Proceedings of the ASME 2010 3rd Joint US-European Fluids Engineering Summer Meeting and 8th International Conference on Nanochannels, Microchannels, and Minichannels
FEDSM-ICNMM2010
August 1-5, 2010, Montreal, Canada

FEDSM-ICNMM2010-30951

FINITE ELEMENT-BOUNDARY ELEMENT MESH GENERATION TECHNIQUE FOR FLUID STRUCTURE PROBLEMS

Hamid Sharifi[1]
[1] **Department of Mechanical Engineering Shiraz University, Shiraz, Iran, hsharifi@shirazu.ac.ir**

Abstract

In this article an approach for fully automatic mesh generation for two dimensional fluid structure problems that respect the integrity of the geometrical boundaries is presented. This approach is based on the modified Quadtree method. First, interior quadrants of the solid structure are created as the original Quadtree method. Boundary quadrants of the solid structure are created between the boundary curves and the interior quadrants using a simple projection algorithm. As a result, the problem of cut quadrants of the original modified Quadtree method is eliminated here. Boundary elements of the fluid region are created on the boundary curves using calculated projection points. The use of closed non uniform composite B-spline curves, for a unified representation of boundaries curves, simplifies the projection algorithm. On the other hand using this type of boundaries representation reduces geometrical incompatibilities of the generated mesh and produces a perfect compatibility between boundary elements and finite elements. This method can be extended to problems of three dimensional mesh generation and eliminate all cases of cut octants. An object-oriented prototype program in C++ has been written and application example is presented in this paper. Several algorithms of this method are suitable for an implementation on parallel computers.

Introduction

The coupled finite element and boundary element method are used more and more in fluid-structure interaction. Such method may be used for a solid domain (finite element domain) with geometrical complexity. To have a reliable application of this technique a valid finite element discretization of the solid domain and a compatible boundary element discretization of the solid-fluid interface are needed. In these types of problems, a fully automatic mesh generation is a device which produces automatically a suitable finite element mesh from the geometrical representation of the solid and a compatible boundary element mesh of the solid-fluid interface. These meshes must be topologically compatible and geometrically similar to the real geometrical model [1-3].

Different approaches have been developed for two and three dimensional fully automatic mesh generation such as the Delaunay triangulation technique [1,3-7], the isoparametric mapping mesh generation [8-12] and modified quadtree/octree methods [2,13-20]. The richness of the tree based data structure and the spatially based mesh control devices of the modified quadtree/octree mesh generator make it one of remarkable techniques for the adaptive finite element analysis using *h* or *hp* type mesh improvement [15,21-23].

The principals of the quadtree method for finite element mesh generation can be summarized as follow: first we must choose a square that encloses the entire solid object. This square is then recursively subdivided into four squares. Each square is tested to see if it is inside the solid (IN), outside the solid (OUT) or partially inside the solid (PARTIAL). Partial quadrants are classified as being either edge or vertex quadrant [13, 19] and then recursively subdivided in the same manner. This process is continued until the criteria of the refinement which define by user is satisfied. The interior quadrants are then subdivided to be compatible with their neighboring interior and partial quadrants. Partial quadrants are cut using some methods so that they closely match the representation of boundary curves (CUT quadrants). Finally, solid finite elements are generated in all quadrants [2, 13-15]. The basic algorithms will be found in [13, 19].

In this work for fluid-structure problems, an automatic mesh generator for two dimensional problems based on the quadtree decomposition technique has been developed. An important difference with the original method is that in place of dealing with cut quadrants as in [2, 13-15, 19], after the quadtree subdivision process, we keep interior quadrants and we eliminate OUT and PARTIAL quadrants. New solid boundary quadrants are created between interface solid-fluid interface curves and free edges of interior quadrants. A simple projection method based on finding the minimum distance between a point and a curve is used. For each free edge of the

interior quadrants one solid boundary quadrant is created. In this way all of different configurations of cut quadrant which were discussed earlier [13-14, 19] are eliminated.

With the use of closed nonuniform composite rational B-spline curve, we can produce a unique mathematical representation of solid-fluid interface curves. This type of representation simplifies solid boundary quadrants generation. After the creation of all solid quadrants in the interior and on the boundary, the shapes of generated solid quadrants can be improved using a repositioning algorithm. Although the spatial addressability of quadrants as discussed in [24-25] are lost using the smoothing operation but, each quadrant can be found easily using the quadtree data structure.

After the generation of the solid elements, boundary elements can be generated easily. Each solid boundary quadrant has a side on the interface curves which is made by two projection points. This side, that cut a segment of the interface curve, defines a boundary element having two nodes in common with its neighbor finite element.

This method can be used for three dimensional problems using octree method by creating solid boundary octants between interior octants and the solid-fluid interface surface. As a result, all of catalogued 4096 different configurations of cut octants are replaced by a simple routine [26].

Geometric representation of the solid-fluid interface

Every mesh generator requires a good geometric representation of the domain, both at a geometry level, i.e. "the mathematical representation of the geometric model in terms of shape and space location", and at a topology level, i.e. "the adjacency relationship of various geometric entities among each other", [6]. The composite nonuniform rational B-spline curves is a versatile curve model that can represent almost any two dimensional domain curves [27-28]. The B-spline curve is a piecewise polynomial function which can be viewed as a generalized polynomial that may have derivative discontinuities at some points. A composite B-Spline representation of boundary curves of a domain can be obtained as defined in [16, 29]:

The composite B-spline curve presents second order (C^2) continuity between adjacent segments of a smooth curve. However, if we use the double repeated knot we obtain C^1 continuity and if we use the triple repeated knot, we obtain C^0 continuity, i.e. continuity of the function only [30-31]. As a result, a boundary curve having C^0 continuity can also be modeled using only this type of the mathematical model. So, a closed composite nonuniform cubic B-spline curve is almost adequate for representing any boundary curve in two dimensional mesh generation problems. We can conclude that any two dimensional object with any geometric complexity can be approximated satisfactorily by one or by a finite number of cubic composite B-Spline curves. This unified representation of the geometry has the advantage of representing straight, quadric and cubic lines by the same third degree polynomial model. This makes the mesh generation algorithm and programming straightforward. As a result, the problem of the boundary modeling mentioned in [3,32] presents no special difficulties.

The domain can be described by three different types of points as follows:

I. Points of the first type are boundary simple points which correspond to points on smooth curves, i.e. without derivative discontinuity.
II. Points of the second type are boundary corner points. At these points, the boundary has the derivative discontinuity of the C^0 order[1].
III. Points of the third type are points inside the solid which are used to define the mesh control criteria inside the domain, i.e. if local refinement of quadrants at a region is needed we introduce these points to define mesh refinement parameter at that region.

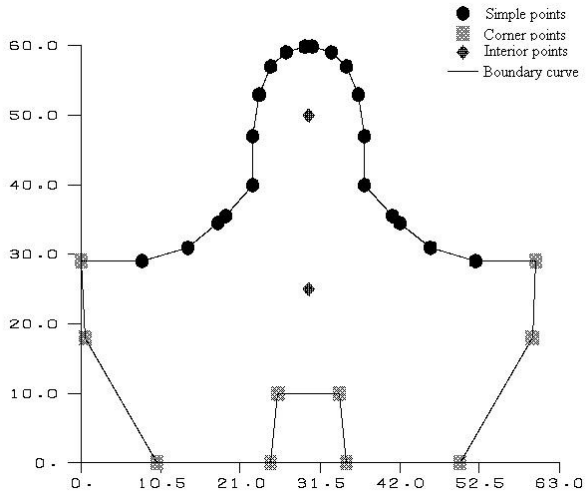

Figure 1: First example of control points and B-spline boundary representation.

The same as the third type, the first and the second type point can also have a mesh control parameter. Mesh control

[1] The second type or corner points are tripled before they are used for finding the B-spline representation of boundary.

parameters at any point on the boundary curve can be calculated by the B-spline approximation of control points on the boundary, or any suitable approximation method.
Figure 1 represent control points and B-spline representation of the solid boundary curve (solid-fluid interface curve) for an example of a mechanical part in interaction with the fluid around it. Figure 2 shows another example.

Quadrant generation

As mentioned before, first a square that can surround the entire solid object must be created. This square is the root of the quadtree. Such a square is subdivided into four quadrants. Each quadrant is then checked to know if it is inside the object (IN), outside (OUT) or partially inside the solid object (PARTIAL). Quadrants are marked as IN or OUT and the PARTIAL quadrants are subdivided further into four quadrants which are then checked and classified in the same way. This process is continued until the object is refined to a degree which is requested by user [2, 13-15].

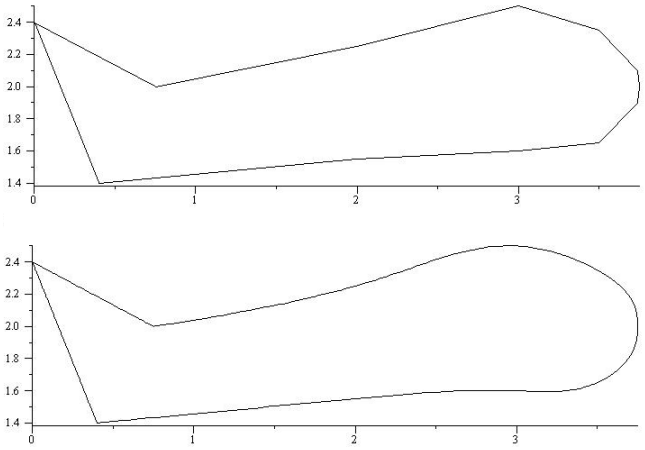

Figure 2: Second example of control points and B-spline boundary representation.

We have used a modified version which is similar to the original method. In version the root quadrant is checked and subdivided into four quadrants as follows:

a) If the greatest mesh control parameter inside the quadrant is greater than the actual level of the quadrant plus one the quadrant will be subdivided into four quadrants and each of the subdivided quadrant will be checked in the same way.
b) If the greatest mesh control parameter inside the quadrant is equal to or less than actual level of the quadrant plus one, the quadrant will be subdivided into four quadrants and the process will be terminated.

The above procedure can be presented in the following recursive algorithm:

Algorithm:check_and_subdivide_quadrant(In_quadrants, Level)

```
Begin
    If Maximum_control_parameter > (Level + 1)
        Then Divide_quadrant_into_four;

        For i   in 1..4 loop

            check_and_subdivision_quadrant
            (quadrants(i), Level+ 1);
        Endloop;
    Else
        Divide_quadrant_into_four;
    Endif;
End;
```

This process is similar to the In/Out algorithm and the recursive subdivision of the partial quadrants of the original method, as most of mesh control parameter are found on the boundary curve. Nevertheless this method produces much more uniform quadrants near the solid boundary curve and mesh control points. As a result, quadrants ***near*** the solid-fluid interface are at the same level of subdivision as requested by the user for the solid-fluid interface curve. Figure 3 shows the first step of the quadrants generation for the second example. It must be mentioned that the above algorithm (check and subdivide quadrant) is suitable for implementation on parallel computer. One processor can subdivide a quadrant and send them to four processors for further refinement.

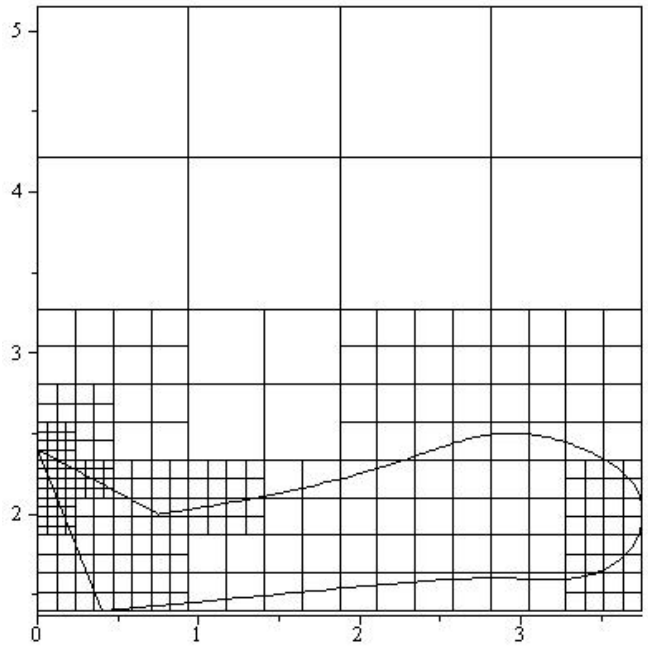

Figure 3: First step of quadrants generation.

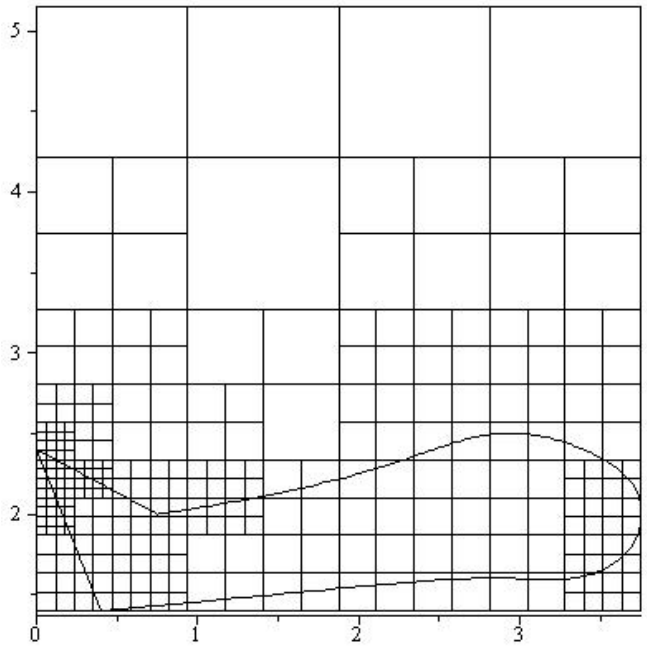

Figure 4: Generated quadrants after one level difference process.

In the above a quadtree data structure is constructed. We will modify and use this tree in the following steps. For smooth transitions between quadrants of different subdivision level, as the original quadtree method the one level difference rule [2, 13-15] is applied. This rule forces that the maximum level difference between any two neighboring quadrants are not greater than one. As a result, if two quadrants have a difference level greater than one the greater quadrant is subdivided recursively until the one level rule is achieved. This process guarantees acceptable aspect ratio of triangular finite elements which will be created later. Figure 4 shows the result of this process for the second example.

In the next step the interior quadrants are separated from other quadrants. This is done by marking the partial quadrant and then using an In/Out test method such as the method of "inflatable balloon". As partial quadrants intersect boundary curves they can be detected with a simple intersection algorithm. Figure 5 shows interior quadrants for the second example. For the rest of the mesh generation procedure we need only interior quadrants as a result other quadrants can be removed from the tree.

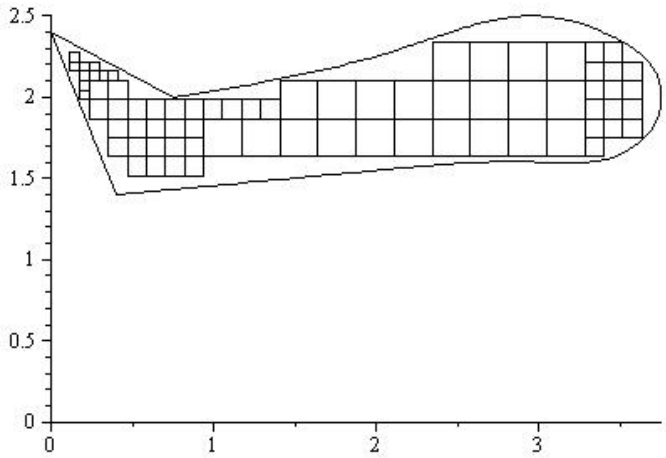

Figure 5: Interior quadrants are selected.

Generation of boundary quadrants

The solid domain decomposition process must generate elements which can representation the solid boundary adequately. A boundary quadrant must be endowed with information describing the boundary curve such as: the type of the curve, intersection points, boundary edges or vertices embed in the quadrant and so on. Even if the boundary curve is well represented, some technical difficulties are still possible. For example we can have:

- Very small or tiny quadrants
- Complex boundary of cut quadrants (due to the complexity of the boundary curve inside the original partial quadrant).

To circumvent these difficulties, we present a new approach for dealing with boundary quadrants. Instead of using partial quadrants generated by the In/Out test during the quadrant generation; we discard them and we use information of interior quadrants (IN) and boundary curves to generate new boundary quadrants.

In the first step, we create a new boundary quadrant between each free edge of interior quadrants and boundary curves. The procedure is as follow:

For a given IN quadrant as shown in Figure 6, take Q_1 and Q_2 as vertices of a free segment L. We will find points P_1 and P_2 on the boundary curve located at minimum distance to points Q_1 and Q_2 respectively. Note that the lines (Q_1, P_1) and (Q_2, P_2) must not intersect any other quadrant. A new boundary quadrant is defined by the segment L, lines (Q_l, P_1) and (Q_2, P_2) and the boundary segment (P_1, P_2).

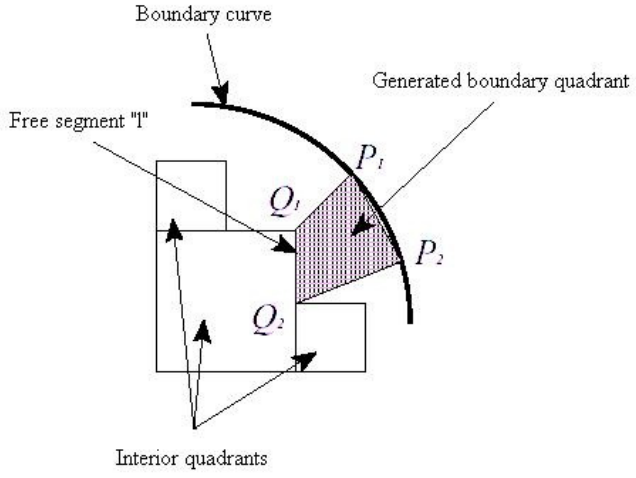

Figure 6: Generated boundary quadrant between the boundary curve and the free edge segment of interior quadrant "l".

The minimum distance between a point Q (in the domain) and a parametric curve P(u) (the boundary curve) can be find as follows: we find a vector (P-Q) such that P belong to the curve P(u) and the vector (P-Q) is perpendicular to the tangent $P^u(u)$ at P. In other word, we must find the point $P \in P(u)$ that satisfy the following equation:

$$(P-Q).P^u = 0. \qquad (1)$$

The equation (1) can be solved using the Newton Raphson method [31].

As we used unified representation and our boundary curve is closed, the answer is not unique and we need to select an acceptable one. We may have more than one occurrence of normal in the domain. If this is the case, all normals must be computed and the shortest (P-Q) vector must be selected as the answer. It is possible that this shortest vector passes through an existing quadrant. If it is the case, we choose another normal vector and if all of normal vectors run into existing quadrants, we start from the closest point from Equation (1) and we move on the boundary curve step by step until we find a point which does not intersect other quadrants. It is also possible that we create a quadrant with zero area, i.e. if points P_1, P_2, Q_1 and Q_2 are collinear. This situation can be handled by displacing point P_1 or P_2 a little or we can treat it later in the smoothing process. In the smoothing process this pathological case will be entirely eliminated automatically without any special consideration. If the point P_1 is coincident with the point P_2, we have a triangular shape in place of the quadrilateral one. This case needs a little attention during the mesh generation process to do not try to divide it to two triangles. Figure 7 shows generated boundary quadrants together with interior quadrants after the first step of boundary quadrant generation.

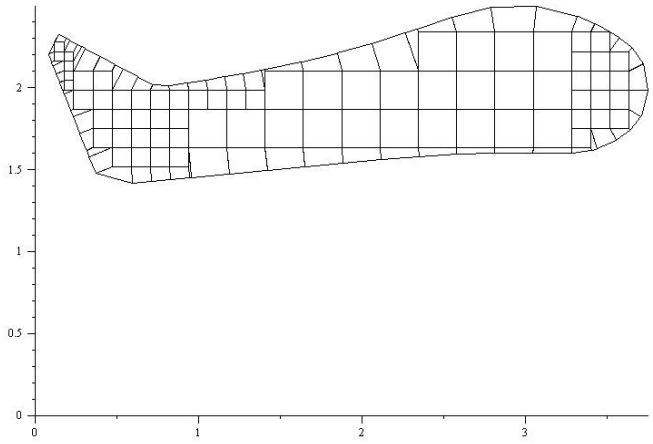

Figure 7: Interior and boundary quadrants after the first step of boundary quadrant generation.

As one can see in Figure 7, these boundary quadrants dose not present a good geometric similarity at corners compare to the B-spline representation of the object. To have an acceptable match with the geometrical model and to add corner points into the model, another operation is needed. In the second step of boundary quadrant generation, nodes of boundary quadrants which are on boundary curves near corner points are pulled to corner points. As we keep the list of corner points in our database this process is easy to execute. Such process guarantees the geometric similarity of the generated model. Figure 8 shows generated quadrants after this process.

Smoothing process

As mentioned in the previous section, it is possible that some of quadrants do not have satisfactory shape. For example there is a considerable sharp transition between adjacent quadrants with different levels of subdivision. To moderate these situations, a smoothing operation may be used after the generation of all quadrants. In this process, nodes of quadrants are repositioned to improve quadrants shape.

One of the widely used repositioning methods in quadtree technique is the Laplacin smoothing method [12, 15, 19]. In this method, the Laplace equation must be satisfied for interior nodes. Here, a given interior node is moved to the centroid of its neighboring nodes. This process can be used several times. In our implemented software, boundary nodes remain at their original locations as they can only move on boundary curves. Generally four iterations are recommended to have a reasonable convergence [13]. Figure 9 shows generated quadrants after applying the smoothing process. This process can also be used after the solid finite element mesh generation process to improve shape of elements.

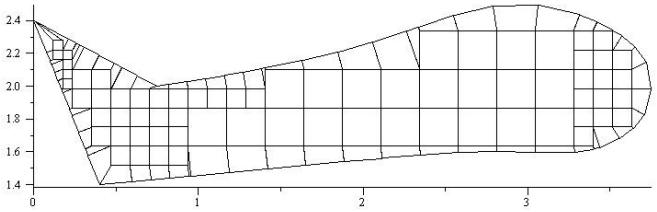

Figure 8: Generated quadrants after pulling to corner process.

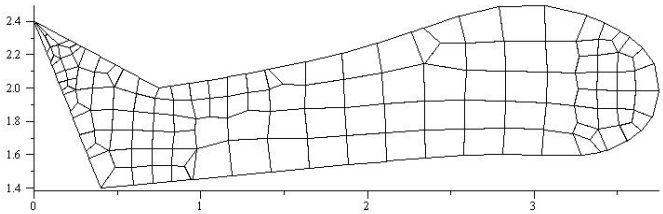

Figure 9: Generated quadrants after smoothing process.

Boundary element mesh generation

If we consider boundary quadrants, we will realize that these are the only quadrants which have a side on common with the solid-fluid interface. They are shown in Figure 10. These common sides have been made from the projection point on the solid-fluid interface curves. So, we have all the information about these sides. These sides make boundary elements of the fluid in interaction with the solid. These boundary elements have been shown in Figure 11.

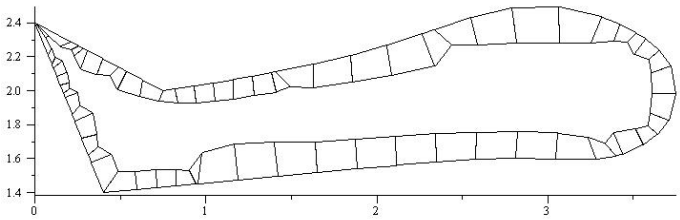

Figure 10: Boundary quadrants.

These boundary elements have common nodes with adjacent solid finite elements which we will create in the next section. As result, there will be no incompatibility between boundary and finite elements.

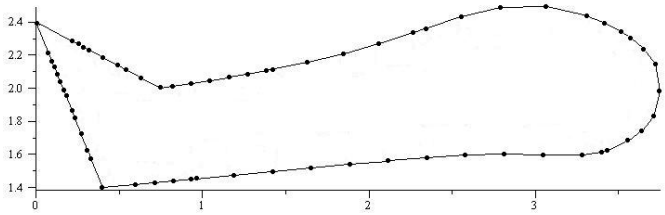

Figure 11: Boundary elements.

Finite element mesh generation

Depends on our finite element package and on their analysis capabilities; conforming elements versus non uniform (nonconforming) elements, possibility of error estimations based on h, p or h-p formation, we can generate a finite element mesh tailored to our solid object. Both triangular and quadrilateral elements can be generated.

If triangular elements are desired, we can use simple triangulation algorithm. It is easier to use predefined templates that match various quadrant configurations. In our case, only six templates are required for the triangulation of interior quadrants. The same templates can be used for the triangulation of boundary quadrants; nevertheless, we usually need only one template here. (The special case of triangular shape boundary quadrant does not need any template.) Generated quadrants after the smoothing process can also be used for the generation of mixed triangular/quadrilateral elements. Note that some interior quadrants have difference level of subdivision with respect to their neighbors. These quadrants have more than four edges (see Figure 9). They should be subdivided using appropriate template to generate quadrilateral or triangular element.

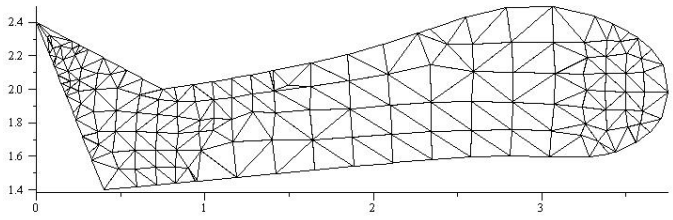

Figure 12: Generated solid mesh.

Figure 12 presents our second example after the triangulation process. Further smoothing processes can be used, if necessary, to produce more uniformity between elements.

General algorithm

The overall algorithm of the mesh generation procedure can be written as follows:

1- Enter geometric data and mesh parameters.
2- Generate the geometric model using NURB [26] technique.
3- Generate quadrants.
   3-1 Perform recursive subdivision.
   3-2 Enforce one level difference.
   3-3 Find interior quadrants.
   3-4 Generate boundary quadrants.
4- Perform the smoothing process.
5- Perform boundary element mesh generation.
6- Perform the finite element mesh generation.
7- Produce output results.

Conclusion

The modified quadtree method is a well suited technique for fully automatic mesh generation in two dimensional solid-fluid interaction problems. This method can be improved by paying special attention to the boundary mesh generation. In this article, we have examined a method for improving the treatment of boundary quadrant generation. Solid-fluid interface curves are approximated by closed composite cubic B-spline curves in order to obtain a unified geometric representation. Interior quadrants are generated using mesh control parameter on the boundary interface and inside the domain. In our method we have the same level of subdivision between partial quadrants and adjacent IN quadrants. OUT and partial quadrants are eliminated and boundary quadrants are generated between free edges of interior quadrants and boundary interface curves. Therefore, all special treatments for dealing with different configurations of cut quadrants are eliminated. Boundary quadrants are generated by using a simple algorithm which is based on the minimum distance between a point inside the domain and boundary curves. As a consequence, boundary quadrants are easily generated and have satisfactory geometric shapes which are suitable for the boundary element and the finite element mesh generation. Boundary elements are created from the common side of the boundary quadrant and interface curves. Finite elements are created from discretization of interior and boundary quadrants according to some predefine templates. This approach for handling boundary quadrants can be extended to three dimensional problems and can condense out all of 4096 different configurations of cut octants [26, 33].

References

[1] Schroeder W.J., Shephard M.S., "Geometry based fully automatic mesh generation and Delaunay triangulation" Int. j. numer. methods eng., 26, pp2503-2515 (1988).

[2] Shephard M.S. and Georges M.K. "Automatic three dimensional mesh generation by the finite octree technique" Int. j. numer. methods eng., 32, pp709-749 (1991).

[3] Weatherhill N.P. "The integrity of geometrical boundaries in two dimensional Delaunay triangulation" Comp. in appl. numer. meth. , 6, pp101-109 (1990).

[4] Cavendish J.C., Field D.A. and. Freg W.H, "An approach to automatic three dimensional finite element mesh generation" Int. j. numer. methods eng., 21, pp329-347 (1985).

[5] Sheton D.N. and Cendes Z.J., "Three dimensional finite element mesh generation using Delaunay tessellation" IEEE Trans. magnetics, MAG21, pp2535-2538 (1985).

[6] Schroeder W.J., Shephard M.S., "A combined Octree/Delaunay method for fully automatic 3D mesh generation" Int.j. numer. methods eng., 29,pp37-55 (1990).

[7] George P.L., Hermellne F. "Delaunay's mesh of a convex polyhedron in dimensional d application to arbitrary polyhedral" Int. j. numer. methods eng. , 33, pp975-995 (1992).

[8] Durocher L.L., "A versatile two dimensional mesh generator with bandwidth reduction" Computer & structures, 10, pp561-575 (1979).

[9] Zienklewicz O.C., Philips D.V. "An automatic mesh generation scheme for plane and curved surface by isoparametric coordinates" Int. j. numer. methods eng., 3,pp 519-528 (1971).

[10] Kadivar M.H., Sharifi H. "A new versatile two dimensional mesh generation" Proceeding of the third int. conference on advances in numer. methods eng. theory and application, Swansea, U.K., pp1090-1096 (Jan.1990).

[11] Kadivar M.H., Sharifi H. "Mesh generation of contacted objects" proceeding of the int. conference on control and modeling, ICCM 90, Tehran, Iran, pp89-93 (July 1990).

[12] Carter L., Wellford J.R.and Gorman M.R. "A finite element transitional mesh generation procedure using sweeping function" Int. j. numer. methods eng. , 26, pp2623-2643 (1988).

[13] Bachman P.L., Wittchen S.L, Shephard M.S., Grice K.R. and Yerry M.A. "Robust geometrically based automatic two dimensional mesh generation" Int. j. numer. methods eng. , 24, pp1043-1078 (1987).

[14] Yerry M.A. and Shephard M.S. "Automatic three dimensional mesh generation by the modified octree technique" Int. j. numer. methods eng. , 20, pp1965-1990 (1984).

[15] Shephard M.S., Yerry M.A. and Bachman P.L. "Automatic mesh generation allowing for efficient a priori and posteriori mesh refinement" Computer methods in applied mechanics and eng., 55, pp161-

180(1986).
[16] Shephard M.S. and Georges M.K. “Reliability of automatic 3D generation” Computer methods in applied mechanics and eng. pp443-462 (1992).
[17] Jung Y.H. and Lee K. “Tetrahedron based octree encoding for automatic mesh generation” Computer aided design, 25, 3, pp141-153 (1993).
[18] Buratynski G.K. “A fully automatic three dimensional mesh generation for complex geometries” Int. j. numer. methods eng. , 30, pp931-952 (1990).
[19] Richet N., Gakwaya A. “Automatization de maillage pour element finis par la methode de quadtree” Rap. tech. No: 2708/1988 lab. CAO Dep. genie mecanique, U. Laval (1988).
[20] Carey G.F., Sharma F. and Wang K.C. “A class of data structures for 2D and 3D adaptive meshing” Int. j. numer. methods eng. , 26, pp2607-2622 (1988).
[21] Tworzydlo W.W., Oden J.T. “Towards an automatics environment in computational mechanics” Comp. meth. in app. mech. & eng., 104, pp87-143 (1992).
[22] Fischer A. & Bar-Yoseph P. Z. “Adaptive mesh generation based on multi-resolution quadtree representation” Int. j. numer. methods eng. , v. 48, iss.11, pp1571-1582 (2000).
[23] Loic M., “A New Approach to Octree-Based Hexahedral Meshing” Proceedings, 10th Intern. Meshing Roundtable, Sandia National Lab., pp209-221, Oct. 7-10 (2001)
[24] Kela A., Saxena M. and Perruchio R. “A Hierarchical structure for Automatic meshing and adaptive FEM analyses” Eng. Comput. , 4, pp104-112 (jun.1987).
[25] Perruchio R., Saxena M. Kela A. “automatic mesh generation from solid models based in recursive spatial decomposition” Int. j. numer. methods eng. , 28, pp2409-2501 (1989).
[26] Sharifi H. "Maillage automatique par la méthode de quadtree/octree modifiée", Thèse présentée à la Faculté des études supérieures de l’Université Laval, Sainte-Foy, Québec, (1995).
[27] Choi B.K., Yoo W.S. and Lee C.S. “Matrix representation for NURB curves and surfaces” Computer aided design, 22, 4, pp235-240 (1990).
[28] Farin G. “Courbe et surfaces pour la C.G.A.O. ” Masson Paris (1992).
[29] Choi B.K. “Surface modeling for CAD/CAM” Elsevier (1991).
[30] Cabral J.J.S.P., Wrobel L.C. and Brebbia C.A. “BEM using B-Spline in Boundary elements in mechanical and electrical engineering.” Springer Valag (1990).
[31] Batels R.H., Beatty J.C. and Barsky B.A. “An introduction to splines for use in computer graphics and geometric modeling” Morgan, Kanfinenn, Les Altos, California (1987).
[32] Kela A. “Hierarchical octree approximations for boundary representation based geometric models” Computer aided design, 21, 6, pp335-362 (1989).
[33] Lescoulie C., Gakwaya A. “Mailleur Automatique pour des solides 3D a' facettes planes: EMMATOM3” Rap. Tech. No 10/02/1990, Lab. de CA.O. , département de génie mécanique, Université Laval, (1990).