Available online at www.sciencedirect.com

ScienceDirect

Mathematics and Computers in Simulation 78 (2008) 605–617

MATHEMATICS AND COMPUTERS IN SIMULATION

www.elsevier.com/locate/matcom

A comprehensive generalized mesh system for CFD applications

Roy Koomullil[a,*], Bharat Soni[a], Rajkeshar Singh[b,1]

[a] *Department of Mechanical Engineering, University of Alabama at Birmingham (UAB), 1530 3rd Avenue South, Birmingham, AL 35294-4461, United States*

[b] *Department of Mechanical and Aerospace Engineering, University of Florida, Gainesville, FL 32611, United States*

Available online 10 April 2008

Abstract

The numerical approaches used for the solution of governing equations of fluid flow are dictated highly by the topology of the domain discretization. Two of the most commonly used discretization approaches are the structured and unstructured topologies. This paper describes the discretization of the domain using generalized elements with an arbitrary number of nodes to combine the advantages of both the structured and unstructured methodologies. Numerical algorithms for the solution of the governing equations for generalized mesh, an approach for handling mesh movement applicable to rotating machineries, and the application of this framework for overset meshes to handle moving body problems are discussed. A library-based approach has been adopted for the implementation of overset capability for the framework. The results from the application of this framework for various applications are presented.

Keywords: Generalized meshes; Rotating machineries; Overset meshes; Library approach

1. Introduction

A preprocessing step for the computational field simulation is the discretization of the domain of interest and is called mesh generation. The process of mesh generation can be broadly classified into two categories based on the topology of the elements that fill the domain. These two basic categories are known as structured and unstructured meshes. The different types of meshes have their advantages and disadvantages in terms of both solution accuracy and the complexity of the mesh generation process. A structured mesh is defined as a set of hexahedral elements with an implicit connectivity of the points in the mesh. The structured mesh generation for complex geometries is a time-consuming task due to the possible need of breaking the domain manually into several blocks [1] depending on the nature of the geometry. An unstructured mesh is defined as a set of elements, commonly tetrahedrons, with an explicitly defined connectivity. The unstructured mesh generation process involves two basic steps: point creation and definition of connectivity between these points [2,3]. Flexibility and automation make the unstructured mesh a favorable choice although solution accuracy may be relatively unfavorable compared to the structured mesh due to the presence of skewed elements in sensitive regions like boundary layers. In an attempt to combine the advantages of both structured and unstructured meshes, another approach in practice is hybrid mesh generation [4]. In a hybrid mesh, the viscous region is filled with prismatic or hexahedral cells while the rest of the domain is filled with tetrahedral

* Corresponding author. Tel.: +1 205 934 0832.
E-mail addresses: rkoomul@uab.edu (R. Koomullil), bsoni@uab.edu (B. Soni), singhr@ge.com (R. Singh).
1 Current address: GE Global Research Center, Niskayuna, NY 12309 United States.

0378-4754/$32.00 © 2008 IMACS. Published by Elsevier B.V. All rights reserved.
doi:10.1016/j.matcom.2008.04.005

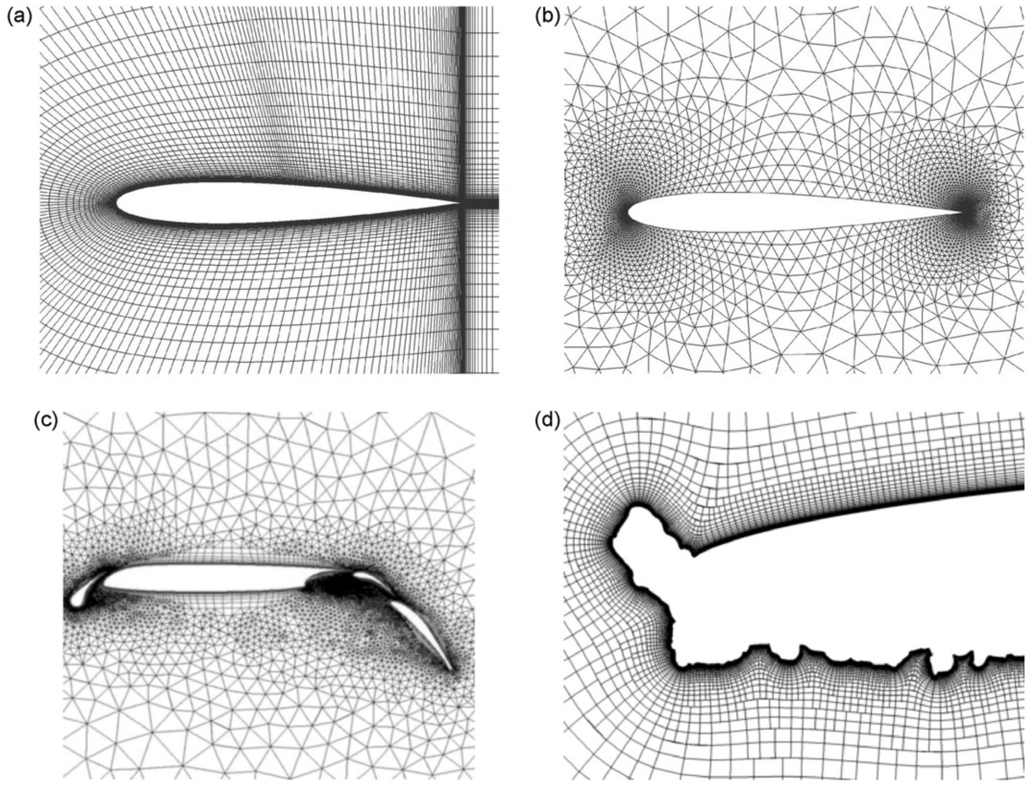

Fig. 1. Examples of mesh topologies: (a) structured mesh; (b) unstructured mesh; (c) hybrid mesh; (d) generalized mesh (*Courtesy*: Dr. David S. Thompson, Mississippi State University).

cells [5,6]. It has been observed that a hybrid mesh in viscous regions creates a lesser number of elements than a completely unstructured mesh with a similar resolution. The idea of using meshes of different element-topologies has been extended further to give rise to what is called a generalized mesh [7,8]. This type of mesh has no restrictions on the number of edges or faces on a cell, which makes it extremely flexible for topological adaptation. Examples of meshes of these various topologies are shown in Fig. 1.

The relative merits and demerits of these different mesh topologies are summarized in Table 1. As given in this table, the unstructured mesh has an advantage over the structured mesh in handling complex geometries, mesh adaptation using local refinement and de-refinements, moving mesh capability by locally repairing the bad quality elements, and load balancing using appropriate graph partitioning algorithms. In the case of a non-matched block-to-block boundary, interpolation issues have to be handled properly to satisfy the conservation principles. However, the structured mesh has a better accuracy for viscous calculations due to the fact that it can handle cells with very high aspect ratio cells in the boundary layer. The generalized mesh framework combines the advantages of both structured and unstructured

Table 1
Relative merits/demerits of different grid topologies

	Unstructured	Structured	Generalized
Geometric flexibility	+	−	+
Moving meshes	+	−	+
Mesh adaptation	+	−	+
Interpolation/conservation	+	−	+
Parallel computing	+	−	+
Viscous computation	−	+	+
Memory requirements	−	+	−
CPU requirements	−	+	−

topologies. The requirements of more memory and CPU time can be addressed by the tremendous growth in the development of faster and better computers. A comprehensive flow simulation system based on generalized mesh topology has been developed and is discussed in this paper. The numerical algorithms employed, the application of the system for rotating machineries, extension of the system for overset applications, and the results from the simulation of benchmark test cases are presented in the following sections.

2. Numerical approach

Due to the presence of the generalized meshes in the domain of interest, the finite volume scheme has been selected as the approach for the spatial discretization of the governing equations. The Navier–Stokes equations have been cast in the integral form and are taken as the governing equations for the fluid flow. For the present work we have used a cell-centered, finite volume scheme, in which cell-averaged flow variables are stored at the cell center. The inviscid numerical flux passing through the cell faces is calculated by Roe's approximate Riemann solver [9] as an exact solution for a linearized Riemann problem. For a first order accurate scheme, the variable vectors at the cell face are taken as the cell averaged values on either side of the face. Higher-order accuracy in the spatial discretization is obtained using a linear reconstruction of the primitive variables using the values of those in the neighboring cells. Using Taylor's series expansion, the flow variables are extrapolated to the cell-faces from either side of the face and those values are used to solve the Riemann problem. The gradient of the conserved variables at the cell center, appearing in the Taylor's series expansion, can be estimated using two different approaches. The first approach utilizes Green's theorem and the second approach uses the least-square principle. During the reconstruction process, local extrema may be created. This may produce spurious values near the regions where there is a sharp jump in the flow variables, as in the vicinity of contact discontinuities, expansion regions, etc. A limiter function has been applied to the Taylor's series expansion to preserve the monotonicity principle. The Barth and Jesperson limiter [10] and the limiter proposed by Venkatakrishnan [11] which is an extension of the van-Albada limiter, have been used for this purpose.

The temporal integration of the governing equations is achieved using an implicit scheme and a linearization procedure for the numerical flux crossing the cell face. The flux vector has to be linearized before the evaluation of the flux through the cell faces. The temporal terms in the governing equations are discretized using a second order accurate scheme. The numerical accuracy of the unsteady simulations has been improved using Newton iterations [12]. The matrix system resulting from the above equation is solved using the symmetric Gauss Seidel method. The geometric conservation law (GCL) [13] has been cast in the integral form and it is used to handle deforming meshes. In order to handle larger meshes, parallel computing is employed where the mesh is decomposed into multiple blocks using the public domain program METIS [14]. The graph of the mesh is utilized in the mesh decomposition process. The information across the mesh interfaces is passed between different blocks using a message-passing interface (MPI [15]).

The details of the above discussed numerical schemes and their validations are presented in Koomullil and Soni [8], Koomullil [16], and Koomullil et.al. [17]. However, for completeness a vortex shedding computation is presented for validation.

3. Application for rotating machinery

Research interests in the field of rotating machines are whetted by its variety of possible applications in aerospace, automotive, space, and marine industries. Due to the expensive nature of experimental investigation, the rapid development of computational power, and the relative economic viability of computational simulation, computational fluid dynamics (CFD) is emerging as a major tool for analysis and better understanding of the complex physics involved in the flow regime of such applications. Structured, unstructured, and hybrid meshes have been extensively employed in the CFD simulation of rotating machines [18–22]. A sliding mesh [23] approach is one of the common tools for controlling the meshes while simulating geometries in relative motion encountered in turbo-machinery applications. In such situations, due to the non-conformity of the faces of the mesh blocks, the solution is interpolated [24] to preserve the conservation laws. Another approach for handling the mesh in such simulations employs mesh deformations in a local region around the interface. In this method, the mesh is locally regenerated when the deformed mesh is no longer within the quality requirements. A similar deformation-based method called local grid distortion (LGD) has been used for structured meshes [18,25]. In the LGD method, when the deformation in the mesh is no longer acceptable, the mesh is suitably reconnected.

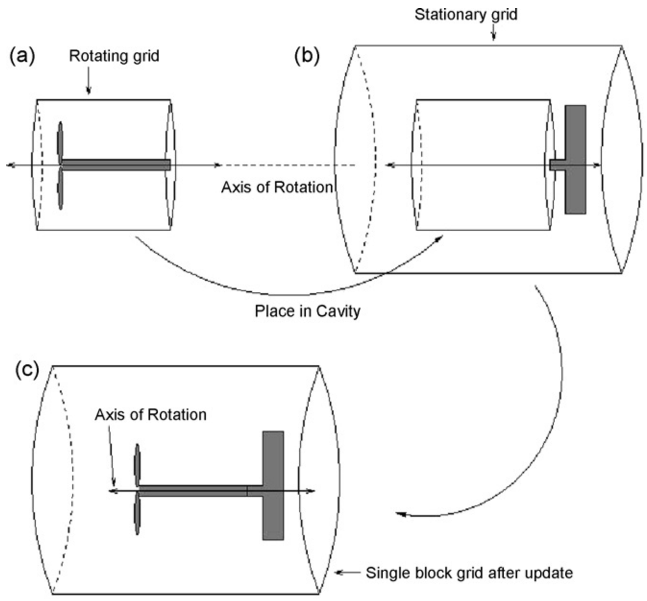

Fig. 2. A schematic diagram of the assembly of the stationary and the rotating grids.

In the current approach, the mesh for a single rotating component is divided into two domains [26]. The first domain is a cylindrical region about the rotating component and the other region is the stationary region that houses the rotating domain. As the rotating component moves, the mesh associated with it also moves rigidly with the moving body. The cell faces at the interface of one block are suitably broken based on the region of overlap with the cell faces at the interface of the other mesh block. Due to this process, generalized faces and cells appear at the interface. Here it may be worth highlighting that all the distortions in the mesh occur only at the interface of the mesh blocks and no new cells are created in the mesh during the rotation process. A schematic representation of these steps for a single rotating fan is shown in Fig. 2. The mesh shown in Fig. 2(a) represents a cylindrical mesh built around a fan while Fig. 2(b) shows the outer stationary mesh block which houses the rotating mesh block. The rotating mesh block, after rotation, is merged with the stationary mesh and the mesh data is updated to form a single block mesh as shown in Fig. 2(c). The resulting single block mesh may have generalized faces and cells at the interface. In the present approach, the shape of the rotating region of the mesh is not limited to a cylindrical shape, but can be any shape that is terminated by two planar surfaces perpendicular to the axis of rotation. Also, in the current approach the speed of rotation of the turbo-machines can be arbitrary. This capability can be utilized for the simulation of accelerating or decelerating rotors and propellers. A few examples of some possible configurations of the rotating mesh are shown in Fig. 3.

The steps involved in the CFD simulation of rotating machinery using the present methodology to handle the changes in the mesh due to rotation of one or more components are shown in Fig. 4, in which the rotating meshes 2 and 3 are optional. In this chart, the stationary mesh that houses all of the rotating blocks, and the other meshes represent the

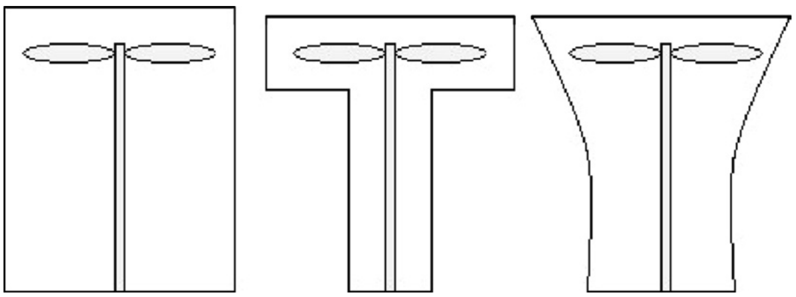

Fig. 3. Examples of a few acceptable configurations for rotating domains.

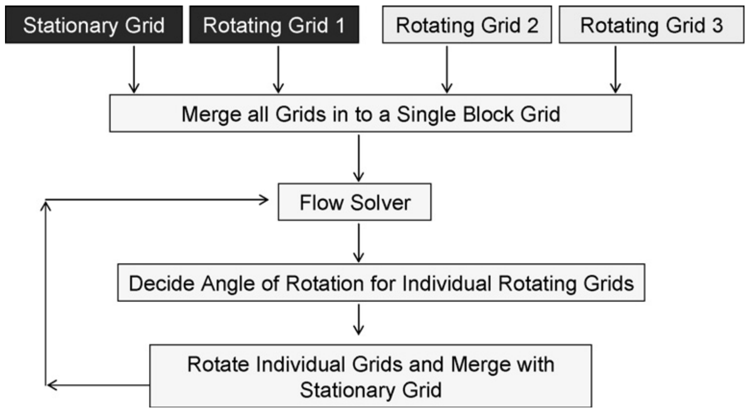
Stationary Grid
Rotating Grid 1
Rotating Grid 2
Rotating Grid 3
Merge all Grids in to a Single Block Grid
Flow Solver
Decide Angle of Rotation for Individual Rotating Grids
Rotate Individual Grids and Merge with Stationary Grid

Fig. 4. An outline of the steps followed for CFD simulation of a rotating machine using the present mesh handling technique.

meshes around the rotating components. As indicated in Fig. 4, all of the meshes are assembled into a single block mesh initially and passed to a solver that, in turn, specifies the angle of rotation for the rotating components. Following this, each mesh block built around the rotating components is rotated rigidly by the prescribed angle. The mesh near the interfaces of these mesh blocks is modified as described in the following sections. The resulting mesh is again combined into a single block mesh with generalized faces near the interfaces of the mesh blocks. Since the original mesh blocks are used to get the mesh after every prescribed rotation, the number of cells, faces and nodes in the mesh are restricted from potentially increasing continuously. The approach for the mesh rotation and the appropriate update of the data structure are described in the following sections.

As a mesh block is rotated by a specified angle, its interface surfaces slide over the corresponding interface surfaces of the stationary mesh block. The mesh on these interfaces is modified appropriately to obtain a valid single block mesh incorporating this rotation. When the surfaces of the rotating mesh slide over the surfaces of the stationary mesh, the overlapping faces of the corresponding two surfaces are split, causing the appearance of generalized elements on these interfaces. This concept is explained with a 2D example below and is extended to a 3D case in the subsequent sections.

Fig. 5 shows 2D unstructured mesh for a rotating component and a stationary mesh-domain that houses this rotating mesh. A schematic orientation of the rotating mesh inside the stationary mesh after the rotating mesh block is rotated by some specific angle is shown in Fig. 6. From this figure, it can be seen that some of the nodes of the rotating mesh have penetrated into the stationary mesh. A detailed view of the mesh line orientation at one of the nodes of the stationary mesh is depicted in Fig. 7(a). In this figure, the nodes $n1'$, $n2'$, $n7$ and $n8$ belong to the rotating mesh faces (*rf*1 and *rf*2) while the faces *sf*1, *sf*2, *sf*3 and *sf*4 belong to the stationary mesh. As shown in Fig. 7(a), the nodes $n1'$ and $n2'$ penetrate in to the stationary mesh due to partial overlapping of faces *rf*1 and *rf*2 with the stationary mesh faces. In

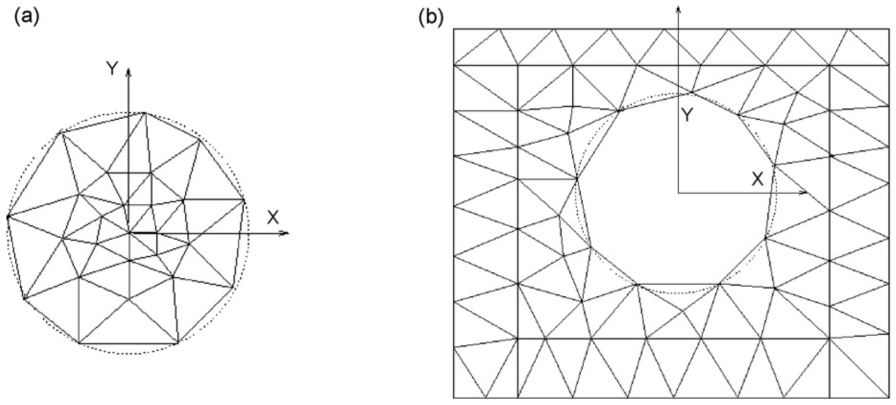

Fig. 5. Schematic mesh-handling strategy: (a) a representative mesh around a rotating object; (b) a stationary mesh to house the rotating mesh.

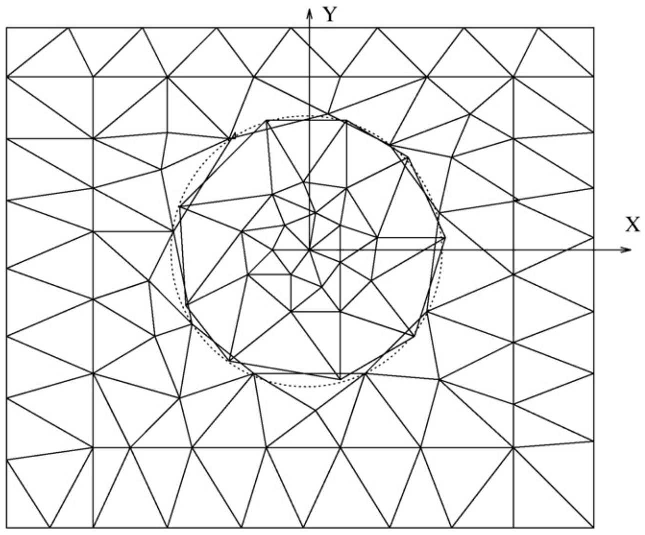

Fig. 6. Orientation of a rotating mesh inside the stationary mesh.

order to remove the overlapping of the rotating and stationary meshes, node $n1'$ is moved to the position of $n9$ shown in Fig. 7(b). The node $n9$ lie on the stationary edge formed by nodes $n1$ and $n6$ in such a way that node $n9$ makes the same angle with the x-coordinate direction in Fig. 6 as node $n1'$ did. Similarly, node $n2'$ is moved to the location of $n10$ in Fig. 7(b). Finally, the edge formed by the nodes $n9$ and $n10$ is broken to contain node $n6$ of the stationary mesh to get the final updated faces as shown in Fig. 7(b).

It can be seen in the above-described 2D mesh updating process that only the interface nodes of the rotating mesh are adjusted. The face areas of the 2D stationary mesh do not change, but their edges on the interface may be broken, thus causing an increase in the number of edges for the interface faces of the stationary mesh. As shown in Fig. 7(b), the stationary faces *sf*1 and *sf*4, after updating, have additional nodes $n9$ and $n10$, respectively, and all the changes in the face areas occur only in the rotating mesh-faces *rf*1 and *rf*2. In a 3D situation, similar to the 2D case, all the cell volume changes occur only in the interface cells of rotating meshes. The interface cells of the stationary mesh may have new faces created after being split by the interface faces of rotating meshes but they do not undergo any volume change. This choice, however, can be reversed without any complication such that all the volume changes occur only in stationary interface cells. However, for the present implementation, all the volume changes are taken only by rotating interface cells.

Similar to the above described 2D situation, in a 3D case when the rotating mesh interface surfaces slide over stationary mesh interface surfaces, the triangular interface faces are broken suitably and the interface cell-connectivity data is updated to incorporate newly created nodes and faces. Due to the discretized nature of the surfaces, the nodes of rotating meshes may penetrate into the stationary mesh, making the task of deciding the splitting of faces difficult. To

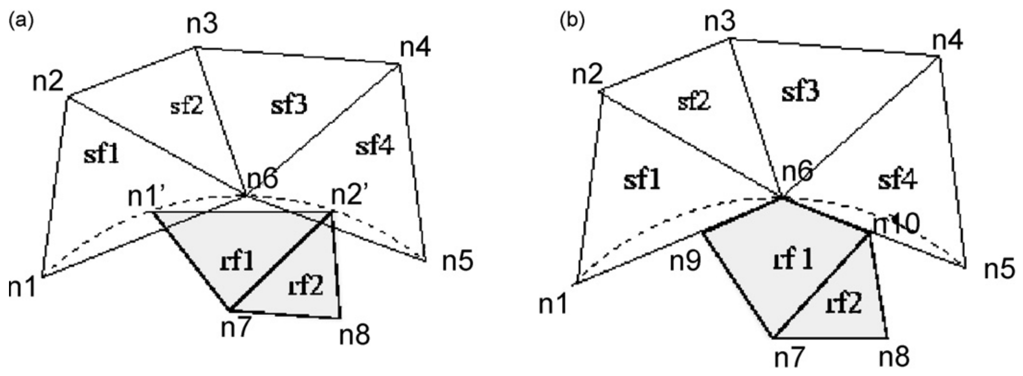

Fig. 7. Detailed view of the mesh near a node in the stationary mesh. (a) Mesh orientation before modification; (b) mesh orientation after modification.

avoid 3D intersection computations, all the rotating and stationary surfaces are mapped to a 2D space, and the strategy to split faces of rotating and stationary mesh in 3D is decided based on the overlap of faces of sliding surfaces in their 2D mapped planes. All the face-splitting operations are carried out in the 2D space and then the newly created faces and nodes are projected back to the 3D space.

4. Generalized overset framework

Computational simulation of multiple bodies in relative motion is a challenging problem and needs the capability to handle moving meshes in the domain. One of the approaches for simulation of this type of problem uses mesh deformation. In this approach as the different bodies move relative to each other, the mesh between them is deformed using different numerical strategies such as tension or torsion spring analogies. The drawback of this approach is that the mesh quality needs to be checked during each time iteration level. Also, the mesh needs to be regenerated either locally or globally when the quality of the mesh deteriorates. This results in the limitation of application of problems involving smaller movement of the bodies.

Another approach for this type of simulation is using overset meshes [27,28]. An overset composite mesh is formed from several overlapping meshes along with the domain connectivity information that specifies the points that are excluded from the computations (hole points), the fringe or receptor points where an interpolated value is required, and the donor locations that will provide the interpolated information. An overset solution requires the modification of the system of equations to change the fringe and hole locations from a field point to a specified boundary point. The solution must be interpolated from the donor meshes, transmitted to the receptor meshes, and applied as a specified boundary value. The bookkeeping of the details of the different meshes and the transfer of information across them is the most difficult task in the development of the overset framework. This makes implementation of the overset mesh capability into various flow solvers very time-consuming.

Structured overset meshes have proven capability in modeling bodies in relative motion. However, generation of structured Chimera meshes for complex geometries is fairly time-consuming. A recent development in the mesh generation for complex geometries is based on generalized mesh topology. This technology has been successfully applied for single mesh applications [4–8]. However, the generalized overset mesh framework has not been reported in the literature. The development and validation of overset mesh technology for generalized meshes are presented below.

In this framework, meshes are generated around individual components of a complex system and the hole cutting of overlapping mesh is achieved using Chimera ToolKit (CTK). The addition of overset mesh capability to an existing CFD solver requires two new software components developed by Dr. Ralph Noack of Army Research Laboratory, Aberdeen Proving Ground. The first one is the component within the flow solver to interpolate the solution from one mesh, to transfer the interpolated data to the appropriate processor, and to apply the data as a boundary condition at the appropriate locations. The second component is the identification of the overset domain connectivity, which includes (1) the specification of locations in the mesh system that are outside the domain of interest (hole points), (2) the appropriate inter mesh boundary points that require interpolated boundary data, and (3) the locations in overlapping meshes that will provide the interpolated data. The present overset capability uses the Donor interpolation Receptor Transaction library (DiRTLib) [29] to provide the overset capability within the flow solver and the SUGGAR [30,31] code for providing the domain connectivity for the overset composite mesh systems.

The Donor interpolation Receptor Transaction library (DiRTLib) [29] is a solver neutral library that simplifies the addition of an overset capability to a flow solver by encapsulating these required operations. DiRTLib makes calls to solver interface functions, which are written for the specific flow solver, to interface to the solver data. The two basic interface functions get data from solver memory for use in the donor interpolations and put data into solver memory for use at the fringe locations. Other DiRTLib function calls are used to fill a solver provided array with a value to indicate if the location is a hole or a fringe point. A few DiRTLib function calls are inserted in the proper location in the solver execution to perform the required interpolation, transmission and application of the data to the appropriate fringe/receptor point. Additional calls are available to identify individual points or cells that are moving and to obtain an appropriate transformation to position the mesh in the new location. The fact that the overset composite mesh consists of multiple overlapping meshes is completely hidden from the unstructured flow solver to minimize the changes required to the solver.

The Structured, Unstructured, and Generalized overset Mesh AssembleR (SUGGAR) code [30,31] is an overset mesh assembly program designed for moving body simulations that can create an overset composite mesh from structured, unstructured, and/or general polyhedral meshes for node and cell centered flow solvers. SUGGAR has several different approaches for determining the interpolation weights for cell-centered solvers. These approaches include a simple inverse distance weighting, a procedure based upon a least square fit of the cell centroids, and a procedure based upon a Laplacian. The weights for the inverse distance weighting procedure will be bounded between zero and one and hence will be monotone: the interpolated value will be between the minimum and maximum values of the donor member values. The other procedure can have weights that are greater than one and hence will not be monotone, which can lead to stability problems in the flow solver. To alleviate this problem, DiRTLib has an option to limit or to clip the interpolated value to be between the extrema of the donor member values.

5. Results and discussions

The numerical approach that has been discussed in the previous sections has been well validated with various benchmark cases and has been reported before [8,16,17,26]. For the completeness of this article a few of the benchmark results are presented below.

5.1. Validation of numerical schemes

For the simulation of laminar vortex shedding from a circular cylinder, a Mach number of 0.2 and the Reynolds numbers of 300, 400, and 500 have been considered as the freestream conditions. The mesh used for this simulation is a structured mesh with O-type topology, and it contains 160,000 cells and 246,003 nodes. For these simulations a non-dimensional time step of 0.02 (based on the freestream velocity) is used. The simulation has been run for 10,000 iteration and the last 4096 iterations has been used for the statistical analysis. The Strouhal numbers for Reynolds numbers 300, 400, and 500 from the simulations is compared with experimental data in Table 2 and are in good agreement with the benchmark data.

A typical power spectral density of drag history from these simulations is shown in Fig. 8. It can be seen from this figure that the power spectral density is the highest for a Strouhal number of approximately 0.19. Also, an instantaneous density distribution during the vortex shedding from the laminar cylinder for a Reynolds number of 400 is shown in Fig. 9.

The turbulent vortex shedding from the circular cylinder is carried out using Spalart–Allmaras, Spalart–Allmaras DES, and SST Hybrid turbulence models [32,33]. The freestream conditions for this case include Mach number 0.2, temperature 278 K, and Reynolds number 8.0E+06. The non-dimensional time based on the freestream speed of air is taken as 0.006, which is equivalent to 9.1E−05 s. The first point of the wall is taken as 1.0E−06, and the mesh consists of 239,469 nodes and 154,800 cells. The computed Strouhal numbers using different turbulence models are tabulated in Table 2 and are in good agreement with the experimental results, which varies between 0.306 and 0.308. A total of 15,000 time steps have been used for these simulations, and the last 4096 iterations were used for statistical analysis and the Strouhal number calculations. The summary of Strouhal number calculations for different turbulence models is given in Table 3.

The turbulent viscosity distributions in the wake region of the cylinder for different turbulence models are given in Fig. 10. It can be seen from Fig. 10(a) and (b) that the magnitude of the turbulent viscosity away from the cylinder is much smaller for SA DES model as compared with the original SA model. This is due to the fact that the destruction terms for DES model is a function of local meshes spacing rather than the normal distance to the wall. Also, a similar distribution of turbulent viscosity as in the case of S–A DES model can be seen for the SST hybrid model.

Table 2
Comparison of Strouhal numbers for laminar vortex shedding

Reynolds number	Simulation	Experimental Strouhal number
300	0.183	0.192–0.208
400	0.183	0.201–0.210
500	0.195	0.205–0.212

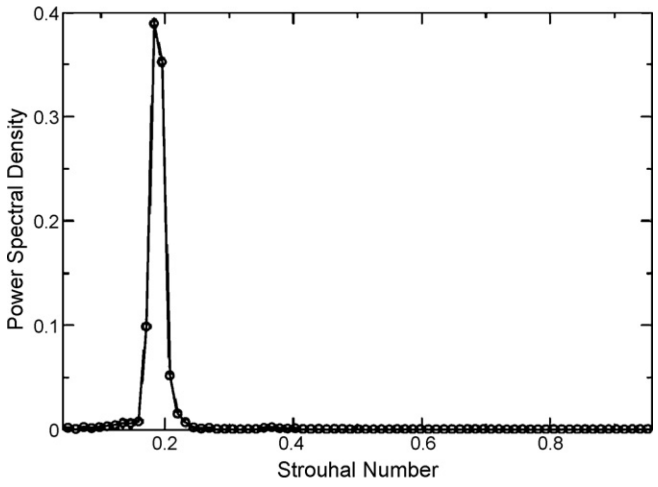

Fig. 8. Power spectral density distribution.

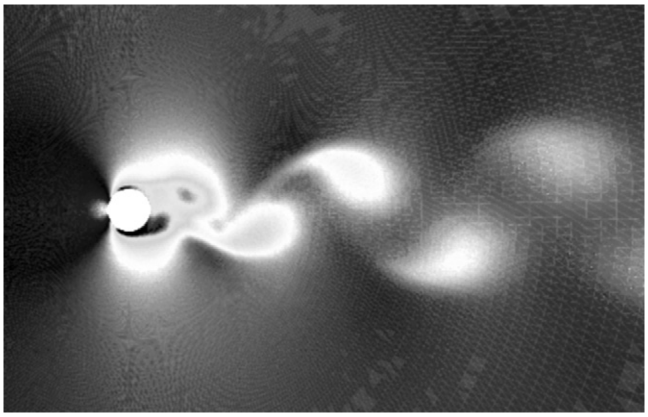

Fig. 9. Density distribution during vortex shedding from a laminar cylinder.

5.2. Generalized meshes for rotating machineries

The result from the application of the methodology that has been presented for rotating machineries is presented in this section. The algorithms for the mesh assembly process were tested for multiple rotating components. Two cylindrical domains housing an un-ducted SR7 fan in each (Fig. 11(a)) were rotated in opposite directions. At every rotation the sliding interfaces were merged to create a single mesh. The plot in Fig. 11(b) shows the variation in the total number of mesh nodes with rotation. Since the angular displacement from the original location (time = 0) is computed at every time step and the original meshes are appropriately rotated to create a single mesh, the total number of nodes in the mesh remains bounded in time.

The proof of principle study has been conducted for the coupling of the generalized mesh that has been generated using this approach and the flow solver using a shock-tube problem with pressure and density ratios of 50. A schematic

Table 3
Comparison of Strouhal numbers for turbulent case

Turbulence model	Strouhal number
SA	0.29
SA DES	0.2848
SST hybrid	0.3255

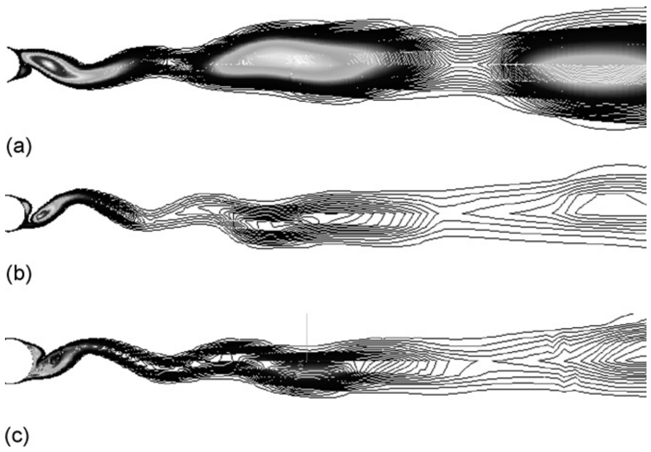

Fig. 10. Turbulent viscosity distribution for vortex shedding from a cylinder: (a) Spalart–Allmaras model; (b) Spalart–Allmaras DES model; (c) SST hybrid model.

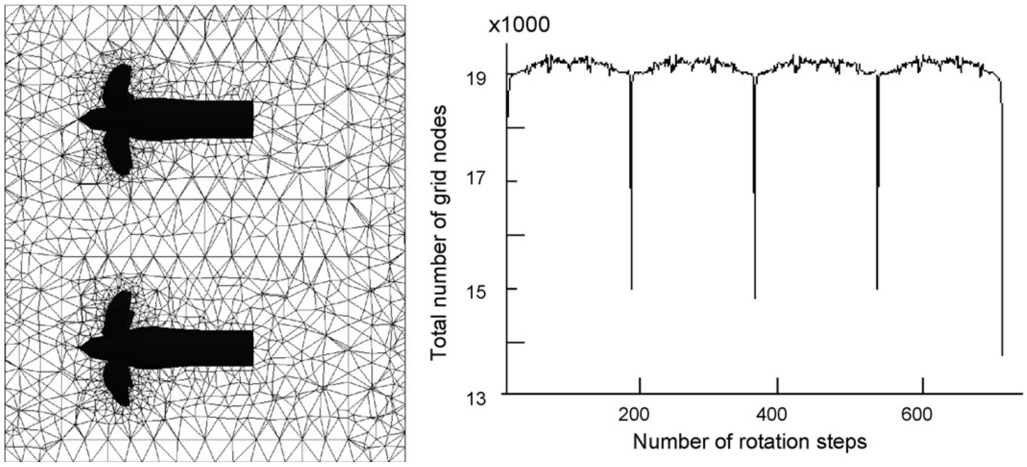

Fig. 11. Demonstration of controlled variation in number of nodes with current mesh-rotation strategy: (a) two un-ducted counter rotating SR7 propellers; (b) variation in number of total number of mesh nodes as the cylindrical mesh encasing the fans are with a speed of 0.5° per step.

diagram of a cylindrical shock tube, which is used for the study, is shown in Fig. 12. A cylindrical portion of the mesh, located in the middle of the shock tube, was rotated continuously at a speed of 2400 rpm to study the effects of the presence of a rotating component on the solution. A comparison of the computed pressure distribution without rotation, with rotation, and analytical results on a line touching the rotating cylindrical interface after 1.4196 s is shown in Fig. 13. The computed pressure distribution for the shock tube problem for stationary and rotating cylinders agree well with the analytical results as shown in Fig. 13. From the CPU time-requirement studies it has been noticed that the time taken for the mesh rotation and update is less than 5% of the simulation time.

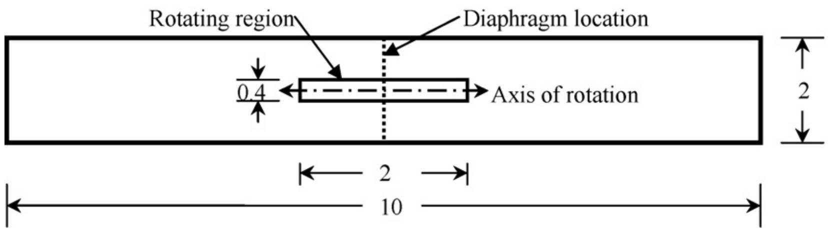

Fig. 12. Dimensions of the shock tube and the rotating cylindrical geometry.

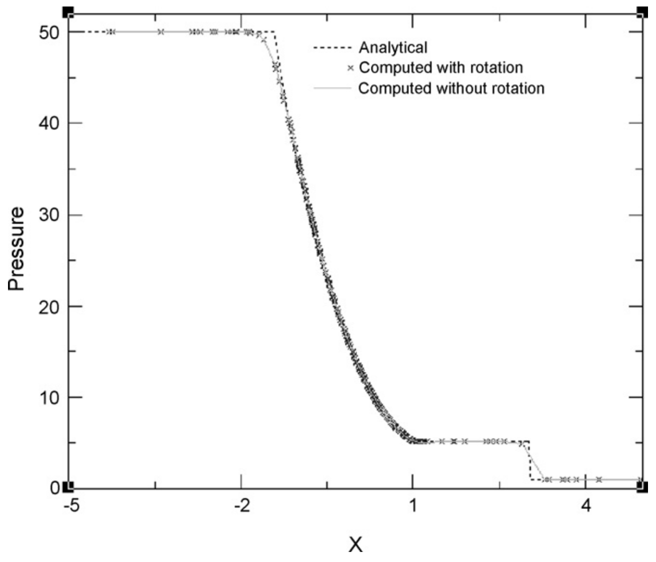

Fig. 13. Comparison of the analytical and computed pressure profile with rotating and stationary cylinders inside the shock tube.

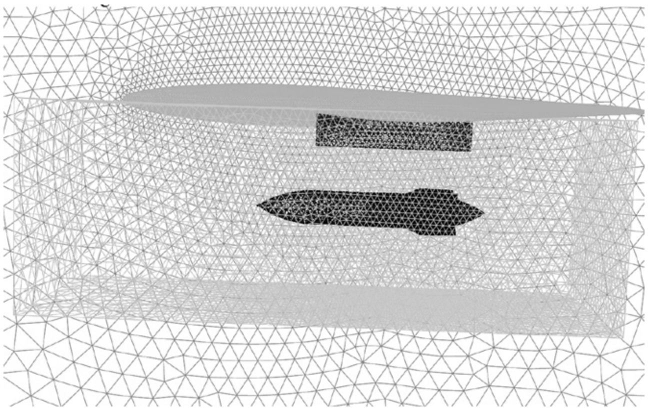

Fig. 14. Detailed view of the mesh for wing-store configuration.

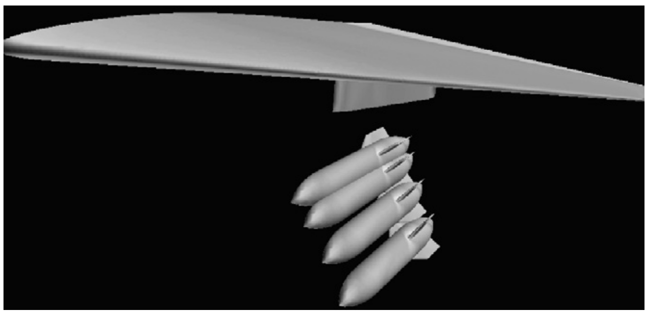

Fig. 15. Orientation of the store at different instance of time.

5.3. Simulation using overset mesh

The result from the simulation of a wing-pylon-store configuration using the generalized mesh framework is presented in this section. A detailed view of the mesh around the wing and the store that is used for the simulation is shown in Fig. 14. It can be seen from the figure that the mesh around the store cuts the wing geometry. For this simulation, the freestream Mach number is taken as 0.95 and the angle of attack as 0.0°. An unsteady simulation has been carried out using a prescribed path for the store during the separation and the results from the simulation is presented in Fig. 15. The location of the store at different time levels and the pressure distribution on the wing and the store are plotted in this figure.

The results presented in this section show the validation and capability of the generalized mesh-based CFD solver. The results from the laminar and turbulent vortex shedding from the cylinder validate the fundamental numerical schemes that are used in the framework, the results from the shock tube problem validate the concept of rotating regions inside a static mesh, and the store separation simulation demonstrate the capability of the framework in handling bodies in relative motion using overset meshes.

6. Conclusions

The numerical algorithms and applications of a generalized mesh-based simulation system have been developed and presented. This system combines the advantages of both the structured and unstructured mesh-based approaches. The applicability of this system for rotating machineries simulations and overset mesh applications has been discussed. The presented generalized mesh-based methodology for rotating machineries can be used for accelerating and decelerating objects without mesh regeneration and data interpolation. A library-based approach has been adopted for the development of the overset capability for the presented system. Results from various simulations have been presented for the validation of the generalized mesh-based simulation framework.

References

[1] J.F. Thompson, A general three-dimensional elliptic mesh generation system on a composite block structure, Comp. Methods Appl. Mech. Eng. 64 (1987) 377–411.
[2] N.P. Weatherill, A method for generating irregular computational meshes in multiply connected planar domains, Int. J. Numer. Methods Fluids 8 (1988) 181–197.
[3] D.L. Marcum, N.P. Weatherill, Unstructured mesh generation using iterative point insertions and local reconnection, AIAA J. 33 (1995) 1619–1625.
[4] J.A. Shaw, Hybrid meshes, in: J.F. Thompson, B.K. Soni, N.P. Weatherill (Eds.), Handbook of Mesh Generation, CRC Press, Boca Raton, FL, 1998.
[5] R. Noack, J. Steinbrenner, A three-dimensional hybrid mesh generation technique, AIAA-Paper, 95-1684CP, In 12th AIAA Computational Fluid Dynamics Conference, June 1995.
[6] Y. Kallinderis, Hybrid meshes and their applications, in: J.F. Thompson, B.K. Soni, N.P. Weatherill (Eds.), Handbook of Mesh Generation, CRC Press, Boca Raton, FL, 1998.
[7] D.S. Thompson, S. Chalasani, B.K. Soni, Generations of generalized meshes by extrusion from surface meshes of arbitrary topology, in: M. Cross, P.R. Eiseman, J. Hauser, B.K. Soni (Eds.), Proceedings of the 7th International Conference on Numerical Mesh Generation in Computational Field Simulations, September, 2000, pp. 1061–1070.
[8] R.P. Koomullil, B.K. Soni, Flow simulation using generalized static and dynamics meshes, AIAA J. 37 (12) (1999) 1551–1557.
[9] P.L. Roe, Approximate Riemann solvers, parameter vector, and difference schemes, J. Comput. Phys. 43 (1981) 357–372.
[10] T.J. Barth, D.C. Jespersen, The design and application of upwind schemes on unstructured meshes, AIAA-Paper 89-0366, January 1989.
[11] V. Venkatakrishnan, On the accuracy of limiters and convergence to steady state solutions, AIAA-Paper 93-0880, January 1993.
[12] D.L. Whitfield, L. Taylor, Discretized Newton-relaxation solution of high resolution flux-difference split schemes, AIAA-Paper 91-1539, June 1991.
[13] P.D. Thomas, C.K. Lombard, Geometric conservation law and its application to flow computations on moving meshes, AIAA J. 17 (10) (1978) 1030–1037.
[14] G. Karypis, V. Kumar, METIS: Unstructured Graph Partitioning and Sparse Matrix Reordering System, Version 2.0, Department of Computer Science, University of Minnesota, 1995.
[15] W. Gropp, E. Lusk, A. Skjellum, Using MPI, portable parallel programming with the message-passing interface, Sci. Eng. Commun. Ser. (1994).
[16] R.P. Koomullil, Flow simulation system for generalized static and dynamics meshes, Ph.D. Dissertation, Mississippi State University, May 1997.
[17] R.P. Koomullil, D.S. Thompson, B.K. Soni, Iced airfoil simulation using generalized meshes, J. Appl. Numer. Math. 46 (3–4) (2003) 319–330.
[18] J.M. Janus, Advanced 3D CFD algorithm for turbomachinary, Ph.D. Dissertation, Mississippi State University, May 1989.
[19] S.S. Neerarambam, A parallel turbomachinary flow solver and application to distortion-induced compressor rotor stall, Ph.D. Dissertation, Mississippi State University, May 1998.
[20] A. Khawaja, Y. Kallinderis, Hybrid mesh generation for turbomachinery and aerospace applications, Int. J. Numer. Methods Eng. 49 (2000) 145–166.

[21] D.R. Lindquist, M.B. Giles, Generation and use of unstructured meshes for turbomachinery, in: Proceedings of the Computational Fluid Dynamics Symposium on Aeropropulsion NASA CP-10045, 1990.

[22] U. Ghia, K.N. Ghia, R. Ramamurti, Hybrid C–H meshes for turbomachinery cascades, in: K.N. Ghia, U. Ghia (Eds.), Advances in Mesh Generation, ASME Publication, New York, 1983, pp. 143–150.

[23] Z. Jaworski, M.L. Wyszynski, R.S. Badham, K.N. Dyster, I.P.T. Moore, A.W. Nienow, N.G. Ozcan-Taskin, J. McKemmie, Sliding mesh CFD flow simulation for stirred tanks, in: Proceedings of the Mixing XV, North American Mixing Conference, Banff, Canada, June, 1995.

[24] M. Rai, A relaxation approach to patched-mesh calculations with the Euler equations, J. Comput. Phys. 66 (1986) 99–131.

[25] A.R. Ghosh, Solutions of three-dimensional thin layer Navier–Stokes equations in rotating coordinates for flow through turbomachinary, M.S. Thesis Report, Mississippi State University, December 1996.

[26] R.K. Singh, R.P. Koomullil, B.K. Soni, Application of Generalized Meshes to CFD Simulation of Rotating machineries, AIAA-Paper 2003-3820. Available on CD ROM, Presented at the 16th AIAA Computational Fluid Dynamics Conference, Orlando, FL, June 23–26, 2003.

[27] W.M. Chan, R.J. Gomez III, S.E. Rogers, P.G. Buning, Best practices in overset mesh generation, AIAA-Paper 2002-3191, 32nd AIAA Fluid Dynamics Conference, 24–26 June, 2002, St. Louis, Missouri.

[28] P. Fasta, M.J. Shelley, A moving overset mesh method for interface dynamics applied to non-Newtonian Hele–Shaw flow, J.Comput. Phys. 195 (2004) 117–142.

[29] R.W. Noack, DiRTlib: a library to add overset capability to flow solvers, Presented at 6th Symposium on Overset Composite Mesh and Solution Technology, Fort Walton Beach, FL, October 8–12, 2002.

[30] R.W. Noack, SUGGAR: a general capability for moving body overset CFD, in preparation.

[31] R.W. Noack, T. Kadanthot, An Octree based overset mesh hole cutting method, in: Proceedings of the 8th International Conference On Numerical Mesh Generation in Computational Field Simulations, Honolulu, HI, 2002, pp. 783–792.

[32] R. Nichols, C. Nelson, Applications of hybrid RANS/LES turbulence models, AIAA-Paper 2003-0083, January 2003.

[33] C. Nelson, R. Nichols, Evaluation of hybrid RANS/LES turbulence models using an LES code, AIAA-Paper 03-3552, June 2003.