IMMERSED BOUNDARY SURFACE METHOD IN FOAM-EXTEND

HRVOJE JASAK[1,2]
[1]*Wikki Ltd, United Kingdom, h.jasak@wikki.co.uk*
[2]*University of Zagreb, Croatia,hrvoje.jasak@fsb.hr*

Keywords: *Immersed Boundary, Non-Conforming Mesh, Direct Cell Cutting, Unstructured Mesh, Finite Volume, OpenFOAM*

This paper describes a new method of handling non-conforming immersed boundaries within the background mesh with minimal level of approximation. The immersed boundary surface is represented by a triangulated surface mesh which interacts with the polyhedral background body-fitted mesh only at the level of implicit and explicit discretisation operators. While no new cells or faces are introduced in the geometry, the Immersed Boundary Surface method accounts for the presence of a non-conforming patch in the manner consistent with the “cut cell” approach. Surface data such as pressure or viscous stress is evaluated on an adaptively coarsened representation of the triangulated surface, created via intersection with the background mesh. The immersed boundary surface method is validated against body-fitted meshes for laminar, turbulent and free surface flows on static and moving mesh/surface cases and deployed with a new implementation of adaptive mesh refinement.

Introduction

The existing implementation of the Immersed Boundary Method (IBM) In `foam-extend-4.0` [1] has proven to be versatile, but with some drawbacks. The most significant are loss of accuracy in the vicinity of the Immersed Boundary (IB), handling of Neumann boundary conditions which cannot be fully implicit, wall function implementation with sampling of turbulence variables beyond a single-cell depth and others.
In free surface flows, the IBM solver exhibits strong instability next to the IB surface, which has proven to be particularly cumbersome. The crux of the problem is the inability of the IBM to match gradients of different variables at the IB, and polynomial fitting of the Volume of Fluid (VoF) field which contains a step-function.
Having failed to stabilise the IBM sufficiently for industrial-grade CFD simulations, our objective is to provide a robust and efficient Immersed Boundary capability for naval hydrodynamics simulations, specifically for motion of floating objects in confined spaces. To achieve this, handling of IB patches is re-examined from first principles.

Existing Immersed Boundary Method

Existing implementation of the Immersed Boundary Method (IBM) is based on polynomial fitting of the solution with respect to the boundary condition and the internal field supporting stencil:

$$\begin{aligned}\phi_P = \phi_{ib} + C_0(x_P - x_{ib}) + C_1(y_P - y_{ib}) \\ + C_2(x_P - x_{ib})(y_P - y_{ib}) + C_3(x_P - x_{ib})^2 + C_4(y_P - y_{ib})^2,\end{aligned} \tag{1}$$

where P and ib denote the cell centre and the IB point, respectively (see Figure 1). Equation 1 requires sufficient resolution in front of the IB point, matching of the IB patch and background mesh resolution and *a-priori* assumption that a variable can be fitted by a polynomial In general none of these hold.
The IBM implementation relies on the imposition of the boundary condition in the bulk of the mesh: this is built into the discretisation matrix at the location of the IB cell centre. While this standard manner of handling IB boundaries may sound adequate, the implementation loses information in the cut cell: reduction in cell volume; loss of precision at the intersection, choice of the supporting stencil for polynomial fitting, handling of the mesh motion fluxes *etc.*
While functionally correct, the idea of polynomial fitting is only adequate for smoothly varying solution variables, but clearly fails for *e.g.*. turbulent wall functions or a a step-profile variables such as the volume fraction α in free surface flows. Attempts to use the Level Set free surface capturing and the Ghost Fluid Method for interface handling [2] proved superior over the original method, but not completely satisfactory.
Further issues with the existing IBM have come to the fore. The most notable is related to evaluation of forces on IB surfaces, as required eg. by a 6-DOF floating body simulation in a free surface flow, which is the main objective of this work. The IB force data can only be correctly evaluated on a continuous and closed STL surface, but whose resolution may vary substantially from the background mesh. Under such conditions, inaccuracy in interpolation from volumetric background mesh data to the IB surface mesh may be significant and further degrade the solution.

Finally, the problem of a moving IB surface is more severe than previously thought. It is necessary to consistently handle the moving wall boundary condition, requiring a manner inconsistent with the conventional moving deforming Finite Volume Method (FVM) [3].
In conclusion: further improvement in robustness and accuracy is sought regarding the precision and stability of discretisation at the IB surface via a completely new approach: The Immersed Boundary Surface (IBS) Method.

New Algorithm: Immersed Boundary Surface

Starting from first principles, we shall discard the idea of polynomial interpolation into IB points which is the basis of existing IBM. Instead, we shall account for the influence of the IB within the mesh as if the mesh is body-fitted, similar to the way a GGI patch imitates the presence of actual cut faces via interpolation factors [4]. The procedure consists of the following steps:

- Introduce the "new" IB face in the cut cell;
- Account for the partial cell volume and face area of affected cells / faces without loss of accuracy; and
- Perform conventional body-fitted FVM discretisation on a partial cell / partial face geometry,

but without changing the geometric mesh at all!
The conventional body-fitted FVM method operates on the following geometrical data:

- Cell-to-face connectivity: owner/neighbour addressing;
- Cell volumes and face area vectors;
- Face interpolation factors and delta-coefficients (1/distance).

For static mesh cases, no further geometrical data is required.
For cells that are unaffected by the IBM or located within the IB surface, conventional discretisation applies without correction. Inactive cells/faces are removed from the matrix by setting their volume/area to zero, without the need for interpolation or stabilisation, as they are completely detached from the region of interest. A cell which intersects with the IB surface is recognised by the fact that some part of it remains on the "live side" of the surface. While this cell is modified, its (topological) connectivity with the rest of the mesh remains unchanged. It however, yields an extra boundary face, representing the IB surface cut.
A schematic approach of the existing IBM and new IBS implementation is shown in Figure 1.

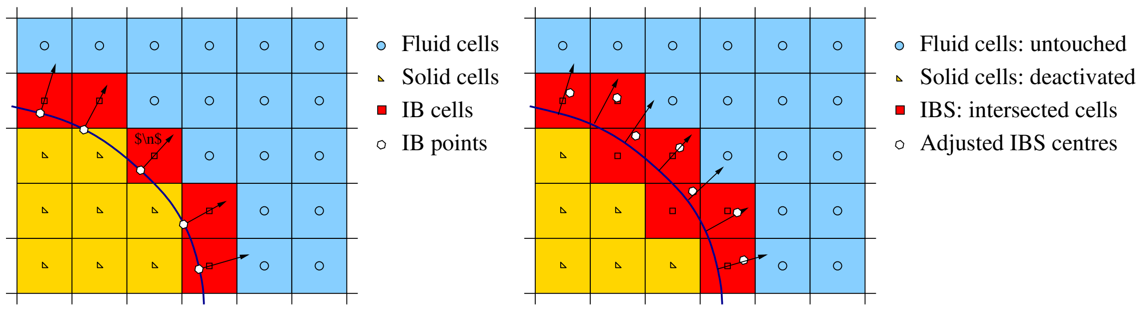

Figure 1: Comparison of the old and new Immersed Boundary Algorithm: dead, interpolated and live cells.

Such an approach has numerous advantages. Firstly, the IB patch is included into the mesh via the distance function: all cells that straddle the immersed boundary remain active and their cell volume and face areas are adjusted using the exact geometrical operations. Secondly, the resolution of the IB STL patch or its quality are unimportant: only the nearest distance data is used. Thirdly, underlying mesh connectivity remains the same after cutting and cut cell re-uses discretisation matrix slot of the original cell. Finally, new faces are created as 1-per-cut-cell and contained in the immersed patch: IB patch face-cell addressing identifies cut cells. In such an arrangement, conventional FVM discretisation can be used without modification or polynomial fitting. The topological connectivity and the sparseness pattern of the background mesh are preserved.

Implementation of the Immersed Boundary Surface Algorithm

The matrix entry for the cut (IB) cell shall represent a geometry which is significantly different from the background cell, and which shall be calculated by cutting with the IB surface. For the cut cell we shall introduce a new IBS face. Cell volume, centroid, face centres, area and interpolation factors for intersected faces are calculated by intersecting the cell and its faces with the IB, via a distance functions, yielding perfect intersection.

The original cell volume and face area entries in the `fvMesh` class for intersected cell/face elements of the background mesh shall be adjusted by intersection and represent the "live" part of cell/face, with appropriate adjustment in the cell/face centroid and norm.
Imposition of boundary conditions on the IB patch is straightforward and consistent with the body-fitted FVM. Immersed boundary is represented in the mesh by STL intersection with cells and its resolution is adjusted "automatically": by intersection with the background mesh. In other words, the STL IB surface is automatically refined/coarsened to comply with the background mesh without intervention.

On static meshes, the IB patch supports regular boundary conditions without change. For moving mesh / moving IB surface cases, the IB cut shall be re-calculated in response to mesh motion, yielding a different number of cut cells (equal to the number of IB patch faces) and different face-cell connectivity. In such cases, the definition of the IB boundary shall be attached to the IB STL surface and interpolated to currently live cut faces. This requires a custom `immersedBoundaryFvPatchField` boundary condition, which also handles data I/O and evaluation of IB patch field data.
Some degenerate cases need to be handled. A "regular intersection" occurs between a cell and the IB surface when there are no exact point-to-point hits between the IB and background mesh, which cannot be assumed. Due to finite accuracy, STL regularly coincides with background mesh points or faces, or does not provide accurate intersection. An example would be a direct face intersection between a Dry and a Wet cell, where the background mesh face in fact becomes the IB face.

Degenerate Intersections

Inaccurate intersection of STL feature edges/points may yield a geometrically open cell, with possible robustness issues. This is resolved using the trick known as the Marooney Manoeuvre [5] to guarantee a closed cell after cutting:

$$\sum_C \mathbf{s}_f = \mathbf{0} \qquad \text{for a regular cell,} \tag{2}$$

$$\sum_C \gamma_f \mathbf{s}_f + \mathbf{s}_{f\,IB} = \mathbf{0} \qquad \text{for an intersected cell,} \tag{3}$$

where γ_f is the area correction, obtained by cutting. The corrected IB face area $\mathbf{s}_{f\,IB}$ is:

$$\mathbf{s}_{f\,IB} = -\sum_C \gamma_f \mathbf{s}_f. \tag{4}$$

Moving Immersed Boundary Surface

Moving mesh Finite Volume Method uses a compensated form of the transport equation for the arbitrary moving/deforming volume, introducing mesh motion fluxes $F_b = \S_f \bullet \mathbf{u}_b$, where U_b is the mesh velocity calculated directly from the deforming geometry. The motion is deemed correct, if the Space Conservation law is satisfied:

$$\int_V \frac{\partial V}{\partial t} - \oint_S (\mathbf{n} \bullet \mathbf{u}_b)\, dS = 0. \tag{5}$$

On body-fitted meshes, its first-order accurate discretised form reads:

$$\frac{V^n - V^o}{\Delta t} - \sum_f F_b = 0. \tag{6}$$

For moving IB surface cases (assuming either static or moving background mesh), the volume swept by the immersed face needs to be evaluated consistently. However, since the IB-to-cell connectivity is no longer clear, a special practice, consistent with the moving mesh FVM is needed.

Volume swept by immersed face is calculated from the new IB face geometry and motion distance Figure 2 and distributed into the affected cells:

$$V_b = \mathbf{x}_b \bullet \mathbf{s}_f{}^n \ \text{ volume swept in red,} \tag{7}$$

where $\mathbf{x}_b$ is geometrical motion distance from old to new STL As the intersected area changes, swept volume must be distributed,Figure 2:

- Cell A: expand from cut volume to full volume: no IB face in new configuration;
- Cell B: cut both at new and old configuration: IB face present;
- Cell C: expand from zero volume to partial volume: IB face present;

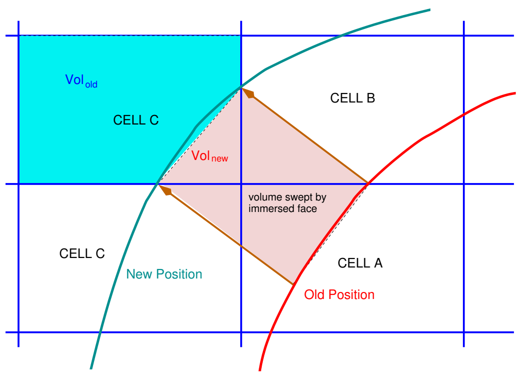

Figure 2: Moving Immersed Boundary Surface: mesh motion fluxes.

- (Cell D): expand from zero to full volume (cell fully swept by IB): no IB face.

To guarantee that the Space Conservation Law, Equation 5 is satisfied, the "Motion Flux manoeuvre" is performed, following the idea of the original Marooney Manoeuvre:

1. If the cell is "dead" at new level, no compensation is needed: $V^n = 0$
2. If the cell is "live" at new level, compensate by adjusting cell volume to keep both V^o and V^n positive: **Cell B, Cell C**
   - Perform virtual re-meshing at old time-level
   $$V^o = V^n - \Delta t \sum_f V_b \; if \; V^o >= 0$$
   - Otherwise, perform virtual re-meshing at new time level:
   $$V^n = V^o + \Delta t \sum_f V_b \; if \; V^n >= 0$$

This yields a consistent and conservative mesh motion fluxes for all cases.

Validation on Turbulent and Free Surface Flows

A number of laminar and turbulent flow cases has been studied with excellent results, some of which are shown in Figure 3.

Conclusion and Future Work

A new method for handling non-conformal Immersed Boundary interfaces has been implemented and validated. The Immersed Boundary Surface method accounts for the presence of the non-conforming boundary by directly changing the discretisation matrix to mimic the form it would have if the IB faces were included directly into the background mesh. The method has been implemented in `foam-extend-4.1`, parallelised and validated in laminar and turbulent flow and multi-phase free surface simulations. It is regularly used with the IB-sensitised polyhedral adaptive mesh refinement, which locally enriches the background mesh in the vicinity of the IB surface to the required level

References

[1] U. Senturk, D. Brunner, H. Jasak, N. Herzog, C. Wodled, and A. Smits, "Benchmark simulations of flow past rigid bodies using an open-source, sharp interface immersed boundary method," *Progress oin CFD*, 2018.

[2] V. Vukčević, H. Jasak, and I. Gatin, "Implementation of the Ghost Fluid Method for Free Surface Flows in Polyhedral Finite Volume Framework," *Comput. Fluids*, vol. 153, pp. 1–19, 2017.

[3] v. Tuković and H. Jasak, "A moving mesh finite volume interface tracking method for surface tension-dominated interfacial fluid flow," *Computers and Fluids*, vol. 55, pp. 70–84, 2012.

[4] M. Beaudoin and J. H., "Development of a generalized grid mesh interface for turbomachinery simulations with openfoam," in *Open Source CFD International Conference, Berlin*, December 2008.

[5] C. Marooney, "The Marooney Manoeuvre," Private Communication.

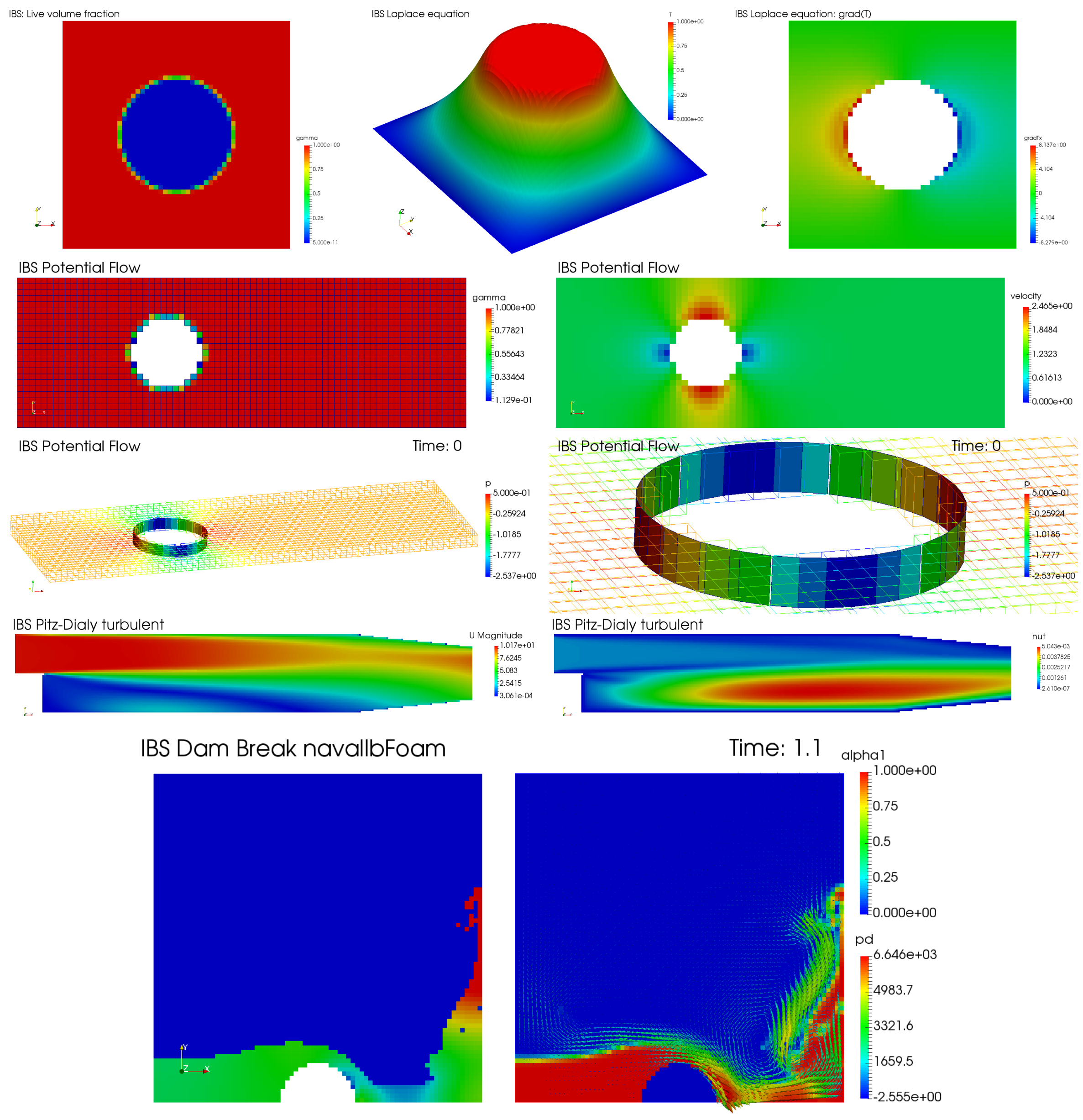

Figure 3: Validation of the Immersed Boundary Surface method for laminar, turbulent and free surface flows.