Unstructured Mesh Motion Using Sliding Cells and Mapping Domains

Sina Arabi, Ricardo Camarero and François Guibault

Département de génie mécanique, École Polytechnique de Montréal, Campus de l’Université de Montréal, 2500 chemin de Polytechnique, Montréal (Québec) Canada H3T 1J4

Email: sina.arabi@polymtl.ca

ABSTRACT
This paper addresses the problem of generating unstructured meshes with fixed connectivity for large rigid body motion. The proposed approach consists in generating a mesh in computational space for a generic configuration of the moving body. The management of body and mesh motion is then carried out in computational space using a sliding mesh paradigm. Afterwards, the mesh in physical space is obtained through the Winslow equation to map the computational mesh to the physical space.

Two new discretization techniques are implemented, validated and compared for performing the Winslow operator on unstructured grids. The first approach used a 9-point Cartesian stencil inside each patch of the computational mesh and discretizes the mapping operators on that using conventional finite difference schemes. The second approach used finite volume discretization technique by linearizing the system of mapping equations. This methodology is applied to complex geometric configurations representative of engineering applications.

1 INTRODUCTION
The numerical modeling of unsteady problems with large amplitude or relative motion of bodies, requires considerable care in the formulation of the domain discretisation specifically in devising schemes for the movement of the grids. These types of problems have led to development of algorithms ([1]) for simulating fluid physics, where dynamic grid generation for both viscous and inviscid regions play a significant role. Despite the considerable efforts addressed at this problem, efficiency and robustness remain critical issues. State of the art mesh motion methodologies applicable to unsteady problems can be divided into two major categories, as illustrated in Fig. 1, namely, one group is based on changing topology whereas the other maintains the mesh topology.

![Figure 1: Classification of Different moving mesh Methodologies](image)

The early attempts in dynamic grid generation are essentially remeshing techniques, where the entire grid is regenerated based on the new position of the boundaries as presented in [2, 3] and [4]. This approach can produce meshes of very high quality if the defined size function is well behaved, for example by equidistributing the error across the domain ([5]). However, it is very expensive computationally, as in addition to the actual remeshing, the technique requires the interpolation of the solution at each time step. A major improvement of the efficiency of this approach is to apply remeshing locally as determined by a mesh quality indicator, [6]. Based on this indicator, elements are removed, resulting in one or more voids in the mesh which are then remeshed according to the required distribution of mesh parameters provided by the error indicating process and merged into the global
mesh. Although applied locally, compared to the complete remeshing, this method remains computationally expensive for transient problems. Another mesh adaptation method, called morphing, applies local edge collapse operations for mesh coarsening, and incremental point insertion algorithms for mesh refinement, [7, 8]. Its robustness depends heavily on maintaining mesh quality during each adaptation cycle.

Mesh motion algorithms by fixed topology have been presented in the literature, with various approaches according to the amplitude of the body motion. A widely used method, considers the mesh as a network of springs and solves the static equilibrium equations for this network to determine the new location of the grid points. However, the major disadvantage of this approach is that the grid smoothness and regularity are lost when the grid is subjected to large motion.

Another promising method is based on the Radial Basis Functions interpolation presented in [9] which can be applied to mesh motion while preserving the cells’ connectivity. This is an interpolation technique where the displacements of boundary nodes are propagated onto the interior nodes. This has been applied successfully for large relative motions in engineering applications but since the cells remain attached to the boundary, it is not suitable for periodic rotary objects like modelling the mixer blades or propellers. PDE operators, such as Laplace equation with various diffusivity coefficients and biharmonic equation, have been used as a mechanism to generate and to smooth meshes. However, they still have limitations for the large linear and rotary motions in unsteady flow problems. Another attempt to solve large mesh deformation has been proposed by [10] using linear-elastic smoothing. One advantage of this approach is that it uses a variable elastic stiffness, inversely proportional to the cell volume, in order to preserve the mesh quality in viscous layers. In [11], an optimization procedure based on the adjoint method for linear elasticity mesh deformation technique is presented. While very robust for several engineering applications, this method has the same limitations as the Laplace equation and gives invalid cells for large motions, specially around high curvature regions or sharp corner points of boundaries.

Another technique is overset grids ([14]) that simplify the mesh management by superposition of the static and moving parts of the grids at the expense of the cost of interpolation.

2 Mesh Movement

The methods described so far, all share one major characteristic, which is that the mesh motion is carried out in physical space. Generally, each method satisfies a particular set of requirements at the expense of their important capabilities. For example, large motions can be handled by local remeshing techniques but present difficulty with accuracy and complexity. Similarly, while the spring analogy is very efficient and easy to apply, it fails for large motions. Furthermore, despite their significant benefits, the Radial Basis Functions interpolation, linear elasticity techniques and biharmonic equations are still not capable of handling large linear or periodic rotary motions encountered in the engineering applications. This is due to the constraint of the attachment of cells to the boundaries and the difficulties of maintaining mesh topology as the motion evolves.

A new approach is presented in this paper, based on a flow analogy, where the grid cells in computational space are advected past moving boundaries by a fictitious potential-like fluid flow. Then, this is followed by mapping the computational mesh to physical space. In this approach, the cells are allowed to slide on the boundaries in order to release the constraints present in the conventional mesh deformation techniques. In addition, the cells connectivities will be maintained except at specific points on the boundary. The main advantage of this approach is to preserve the mesh connectivity as time evolves, making it well suited for Arbitrary Lagrangian Eulerian (ALE) flow solvers. This approach is shown with a bold line in the general frame work for unstructured mesh motion technics, presented in Fig. 1.

The proposed procedure for this method consists in three major steps. First, an unstructured mesh is generated in computational space around a generic body. Then, the generic boundary is made to slide through the cells according to the defined trajectory for the physical boundary, and finally, the mesh is mapped on the physical domain. The mapping to an arbitrary domain in the physical space is carried out using a procedure detailed in the following sections, where the shape is imposed by the body coordinates through boundary conditions of the Winslow equations. Fig. 2 shows the independent and dependent variables in computational $C$ and physical $\Omega$ domains, respectively, which appear in the governing equations.

3 Mapping Domains

In the present work, the Winslow operator, Eqns. 1, are solved on a triangulated computational space to transform the mesh from the computational to the physical domain as motion evolves.
This system of equations are quasi-linear, coupled and elliptic in type.

\[
g_{11}x_{\xi\xi} - 2g_{12}x_{\xi\eta} + g_{22}x_{\eta\eta} = 0 \\
g_{11}y_{\xi\xi} - 2g_{12}y_{\xi\eta} + g_{22}y_{\eta\eta} = 0
\]  

where \( g_{11} = x_{\eta}^2 + y_{\eta}^2 \), \( g_{12} = x_{\eta}x_{\xi} + y_{\eta}y_{\xi} \), \( g_{22} = x_{\xi}^2 + y_{\xi}^2 \)

The most straightforward method to solve the system of Eqns. 1 is using a finite difference scheme on a structured mesh. However, little attention has been addressed to extend the method to unstructured meshes. The reason is that in contrast to Laplace and Poisson equations, the Winslow operator is in non-conservative form, and therefore, the conventional discretization schemes cannot be applied to unstructured meshes. In [12], a method based on Taylor series expansion to solve this operator on equilateral triangles where all the angles are equal to \( \pi/3 \) has been proposed. Another method is presented in [13] based on generating a virtual control volume in physical domain locally around each node in \( \Omega \) as a local computational space with the same number of neighboring nodes in physical space. Two new methods based on finite difference and finite volume schemes are presented to solve Eqns. 1 on unstructured meshes.

4 Numerical discretization

4.1 Finite difference scheme

In this work, a novel finite difference method on unstructured meshes is introduced, which is easier to apply compared to the previous works and closer to the classical schemes used on structured grids. This algorithm is based on a 9-point Cartesian stencil formed inside each patch around a node \((\xi_i, \eta_i)\) in computational space, as shown in Fig. 3. The values at the stencil points can be obtained by interpolation or a least-squares reconstruction method. From these, the first and second order derivatives are approximated to second order accuracy on the Cartesian stencil with an equal spacing \( \Delta \xi_i = \Delta \eta_i \). In the present study, the values of the dependent variables on the stencil nodes are linearly interpolated from the nodal values of the elements surrounding the node, and \( \Delta \xi_i, \Delta \eta_i \) are chosen as a fixed fraction of the length of the shortest edge connected to node \((\xi_i, \eta_i)\). The discrete operators are solved using an SOR-type iterative procedure. Since, almost all cells generated in the computational space satisfy the Delaunay criteria, the polygons are very close to the ideal regular polygon presented in [12]. This property insures that all the information from the polygon surrounding the node contributes to the solution process. In this method, if a triangle does not contribute directly to the construction of the stencil’s model, it is estimated that this would, heuristically, be smoothed out by the neighboring stencils.

4.2 Finite volume scheme

As mentioned in the previous section, the Winslow operator is in nonconservative form because the three coefficients, \( g_{11}, g_{12} \) and \( g_{22} \) are functions of the dependent variables in the computational space. Using a linearization procedure ([13]), Eqns. 1 can be integrated over a control volume defined around each point of the mesh in computational space. The integration path for the application of Green’s theorem is formed by joining the centroid of each triangular element to the midpoints of its sides, as shown by the dashed lines in Fig. 4. The hashed region in this figure, indicates a control volume with a centroid node which is the storage location of all dependent variables.

This results in the integral form of the linearized
Figure 4: Computational mesh and a control volume

Eqns. 1

\[ g_{11} \int \int x_{\xi\xi} d\Gamma - 2g_{12} \int \int x_{\xi\eta} d\Gamma + g_{22} \int \int x_{\eta\eta} d\Gamma = 0 \]  

(2)

Applying the divergence theorem to the second order derivative terms, for example for the first one, gives

\[ \int \int x_{\xi\xi} d\Gamma = \int \int \nabla \cdot F d\Gamma \]  

(3)

where the components of function \( F \) is \( F = (x_{\xi}, 0) \).

A similar procedure is applied to the \( x_{\eta\eta} \) term. One critical step in the procedure is the calculation of the cross-derivatives which requires a special treatment. In [13], the authors proposed the use of augmented cells around the control volumes. In the present work, it is proposed to solve all the operators in computational domain and without considering the virtual control volumes described in [13]. This method uses actual control volume in computational space leading to a simpler arithmetic procedure. In addition, the cross-derivatives are found using the same control volume and without considering auxiliary cells. Applying this procedure to the term \( x_{\xi\eta} \)

\[ \int \int x_{\xi\eta} d\Gamma = \int \int \nabla \cdot Q d\Gamma \]  

(4)

In this work, the same control volume shown in Fig. 4 is used to compute the cross-derivative terms and by integrating the terms in both \( \xi \) and \( \eta \) directions, the value of fluxes in these two directions will be obtained. The average value of these two fluxes gives the net flux of that cross-derivative term,

\[ Q = \frac{1}{2} (Q_1 + Q_2) \]  

(5)

where \( Q_1 = (0, x_\eta) \) and \( Q_2 = (x_\xi, 0) \).

Integrating over the control volume and applying the divergence theorem for each dependent variable, for example \( \xi \), gives

\[ \int \int \nabla \cdot F d\Gamma = \oint F \cdot n dS \]  

(6)

The term under the RHS integral represents the net flux that passes through the surface of the volume and, for the Winslow operator, can be evaluated as

\[ g_{11} \oint x_{\xi} n_\xi dS - 2g_{12} \frac{1}{2} \left( \oint x_{\eta} n_\eta dS + \oint x_{\xi} n_\eta dS \right) + g_{22} \oint x_{\eta} n_\eta dS = 0 \]  

(7)

It has been our specific experience that, taking only one component of the cross-derivative term after applying the Green’s theorem, wrongly deforms the final mesh.

5 Sliding Cells

The unstructured mesh management and mesh sliding technique require a computational mesh with a set of generic boundary definitions that match the element topology of the physical mesh. These generic boundaries for the translation and rotational motions are shown in Fig. 5.

Figure 5: Mapping of the generic configurations for translation and rotational motions.

To manage the mesh for translation or rotational motions, a computational mesh is created such that the boundary points lie on a slit of zero thickness or a circle, respectively. These boundary points are displaced or relocated at each step in the computational domain so that they conform to the correct geometry in physical space by applying mapping operators. Figure 6 shows the link between boundary nodes (circles) and mesh nodes of the body (slit) (squares). The boundary conditions applied to the mesh nodes are the co-
ordinates of the corresponding body nodes. In Fig. 7, the moving body is shown a set of circles: single circles at leading and trailing edges and pairs of circles in between. At a given step, the motion consists of changing the topological connection of the mesh nodes (squares) to the body nodes (circles) as the body moves through the mesh. Translation motion, the mesh nodes in computational domain remain fixed and the boundary nodes split the cells that lie on the defined trajectory. Rotational motion is obtained by rotating about the circle’s center which differs from the translation motion. This rotary motion is realized by the generic configuration of a circle rotating inside a mesh in computational space, \((\xi, \eta)\), as shown in Fig. 8. Then, the mapping operator is solved at each time step, but with different values of the boundary conditions.

![Figure 6: Arrangement of the nodes on the slit](image)

**Figure 6: Arrangement of the nodes on the slit**

![Figure 7: Sliding mesh nodes on the slit](image)

**Figure 7: Sliding mesh nodes on the slit**

### 6 RESULTS

Figures 9 and 10 illustrate the application of the translation motion at three time steps. Tracking specific nodes, in both computational and physical spaces (for example nodes 456 and 1366), clearly shows the mesh deformation resulting from the motion of the cylinder. It is necessary to mention that all the dimensions and lengths, except moving boundaries, in both computational and physical domains are equal. This is not a requirement, but is used here to simplify the procedure. The second example is the rotation of an airfoil shown in Fig. 11. The results obtained by the Winslow’s operator are shown at three angular positions. The mesh motion can be understood by following the path of node 1 which represents a boundary node on the airfoil and cell number 530 which slides on the boundary. The mapping operator for this application is discretized using the finite difference scheme explained in the previous section.

![Figure 8: Rotating physical boundary and sliding computational boundary over the mesh points](image)

**Figure 8: Rotating physical boundary and sliding computational boundary over the mesh points**

![Figure 9: Computational mesh at first, intermediate and final steps of sliding a slit in a cavity](image)

**Figure 9: Computational mesh at first, intermediate and final steps of sliding a slit in a cavity**
Figure 10: Corresponding physical mesh at first, intermediate and final steps of sliding a circle in a cavity

Figure 11: Rotating and sliding the cells around an airfoil in computational and physical domains in three different positions

The final example is the rotation of two four-petal configurations rotating in opposite directions as shown schematically in Fig. 12. Following the proposed procedure, the moving boundaries are two circles in computational space, mapped to two four-petal roses in physical space. For this case, the circles are discretized with the same number of nodes.

Figure 12: Boundaries in computational and physical spaces

The objective for this case is to illustrate relative motion with a wide range of amplitudes for the proximity of boundaries, while maintaining a fixed connectivity. This complex motion is shown in Figs. 13-15 for three different steps obtained using the Winslow equations solved by the finite volume discretization scheme. As is shown in these figures, the cells in the computational domain at different time steps are fixed except for the pointers on the boundaries. For example, tracking boundary nodes 1 and 240 in computational space at different time steps shows how these two circles slide inside the computational mesh. In addition, cell number 1, as a boundary cell number in the physical space, reveals how the boundary cells slide over the surface.

7 Conclusion

In this paper, a novel method for large rigid body motions based on the sliding cells and mapping domains was implemented and applied to generic type of motions. This method was found to overcome the difficulties encountered in traditional mesh motion techniques such as using Laplace, Poisson or linear elasticity equations, biharmonic method or RBF where the boundary cells remain attached to the body as motion evolves.

This approach allows the cells connectivities to remain constant therefore, avoids interpolation errors in moving mesh problems which has a direct influence on the accuracy and efficiency of the simulations.

It was found that using a computational mesh facili-
tated the grid management, when the boundaries are simplified for translation and rotational motions.

Moreover, since the cells slide over the boundaries as much as one boundary edge at each motion step, GCL can be applied on the physical mesh correctly. In other words, the time rate of change of the total volume of the physical domain remains constant.

Finally, the decomposition of movement applies equally well in three dimensional space, and the proposed approach could therefore be extended to 3D. However, the simplification process, whereby complex three dimensional objects are represented by simpler objects in computational space, becomes much more difficult to handle in 3D. Work is currently underway to generalize this aspect of the method to 3D.

REFERENCES


Figure 15: Sliding the cells around the boundaries in computational and physical domain at $t_{110}$


