Transactions on the Built Environment vol 71, © 2004 WIT Press, www.witpress.com, ISSN 1743-3509

Fluid-structure interaction using adaptive embedded unstructured grids

R. Löhner[1], J.D. Baum[2], E.L. Mestreau[2], C. Charman[3]
& D. Pelessone[4]
[1]*School of Computational Sciences and Informatics, George Mason University, USA.*
[2]*Applied Physics Operations, Science Applications International Corp., USA.*
[3]*General Atomics, USA.*
[4]*Engineering Software System Solutions, USA*

Abstract

Fluid-structure interaction cases with severe topological change due to fragmentation or rupture in the structure have prompted the development of flow solvers using a so-called embedded mesh approach. A simple embedded domain method for node-based unstructured grid solvers is presented. Edges crossing embedded surface faces are either removed or duplicated. Several techniques to improve the treatment of boundary points close to the immersed surfaces are explored. Adaptive mesh refinement based on proximity to the curvature or corners of the embedded CSD surfaces is used to enhance the accuracy of the solution. User-defined or automatic deactivation for the regions inside immersed solid bodies is employed to avoid unnecessary work. Several examples are included that show the viability of this approach for coupled fluid-structure problems.

Transactions on the Built Environment vol 71, © 2004 WIT Press, www.witpress.com, ISSN 1743-3509

1 Introduction

The numerical solution of Partial Differential Equations (PDEs) is usually accomplished by performing a spatial and temporal discretization with subsequent solution of a large algebraic system of equations [Löh01]. The spatial discretization is commonly performed via polyhedra, also called (finite) volumes or elements. The final assembly of these polyhedra yields the so-called mesh. The transition from an arbitrary surface description to a proper mesh still represents a difficult task. Considering the rapid advance of computer power, together with the perceived maturity of field solvers, an automatic transition from arbitrary surface description to mesh becomes mandatory.

Two types of grids are most commonly used: body-conforming and embedded. For body-conforming grids the external mesh faces match up with the surface (body surfaces, external surfaces, etc.) of the domain. This is not the case for the embedded approach (also known as ficticious domain, immersed boundary or Cartesian method), where the surface is placed inside a large mesh (typically a regular parallelepiped), with special treatment of the elements close to the surfaces.

Considering the general case of moving or deforming surfaces with topology change, both approaches have complementary strengths and weaknesses:

a) Body-Conforming Moving Meshes: the PDEs describing the flow need to be cast in an arbitrary Lagrangean-Eulerian (ALE) frame of reference, the mesh is moved in such a way as to minimize distortion, if required the topology is reconstructed, the mesh is regenerated and the solution reinterpolated. All of these steps have been optimized over the course of the last decade, and this approach has been used extensively [Bau96, Bau99, Sha00]. The body-conforming solution strategy exhibits the following shortcomings: the topology reconstruction can sometimes fail for singular surface points; there is no way to remove subgrid features from surfaces, leading to small elements due to geometry; reliable parallel performance beyond 16 processors has proven elusive for most general-purpose grid generators; the interpolation required between grids invariably leads to some loss of information; and there is an extra cost associated with the recalculation of geometry, wall-distances and mesh velocites as the mesh deforms.

b) Embedded Fixed Meshes: the mesh is not body-conforming and does not move. Hence, the PDEs describing the flow can be left in the simpler Eulerian frame of reference. At every timestep, the edges crossed by CSD faces are identified and proper boundary conditions are applied in their vicinity. While used extensively [Mel93, Kar95, Pem95, Lan97, Aft00, Pes02] this solution strategy also exhibits some shortcomings: the boundary, which has the most profound influence on the ensuing physics, is also the place where the worst elements are found; at the same time, near the boundary, the embedding boundary conditions need to be applied,

Transactions on the Built Environment vol 71, © 2004 WIT Press, www.witpress.com, ISSN 1743-3509

reducing the local order of approximation for the PDE; no stretched elements can be introduced to resolve boundary layers; adaptivity is essential for most cases; and there is an extra cost associated with the recalculation of geometry (when adapting) and the crossed edge information.

The development of the present embedded, adaptive fixed mesh capability was prompted by the inability of Computational Structural Dynamics (CSD) codes to ensure strict no-penetration during contact. Several blast-ship interaction simulations revealed that the amount of twisted metal was so considerable that any enforcement of strict no-penetration (required for consisted topology reconstruction) became impossible. Hence, at present the embedded approach represents the only viable solution for this class of fluid-structure interaction problems.

It may appear somewhat contradictory to use an unstructured (tetrahedral) solver in conjunction with surface embedding. Most of the work carried out to date was in conjunction with Cartesian solvers [Mel93, Kar95, Pem95, Lan97, Aft00], the argument being that flux evaluations could be optimized due to coordinate alignment. However, the achievable gains of such coordinate alignment may be limited due to the following mitigating factors:

a) For most of the high resolution schemes the cost of limiting and the approximate Riemann solver far outweigh the cost of the few scalar products required for arbitrary edge orientation;

b) The fact that any of these schemes (Cartesian, unstructured) requires mesh adaptation in order to be successful immediately implies the use of indirect addressing; given current trends in microchip design, indirect addressing, present in both types of solvers, may outweigh all other factors;

c) Three specialized (x,y,z) edge-loops versus one general edge-loop, and the associated data reorganization implies an increase in software maintenance costs.

For a tetrahedral based solver, surface embedding represents just another addition in a toolbox of mesh handling techniques (mesh movement, overlapping grids, remeshing, h-refinement, etc.).

In what follows, we denote by CSD faces the surface of the computational domain that is embedded. We implicitly assume that this information is given by a triangulation, which typically is obtained from a CAD package via STL files, remote sensing data or from a CSD code in coupled fluid- structure applications.

2 Treatment of embedded surfaces

Two basic approaches have been proposed to modify field solvers in order to accommodate embedded surfaces: force-based and kinematics-based. The first type applies an **equivalent balancing force** to the flowfield in order to achieve the kinematic boundary required at the embedded surface [Pin01, Pes02]. The second approach, followed here, is to apply **kinematic boundary conditions** at the nodes close to the embedded surface. Depending on the required order of accuracy and simplicity, a first or second-order (higher-order) scheme may be chosen to apply the kinematic boundary conditions. Figure 1 illustrates the basic difference between these approaches.

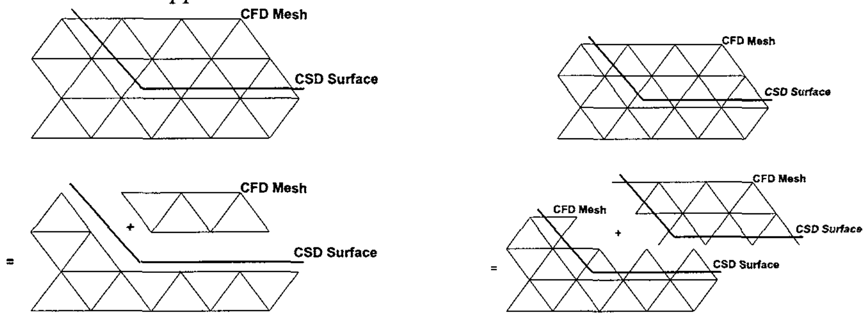


Figure 1a: 1st and 2nd Order Treatment of Embedded Surfaces

A first-order scheme can be achieved by:

- Eliminating the edges crossing the embedded surface;
- Forming boundary coefficients to achieve flux balance;
- Applying boundary conditions for the end-points of the crossed edges based on the normals of the embedded surface.

A second-order scheme can be achieved by:

- Duplicating the edges crossing the embedded surface;
- Duplicating the end-points of crossed edges;
- Applying boundary conditions for the end-points of the crossed edges based on the normals of the embedded surface.

3 Determination of crossed edges

Given the CSD triangulation and the CFD mesh, the first step is to find the CFD edges cut by CSD faces. This is performed by building first an octree of the CSD faces. Then, a (parallel) loop is performed over the edges. For each edge, the bounding box of the edge is built. From the octree, all the faces in the region of the bounding box are found. This is followed by an in-depth test to determine which faces cross the given edge. The crossing face closest to each of the edge end-nodes is stored. This allows to resolve cases of thin gaps or cusps. Once the faces crossing edges are found, the closest face to the end-points of crossed edges is also stored. This allows to apply boundary conditions for the points close to the embedded surface. For transient problems, the procedure described above can be improved considerably. The key assumption is that the CSD triangulation will not move over more than 1-2 elements during a timestep. If the topology of the CSD triangulation has not changed, the crossed-edge information from the previous timestep can be re-checked. The points of edges no longer crossed by a face crossing them in the previous timestep are marked, and the neighbouring edges are checked for crossing. If the topology of the CSD triangulation has changed, the crossed-edge information from the previous timestep is no longer valid. However, the points close to cut edges in the previous timestep can be used to mark 1-2 layers of edges. Only these edges are then re-checked for crossing.

4 Boundary conditions

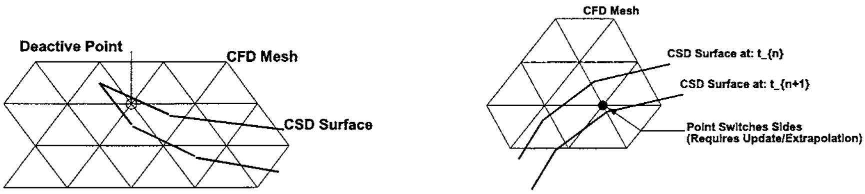
For the new boundary points belonging to cut edges the proper PDE boundary conditions are required. In the case of flow solvers, these are either an imposed velocity or an imposed normal velocity. For limiting and higher-order schemes, one may also have to impose boundary conditions on the gradients. The required surface normal and boundary velocity are obtained directly from the closest CSD face to each of the new boundary points. These low-order boundary conditions may be improved by extrapolating the velocity and normal pressure gradient from the surface with field information. A higher-order treatment of embedded surfaces may be achieved by using ghost points or mirrored points to compute the contribution of the crossed edges to the overall solution. This approach presents the advantage of not requiring the modification of the mass matrix as all edges (even the crossed ones) are taken into consideration. It also does not require an extensive modification of the various solvers. On the other hand, it requires more memory due to duplication of crossed edges and points, as well as (scalar) CPU time for renumbering/reordering arrays. Particularly for moving body problems, this may represent a considerable CPU burden.

5 Deactivation of interior regions

For highly distorted CSD surfaces, or for CSD surfaces with thin reentrant corners, all edges surrounding a given point may be crossed by CSD faces (see Figure 2). The best way to treat such points is to simply deactivate them [Löh01, Chapter 16]. This deactivation concept can be extended further in order to avoid unnecessary work for regions inside solid objects. Two approaches were pursued in this direction: seed points and automatic deactivation.

a) <u>Seed Points</u>: In this case, the user specifies a point inside an object. The closest CFD field point to this so-called seed point is then obtained. Starting from this point, additional points are added using an advancing front (nearest neighbour layer) algorithm, and flagged as inactive. The procedure stops once points that

are attached to crossed edges have been reached.
b) Automatic Deactivation: For complex geometries with moving surfaces, the manual specification of seed points becomes impractical. An automatic way of determining which regions correspond to the flowfield one is trying to compute and which regions correspond to solid objects immersed in it is then required. The algorithm employed starts from the edges crossed by embedded surfaces. For the end-points of these edges an in/outside determination is attempted. Once this in/outside determination has been done for the end-points of crossed edges, the remaining points are marked using an advancing front algorithm.



Figures 2,3: Deactive Point and Extrapolation of Solution

6 Extrapolation of the solution

For problems with moving boundaries, mesh points can switch from one side of a surface to another (see Figure 3). For these cases, the solution must be extrapolated from the proper state. The conditions that have to be met for extrapolation are as follows:

a) The edge was crossed at the previous timestep and is no longer crossed;
b) The edge has one field point (the point donating unknowns) and one boundary point (the point receiving unknowns); and
c) The CSD face associated with the boundary point is aligned with the edge.

7 Adaptive mesh refinement

Adaptive mesh refinement is very often used to reduce CPU and memory requirements without compromising the accuracy of the numerical solution. For transient problems with moving discontinuities, adaptive mesh refinement has been shown to be an essential ingredient of production codes [Bau99, Löh99a, Löh99b]. For embedded CSD triangulations, the mesh can be refined automatically close to the surfaces. This has been done in the present case by including three additional refinement indicators (on top of the usual ones based on the flow variables). The first one looks at the edges cut by CSD faces, and refines the mesh to a certain element size or refinement level. The second, more sophisticated indicator, looks

Transactions on the Built Environment vol 71, © 2004 WIT Press, www.witpress.com, ISSN 1743-3509

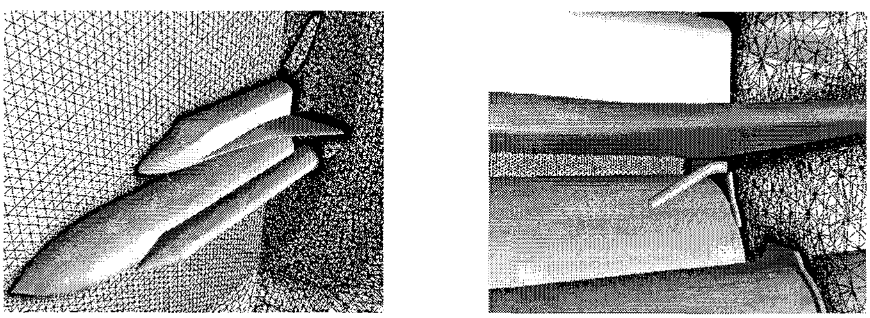
at the curvature of smooth regions of the surface, and refines the mesh only in regions where the element size is deemed insufficient. The third refinement indicator is especially taylored to ridges, edges or corners in the surface, and triggers a refinement in those regions.

8 Load/flux transfer

For fluid-structure interaction problems, the forces exerted by the fluid on the embedded surfaces need to be evaluated. This is done by computing first the stresses (pressure, shear stresses) in the fluid domain, and then interpolating this information to the embedded surface triangles. In principle, the integration of forces can be done with an arbitrary number of Gauss-points per embedded surface triangle. In practice, one Gauss-point is used most of the time.

9 Examples

Shuttle Ascend Configuration: The first example considered is the Space Shuttle Ascend configuration shown in Figure 4a. This is a flow-only case and considers the external flow at $Ma = 2$ and angle of attack $\alpha = 5^o$. The surface definition consisted of approximately 161 Ktria faces. The base CFD mesh had approximately 1.1 Mtet. For the geometry, a minimum of 3 levels of refinement were specified. Additionally, curvature-based refinement was allowed up to 5 levels. This yielded a mesh of approximately 16.9 Mtet. The grid obtained in this way, as well as the corresponding solution are shown in Figures 4b,c. Note that all geometrical details have been properly resolved. The mesh was subsequently refined based on density, up to approximately 28 Mtet. This physics-based mesh refinement is evident in Figures 4c-d.



Figures 4a,b: Shuttle: General View and Detail

Transactions on the Built Environment vol 71, © 2004 WIT Press, www.witpress.com, ISSN 1743-3509

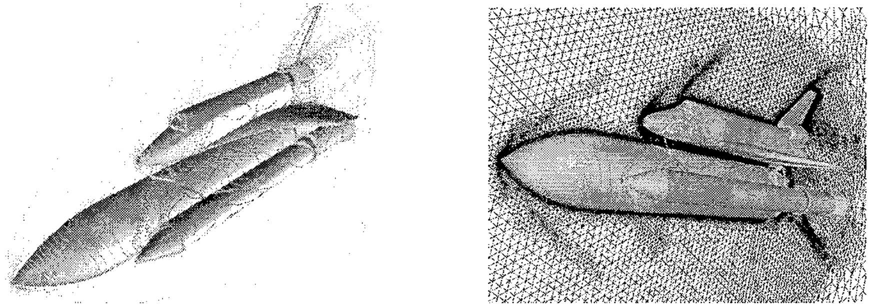


Figure 4c,d: Surface Pressure and Field Mach-Nr.; Surface Pressure and Mesh (Cut Plane)

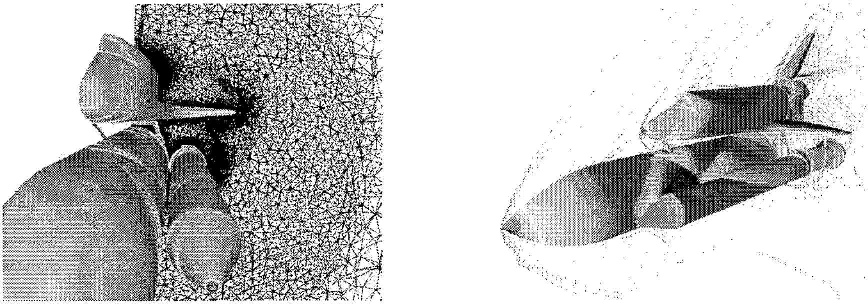
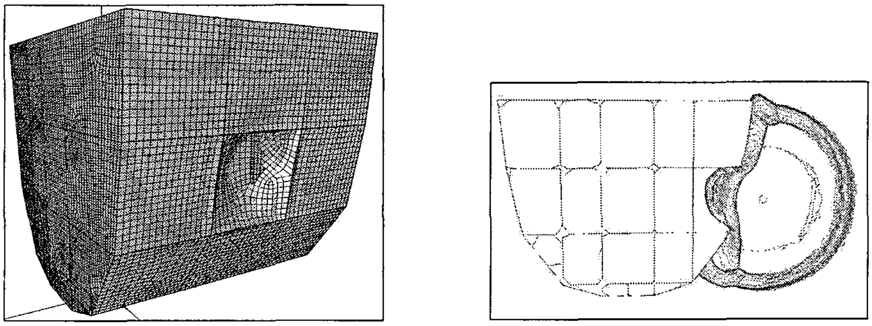


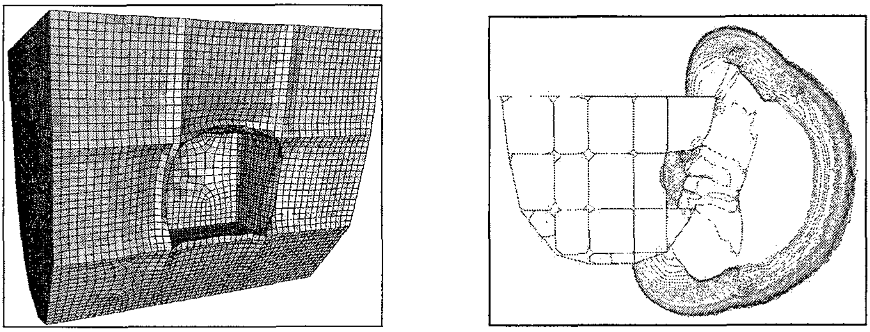
Figure 4e,f: Surface Pressure and Mesh (Cut Plane); Surface Pressure and Field Mach-Nr.

Blast Interaction With a Generic Ship Hull: Figure 5 shows the interaction of an explosion with a generic ship hull. For this fully coupled CFD/CSD run, the structure was modeled with quadrilateral shell elements, the fluid as a mixture of high explosive and air, and mesh embedding was employed. The structural elements were assumed to fail once the average strain in an element exceeded 60%. As the shell elements failed, the fluid domain underwent topological changes. Figures 5a-d show the structure as well as the pressure contours in a cut plane at two times during the run. The influence of bulkheads on surface velocity can clearly be discerned. Note also the failure of the structure, and the invasion of high pressure into the chamber. The distortion and inter-penetration of the structural elements is such that the traditional moving mesh approach (with topology reconstruction, remeshing, ALE formulation, remeshing, etc.) will invariably for this class of problems.

Transactions on the Built Environment vol 71, © 2004 WIT Press, www.witpress.com, ISSN 1743-3509



Figures 5a,b: Surface of Hull and Pressure in Cut Plane at 20msec



Figures 5c,d: Surface of Hull and Pressure in Cut Plane at 50msec

In fact, it was this particular type of application that led to the development of the present embedded CSD capability.

10 Conclusions and outlook

A CFD solver based on unstructured, body conforming grids has been extended to treat embedded or immersed CSD surfaces given by triangulations. Edges crossing embedded surface faces are either removed or duplicated. Adaptive mesh refinement based on proximity to, the curvature or corners/edges of the embedded CSD surfaces is used to enhance the accuracy of the solution. User-defined or automatic deactivation for the regions inside immersed solid bodies is employed to avoid unnecessary work. Several examples have been included that show the viability of this approach for coupled fluid-structure problems. Future work will center on improved boundary conditions, as well as faster crossing checks for transient problems.

Transactions on the Built Environment vol 71, © 2004 WIT Press, www.witpress.com, ISSN 1743-3509

Acknowledgements

This work was partially supported by DTRA and AFOSR. The technical monitors were Drs. Michael Giltrud, Young Sohn and Leonidas Sakell.

References

[Aft00] M.J. Aftosmis, M.J. Berger and G. Adomavicius - A Parallel Multilevel Method for Adaptively Refined Cartesian Grids with Embedded Boundaries; *AIAA*-00-0808 (2000).
[Bau96] J.D. Baum, H. Luo, R. Löhner, C. Yang, D. Pelessone and C. Charman - A Coupled Fluid/Structure Modeling of Shock Interaction with a Truck; *AIAA*-96-0795 (1996).
[Bau99] J.D. Baum, H. Luo, E. Mestreau, R. Löhner, D. Pelessone and C. Charman - A Coupled CFD/CSD Methodology for Modeling Weapon Detonation and Fragmentation; *AIAA*-99-0794 (1999).
[Kar95] S.L. Karman - SPLITFLOW: A 3-D Unstructured Cartesian/ Prismatic Grid CFD Code for Complex Geometries; *AIAA*-95-0343 (1995).
[Lan97] A.M. Landsberg and J.P. Boris - The Virtual Cell Embedding Method: A Simple Approach for Gridding Complex Geometries; *AIAA*-97-1982 (1997).
[Löh99a] R. Löhner, C. Yang, J.D. Baum, H. Luo, D. Pelessone and C. Charman - The Numerical Simulation of Strongly Unsteady Flows With Hundreds of Moving Bodies; *Int. J. Num. Meth. Fluids* 31, 113-120 (1999).
[Löh99b] R. Löhner, C. Yang, J. Cebral, J.D. Baum, H. Luo, E. Mestreau, D. Pelessone and C. Charman - Fluid-Structure Interaction Algorithms for Rupture and Topology Change; *Proc. 1999 JSME Computational Mechanics Division Meeting*, Matsuyama, Japan, November (1999).
[Löh01] R. Löhner - *Applied CFD Techniques*; J. Wiley & Sons (2001).
[Mel93] J.E. Melton, M.J. Berger and M.J. Aftosmis - 3-D Applications of a Cartesian Grid Euler Method; *AIAA*-93-0853-CP (1993).
[Pem95] R.B. Pember, J.B. Bell, P. Colella, W.Y. Crutchfield and M.L. Welcome - An Adaptive Cartesian Grid Method for Unsteady Compressible Flow in Irregular Regions; *J. Comp. Phys.* 120, 278 (1995).
[Pes02] C.S. Peskin - The Immersed Boundary Method; *Acta Numerica* 11, 479-517 (2002).
[Pin01] S. Del Pino and O. Pironneau - Fictitious Domain Methods and Freefem3d; *Proc. ECCOMAS CFD Conf.*, Swansea, Wales (2001).
[Sha00] D. Sharov, H. Luo, J.D. Baum and R. Löhner - Time-Accurate Implicit ALE Algorithm for Shared-Memory Parallel Computers; *First International Conference on Computational Fluid Dynamics*, Kyoto, Japan, July 10-14 (2000).