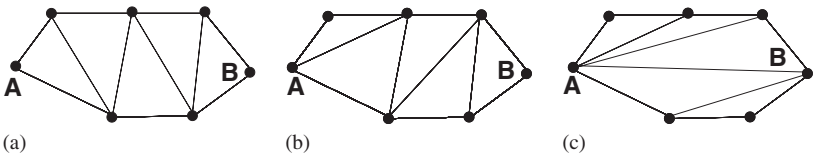

Figure 3. Boundary edge recovery: (a) AB is missing; and (b, c) recovery via random edge swapping.

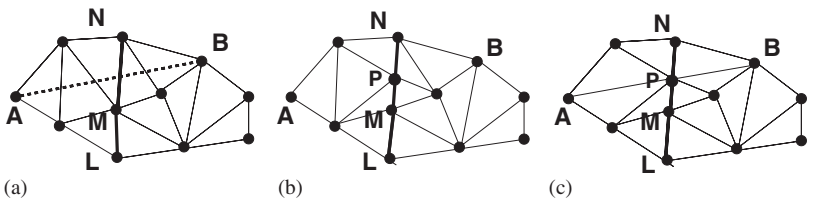

Figure 4. Conforming edge recovery: (a) AB is missing and LM, MN are boundary edges of the lower contour; (b) intersection points added; and (c) segments recovery.

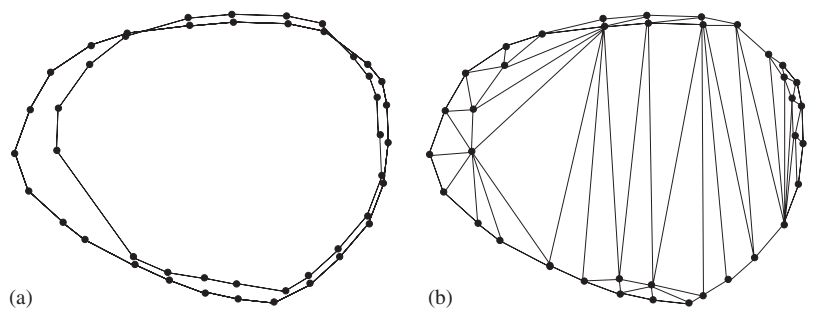

Figure 5. Small gap treatment: (a) two contours with minute gaps; and (b) Delaunay triangulation with mergings.

intersect any of the edges from the lower contours. In the case where a missing edge would intersect with one or more edges forming the lower contour, the intersection points are added in a Delaunay conforming manner, and the intersecting edges are subdivided into an appropriate number of segments. Recursive edge swapping is then performed for all non-boundary edges, and the missing edge is recovered, segment by segment, via the random edge swapping. All the edges of the upper contours are given the label equal to 2. Figure 4 shows the above procedure applied for the recovery of edge AB which intersects the lower contour edge MN. During the recovery procedure, all the intersection points added are marked as overlapping points.

The success of the above CDT depends mainly on the robustness of the edge recovery procedure. When a lower and an upper contour have a very similar topology, small gaps between the two contours will exist as shown in Figure 5(a). In this case, existing methods utilize one of two approaches [14–19]. The first approach perturbs the vertices of one of the

contours by a certain distance, while the second approach expands one of the two contours until the gap between them is large enough for the robustness requirement. However, both approaches require a choice of the direction for the perturbation of the vertices. A wrong selection may result in the enlargement of the gap in one place, with the reduction of a gap in another place. In the present work, a more systematic and robust method is used to overcome the problems which arise in the case of similar contours. The steps applied during the triangulation process are described in the following:

1. The parametric co-ordinate of a point on the upper contours $\mathbf{C}_u$ is taken to be identical to the parametric co-ordinate of an existing point on the lower contours $\mathbf{C}_l$ if the distance between the two corresponding points in the parametric domain is within a given tolerance.
2. A vertex on the upper contours $\mathbf{C}_u$ is projected onto an existing edge of the lower contours $\mathbf{C}_l$, in the parametric domain, if the distance is also within the prescribed tolerance.
3. During the boundary edge recovery of the upper contours, the vertices of the lower contours which lie within the given tolerance of the edges to be recovered, are added as intersection points.

It should be noted that all the above modifications to the co-ordinates will take place only in the parametric domain and, hence, the moved points will take their defined physical space co-ordinates when the mapping is completed.

After the above modifications of the point insertion and the edge recovery, no gaps smaller than the given tolerance will exist. This is illustrated in Figure 5(b). This modification is essential, in particular when the two contours are almost similar but defined by slightly different points. All the above operations are local to each of the displaced nodes and, hence, the CDT procedure maintains its efficiency.

3.2.2. Classification of subtriangulations. Having obtained a triangulation which contains all the edges forming the polygons of the lower and upper contours, it is now possible to use the orientation of the polygons to select the triangles which form the parametric domain. The union of the subtriangulations which lie inside the contours of one section and outside the contours of the other section will represent the triangulated parametric domain. The subtriangulation is achieved by marking the elements with respect to the lower contours and with respect to the upper contours independently. The aim of the marking is to distinguish between the elements lying inside and outside the polygons forming the contours of each section. The marking is carried out using a recursive neighbouring colouring technique. The data structure required for the above procedure includes element connectivity, edge connectivity, the elements neighbouring each element and the edges forming each element. In addition, edges which form the polygons of the contours should be marked as boundary edges with a marker equal to 1 for edges on the lower contours and marker equal to 2 for edges on the upper contours. Two vectors, LOWER$(1{:}N)$ and UPPER$(1{:}N)$, where N is the number of triangles in the mesh, are needed to complete the classification. The algorithm used for the classification of the triangulation is illustrated in Figure 6 and may be described as:

- Initialize the two vectors LOWER and UPPER to zero, i.e. LOWER$(i) =$ UPPER$(i) = 0$, $i = 1, N$ and initialize the number of elements in a stack to be zero.
- Loop over the edges which lie on the lower contours. Place the element, IL, which lies to left of the edge, in the stack and mark it with 1, i.e. LOWER(IL) $= 1$ and place the

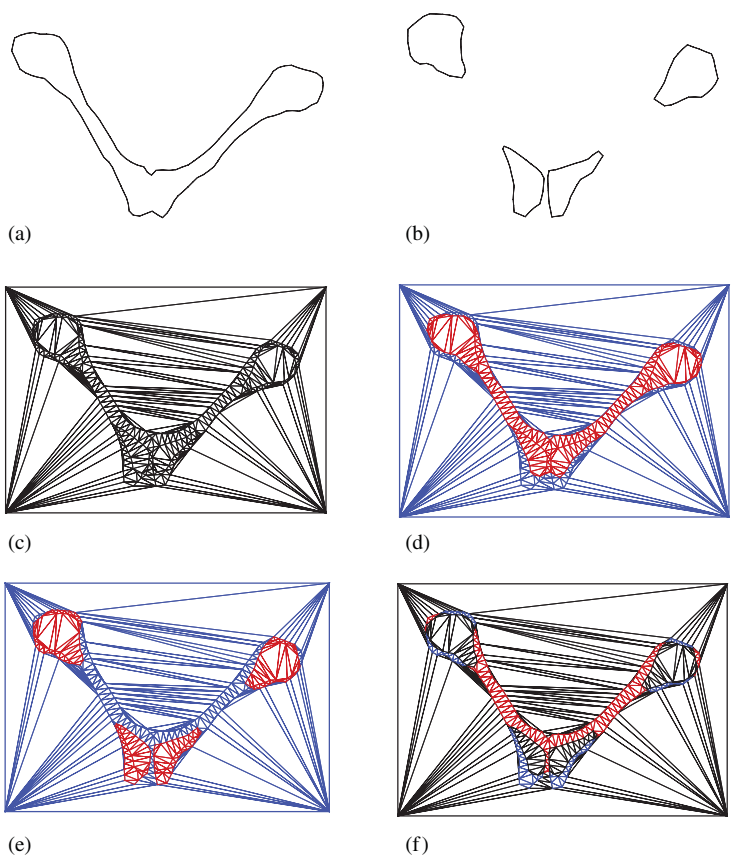

Figure 6. Parametric domain triangulation identification: (a, b) the lower and upper section; (c) constrained Delaunay triangulation of the two sections; (d) classification with respect to the lower contour: the green triangles are marked -1 and the red is 1; (e) classification with respect to the upper contour: green -2 and red is 2; and (f) the parametric domain triangulations defined by the red and blue parts marked as $(1, -2)$ and $(-1, 2)$, respectively.

element, IR, which lies to the right of the edge in the stack and mark it with -1, i.e. $\mathrm{LOWER(IR)} = -1$.

- For all elements in the stack, if any element edge is not a boundary edge, place the element which shares the edge in the stack and mark it with the same marker as the element under consideration.
- Select an unmarked element and check the location of its centroid in relation to the polygons forming the lower contours. If the centroid is inside the contours, the element is marked with 1, otherwise it is marked with -1. Add the element to the stack and continue the recursive colouring.
- Use the edges of the upper contours and repeat the above process to fill the vector $\mathrm{UPPER}(i)$ with 2 if the element i is inside the polygons forming the upper contours and -2 if the element i is outside the polygons forming the same contours.

- Select the triangles which have the LOWER marker equal to 1 and the UPPER marker equal to -2, i.e. inside the lower contours and outside the upper contours, or the LOWER marker equal to -1 and the UPPER marker equal the 2, i.e. outside the lower contours and inside the upper contours. This set of triangles defines the triangulation of the parametric domain.

To determine whether the centroid of an element lies inside or outside a polygon, we utilize the triangulation of the section which was carried out initially in order to decide the orientation of the polygons forming the contours of the section. The element from that mesh which contains the centroid of the element under consideration is identified. If the region of the identified element is marked as even, the point will be an external point while, an odd numbered region will indicate an internal point.

Since no intersection calculation is necessary, the CPU time required for the classification of the triangles and the parametric domain triangulations is also negligible and much less than an explicit parametric domain computation from contours which needs about $O(n)\ln(n+s)$ computations.

3.3. Intermediate points addition

In the parametric domain triangulation, the triangulation may have non-boundary edges which connect vertices of contours from the same section and which do not form a boundary edge. In this case, the mapping of the edge into three dimensions will produce triangles lying completely on either the lower or upper sections. This, generally, is unrealistic and will result in the failure of the method to deal with many features, such as branching. To overcome this limitation, extra points are introduced into the triangulation of the parametric domain. The new points are added on the middle of non-boundary edges which connect the same contours. The points are inserted in the same constrained manner utilized for the insertion of the boundary points [14, 15]. Mapping of the added points from the parametric domain to the physical domain is determined by the interpolation based on the equal importance criterion presented in Section 2.

3.4. Treatment of overlapping points and edges

For the construction of the parametric domain, it was necessary to insert the contours of one section into the Delaunay triangulation of the other section. To recover all boundary edges of both sections, subdivision of the intersecting edges is carried out, with the intersection points being added, in a constrained manner, into the Delaunay triangulation of the parametric domain. In addition, to avoid small gaps, points are snapped, in the parametric domain, to the closest edges. The mapping into the physical domain of these cases will result in the creation of holes in the three-dimensional surface mesh. Traditionally, the created holes are dealt with after the mapping process is completed. However, this technique is not trivial and requires an expensive procedure to ensure no surface self-intersection is taking place. In the present scheme a systematic and general method is developed. The method may be described as:

- Place two points on each intersecting edge in the parametric domain. One point, on each side of the intersection point, is added in a constrained Delaunay manner. The points are placed at a distance equal to 10% of the minimum edge length in the section under consideration.

- In the parametric domain, form two temporary triangles using the intersection point and two of the inserted points. For each triangle, the recovery of an edge made from two of the inserted points is required. This is achieved by using a random edge swapping technique.
- The physical co-ordinates of the first added point on the edge of the lower contours and the second added point on the edge of the upper contours are taken to be equal to the projection of the intersection point on the upper contours, while the physical co-ordinates of the other two points are taken to be the projection of the intersection point on the lower contours.
- Map the triangles, created in the parametric domain, into the physical domain and remove the two added triangles which will collapse after the mapping is completed.

A simplified version of the above process is demonstrated in Figure 7. Figure 7(a) shows the point P of the two intersecting edges, AB and MN, that form part of the lower and upper contours, respectively. The added nodes on the intersecting edges and the temporarily generated triangles are shown in Figure 7(b). Here, p_1 and q_1 represent the two points generated on the edge of the lower contours AB, while p_2 and q_2 represent the two points generated on the edge of the upper contour MN. Edges p_1p_2 and q_1q_2 are recovered using the random edge swapping technique. Finally, Figure 7(c) shows the triangulation in the physical domain where points p_1 and q_2 are mapped to the projection of the intersection point P onto the upper contours while the points p_2 and q_1 are mapped to the projection of the intersection point P onto the lower contours.

In the case of an overlapping edge, the intersection points are duplicated and two triangles with zero area are formed in the parametric domain. After the mapping into the physical domain, the two added triangles will have a non-zero area and they will fill the hole which would have resulted in the absence of this procedure. Figure 8 shows the treatment adopted in the case of overlapping edges.

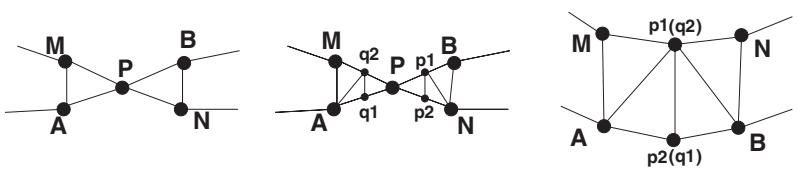

Figure 7. Overlapping point treatment: (a) contour intersection; (b) points are added; and (c) lifting up.

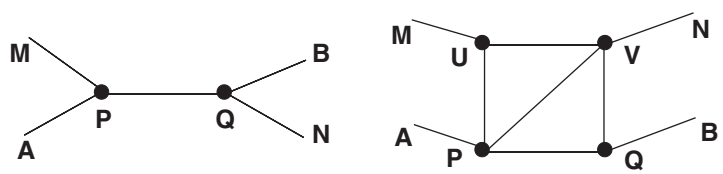

Figure 8. Overlapping edge treatment: (a) edge overlapping; and (b) two triangles generated.

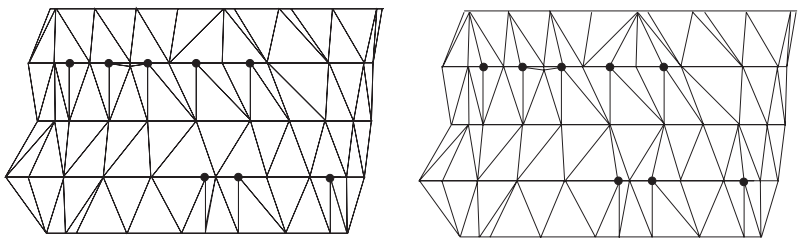

Figure 9. Adding up: (a) non-conformity due to hanging points; and (b) edges splitting.

3.5. Surface mapping and stacking of consecutive sections

As stated above, during the triangulation of the parametric domain of any two consecutive sections, points may need to be added to the contours of the lower and upper sections if intersections occur. These added points may not be required when the next two neighbouring sections are considered for surface reconstruction. This will lead to the appearance of hanging nodes when the reconstruction is completed. In order to ensure conformity, the mapped surface triangles of the upper or lower adjacent sections are subdivided in a consistent manner. Figure 9(a) shows a section of a reconstructed surface which contains hanging nodes. For conformity, the triangles which contain the hanging nodes are divided and new triangles are generated, as shown in Figure 9(b).

4. SURFACE SMOOTHING AND REMESHING

Most of the data from medical images, from which the sectional contours are derived, contain noise and so we need to smooth the above reconstructed surface triangulation to produce meshes of a quality appropriate for numerical simulation. An improved Laplacian smoothing method for a noisy surface is applied. Since this technique has been discussed in detail in Reference [28], only a brief explanation is given here. The modified smoothing algorithm first moves each vertex on a surface mesh to a new position which is the average of the locations of the neighbouring vertices; then, it pushes the modified vertex back towards a weighted average of its previous position and the original position. This modified Laplacian smoothing avoids the volume shrinkage.

After the surface smoothing, remeshing is applied to achieve a specified size or to ensure proper curvature representation [27, 29, 30]. Initially, surface features such as ridges and corners are correctly identified using a check on dihedral angles and a comparison of the ratio of triangle edge lengths.

The edges length required at each point can be controlled in various manners. A user specified size can be defined or a curvature-controlled metric can be used to determine the sizing required to achieve a prescribed accuracy for the surface representation. The main radii of the curvature are used to construct a geometric metric and this metric prescribes mesh sizes as well as element stretching directions at mesh vertices. For isotropic meshing, the largest radius is used. To achieve a smooth variation of the generated mesh, the above

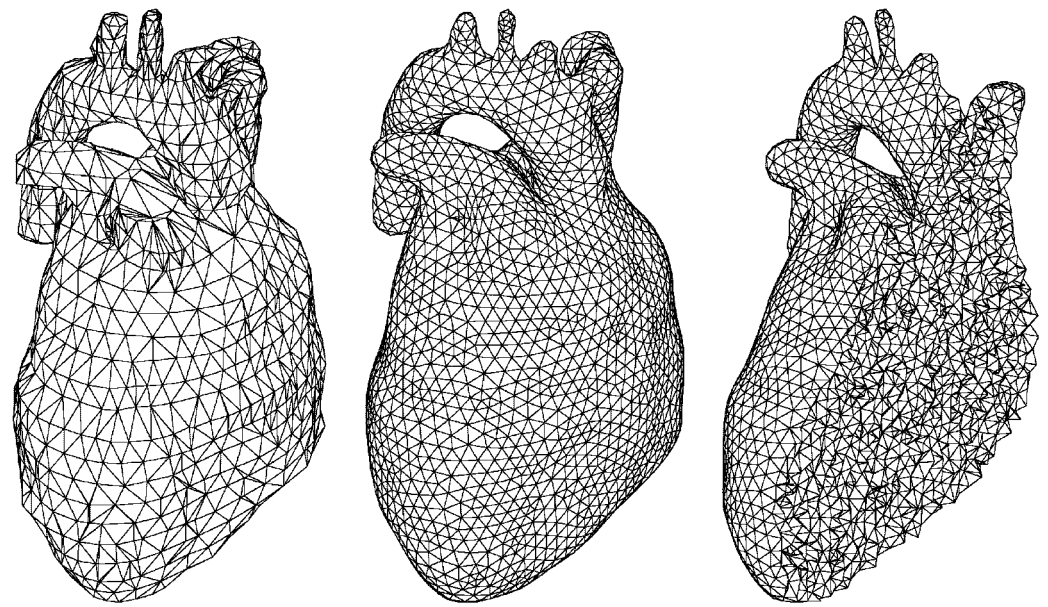

Figure 10. Geometry of a heart: (a) reconstructed surface triangulation; (b) final surface triangulation; and (c) cutting view of the tetrahedral mesh.

sizing given or computed from curvatures are graded using the H-correction procedure proposed in Reference [31]. This correction will reduce the ratio of neighbouring element edges length.

Using the geometric metric and the prescribed mesh size a normalized edge length is computed [29]. The normalize edge length is then used as a criterion for the refinement or coarsening of the given edge. Usually, the splitting criterion is set to be $\sqrt{2}$, and the contraction counterpart is set to be $1/\sqrt{2}$. Repeated edge swapping is performed after each splitting or coarsening operation in order to improve the quality of the adjacent elements. G1-interpolation is utilized for the placement of new nodes or the relocation of existing nodes. Finally, the surface triangulation is further optimized via a combination of edge swapping, point smoothing and node connectivity optimization or mesh relaxation [30].

5. NUMERICAL EXPERIMENTS

The proposed method was applied to reconstruct the geometry of various configurations starting from CT and MRI scans. To check the validity of the generated surfaces, a three-dimensional tetrahedral volume generator was utilized to fill the volume of the reconstructed domain [32]. The time required for the reconstruction process and for the mesh enhancement of the reconstructed mesh is given for all cases.

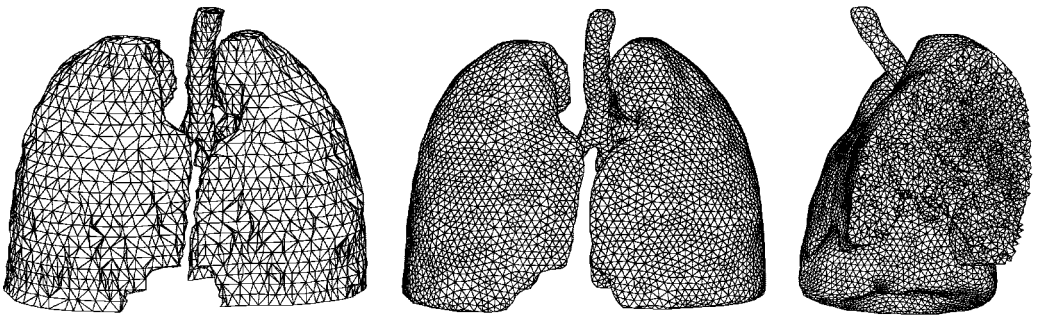

Figure 11. Geometry of a lung: (a) reconstructed surface triangulation; (b) final surface triangulation; and (c) cutting view of the tetrahedral mesh.

The first example considered is a model of the human heart. The input data consist of 29 sections with 1222 vertices in total. The surface reconstruction procedure required less than one CPU second to produce the surface mesh of 3618 triangles shown in Figure 10(a). The reconstructed surface was then enhanced to produce a final surface mesh of 5810 triangular elements, shown in Figure 10(b). The remeshing procedure required 6.5 CPU seconds to produce the required mesh size. The final enhanced surface was used to fill the domain with an unstructured tetrahedral mesh. Figure 10(c) shows a cut through the generated tetrahedral mesh.

The second geometry reconstructed was a human lung. In this case 32 sections containing 2629 vertices formed the scanned images. The reconstruction required 1.2 CPU seconds. The initial reconstructed surface triangulation consisted of 9128 triangles and the final enhanced surface contained 16 308 triangles and required 14.1 CPU seconds. The two surface triangulations are shown in Figure 11(a) and (b). A cut through the volume mesh generated from the enhanced surface triangulation is shown in Figure 11(c).

The third example considered is a female femur. The geometry consists of 87 planar cross sections with 12 269 vertices. 5.5 CPU seconds was required to complete the reconstruction. The reconstructed surface was then enhanced to produce a triangular mesh consisting of 29 072 elements in 21.7 CPU seconds. A view of the reconstructed and final enhanced external and internal surfaces are shown in Figure 12(a) and (b) and Figure 12(c) and (d), respectively. A cut through the tetrahedral mesh generated between the two surfaces is displayed in Figure 12(e).

Various other geometries also were considered to demonstrate the robustness and the efficiency of the developed technique. Surfaces were reconstructed from scanned images of a hip, a skull and a foot. The two surface triangulations of the hip are shown in Figure 13(a) and (b), and the final enhanced surfaces of the skull and the foot are shown in Figure 14. Here, for simplicity, the two reconstructed surface triangulations are omitted. Table I shows the number of sections and the number of vertices representing each geometry, the number of triangles after the surface reconstruction (R.S.T.), the number of triangles after the surface enhancement (F.S.T.), the CPU time for the surface reconstruction (Rec. CPU) and the CPU time (in seconds) for the surface enhancement (Enh. CPU).

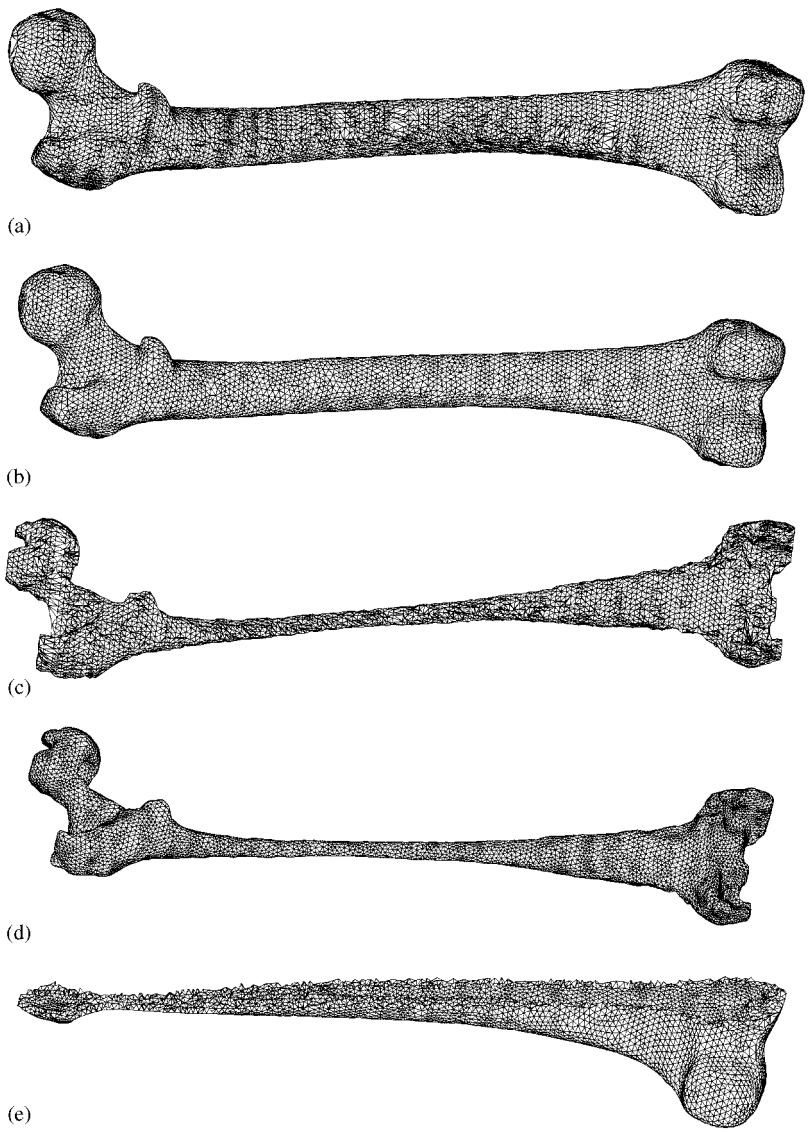

Figure 12. Geometry of a femur: (a) reconstructed surface triangulation of the outside; (b) final outside surface triangulation; (c) reconstructed surface triangulation of the inside; (d) final inside surface triangulation; and (e) cutting view of the tetrahedral mesh.

6. CONCLUSION

A new surface reconstruction algorithm with almost interactive efficiency has been proposed in this paper. All the required steps for the reconstruction were based on a consistent use of the constrained Delaunay triangulation. The correspondence, tiling and branching problems

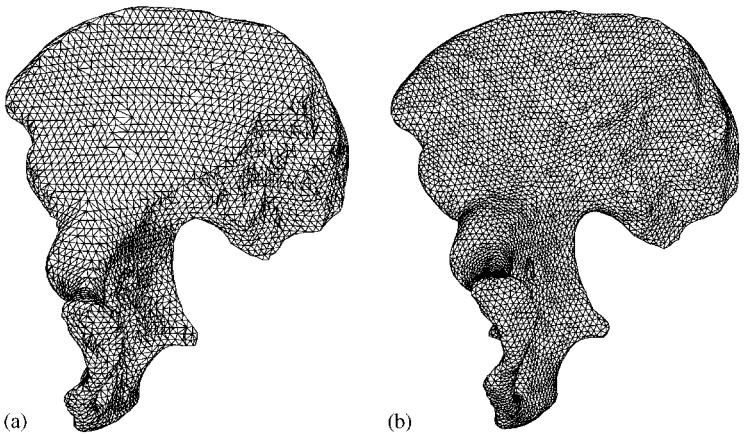

Figure 13. Geometry of a hip: (a) reconstructed surface triangulation; and (b) final surface triangulation.

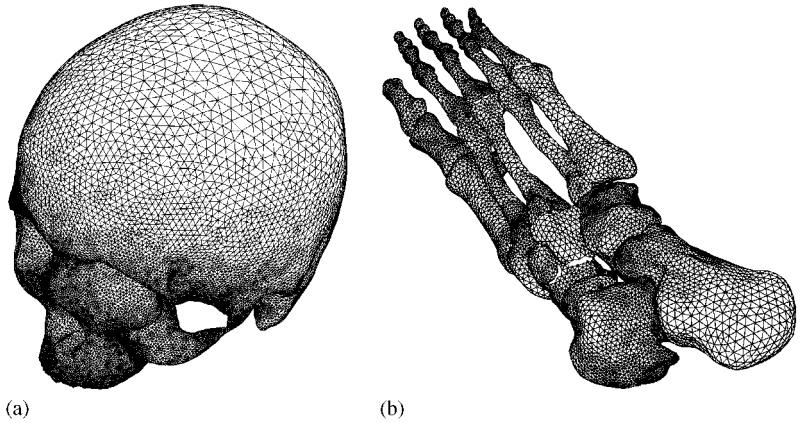

Figure 14. Final surface triangulations: (a) a skull; and (b) a foot.

Table I. Statistics of surface reconstruction examples.

Example	Sections	Cont. vertices	R.S.T. elements	F.S.T. elements	Rec. CPU	Enh. CPU
Hip	78	3123	8764	15516	1.8	10.6
Skull	133	8324	26411	97094	4.1	51.4
Foot	142	11527	36772	35544	5.2	24.3

are naturally solved. The parametric domain was directly extracted from the triangulation without the need for any explicit definition. Intermediate points were added and mapped to three dimensions using natural distance interpolation to avoid the appearance of any horizontal

triangles. Images of various human organs were reconstructed to demonstrate the robustness and efficiency of the proposed method.

REFERENCES

1. Bajaj C, Coyle E, Lin K. Arbitrary topology shape reconstruction from planar cross sections. *Graphical Models and Image Processing* 1996; **58**(6):524–543.
2. Wu Z, Sullivan JM. Multiple material marching cubes algorithm. *International Journal for Numerical Methods in Engineering* 2003; **58**(2):189–207.
3. Oliva JM, Perrin M, Coquillart S. 3D reconstruction of complex polyhedral shapes from contours using a simplified generalized Voronoi diagram. *Computer Graphics Forum* 1996; **15**(3):397–408.
4. Lin W, Chen S. A new surface interpolation technique for reconstructing 3D objects from serial cross-sections. *Computer Vision*, *Graphics and Image Processing* 1989; **48**:124–143.
5. Boyer E, Berger M-O. 3D surface reconstruction using occluding contours. *International Journal of Computer Vision* 1997; **22**(3):219–233.
6. Fuchs H, Kedem ZM, Uselton SP. Optimal surface reconstruction from planar contours. *Communications of the ACM* 1977; **20**(10):693–702.
7. Zyda M, Jones A, Hegan PG. Surface construction from planar contours. *Computers and Graphics* 1987; **11**: 393–408.
8. Jones MW, Chen M. A new approach to the construction of surfaces from contour data. *Computer Graphics Forum* 1994; **13**(3):75–84.
9. Keppel E. Approximating complex surfaces by triangulation of contour lines. *IBM Journal of Research Development* 1975; **19**:2–11.
10. Ganapathy S, Dennechy TG. A new general triangulation method for planar contours. *Computers and Graphics* 1982; **16**:69–75.
11. Boissonnat J. Shape reconstruction from planar cross sections. *Computer Vision*, *Graphics*, *and Image Processing* 1988; **44**(1):1–29.
12. Cheng SW, Dey TK. Improved construction of Delaunay based contour surfaces. *Proceedings of the ACM Symposium on Solid Modelling and Applications* 1999; **99**:322–323.
13. Meyes D, Skinner S, Sloan K. Surface from contours. *ACM Transactions on Graphics* 1992; **11**(3):228–258.
14. Cong G, Parvin B. Robust and efficient surface reconstruction from contours. *The Visual Computer* 2001; **17**: 199–208.
15. Barequet G, Sharir M. Piecewise-linear interpolation between polygonal slices. *Computer Vision and Image Understanding* 1996; **63**:251–272.
16. Barequet G, Goodrich MT, Levi-Steiner A, Steiner D. Straight-skeleton based contour interpolation. *Proceedings of the 14th Annual ACM—SIAM Symposium on Discrete Algorithms*, Baltimore, MD, 2003; 119–127.
17. Bereg S, Jiang M, Zhu B. Contour interpolation with bounded dihedral angles. *ACM Symposium on Solid Modeling and Applications* 2004; 303–308.
18. Surazhsky T, Surazhsky V, Barequet G, Tal A. Blending polygonal shapes with different topologies. *Computers and Graphics* 2001; **25**(1):29–39.
19. Hormann K, Spinello S, Schröder P. C^1-continuous terrain reconstruction from sparse contours. *Proceedings of Vision*, *Modeling and Visualization* 2003; 289–297.
20. Bentley JL, Ottmann T. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on Computers* 1979; 643–647.
21. Rourke JO. *Computational Geometry in C* (2nd edn). Cambridge University Press: Cambridge, 1998; 264–266.
22. George PL, Borouchaki H. *Delaunay Triangulation and Meshing. Application to Finite Elements Methods*. Hermès: Paris, 1998.
23. George PL, Hecht F, Saltel E. Automatic mesh generator with specified boundary. *Computer Methods in Applied Mechanics and Engineering* 1991; **92**:269–288.
24. Bowyer A. Computing Dirichlet tessellations. *The Computer Journal* 1981; **24**:162–166.
25. Marcum DL, Weatherill NP. Unstructured grid generation using iterative point insertion and local reconnection. *AIAA Journal* 1995; **33**:1619–1625.
26. Lawson CL. Properties of n-dimensional triangulations. *Computer Aided Geometric Design* 1986; **3**:231–246.

27. Thompson J, Soni BK, Weatherill NP. *Handbook of Grid Generation*. CRC Press LLC: Boca Raton, FL, 1999.
28. Vollmer J, Mencl R, Müller H. Improved Laplacian smoothing of noisy surface meshes. *Computer Graphics Forum* 1999; **18**(3):131–138.
29. Frey PJ. About surface remeshing. *9th International Meshing Roundtable* 2000; 123–156.
30. Wang D, Hassan O, Morgan K, Weatherill NP. Enhanced remeshing from STL files and its application to surface grid generation. *Communications in Numerical Methods in Engineering*, 2005, submitted.
31. Borouchaki H, Hecht F, Frey PJ. Mesh gradation control. *International Journal for Numerical Methods in Engineering* 1998; **43**(6):1143–1157.
32. Weatherill N, Hassan O. Efficient three dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints. *International Journal for Numerical Methods in Engineering* 1994; **37**: 2005–2039.

