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

FEDSM-ICNMM2010-31097

AN IMMERSED BOUNDARY METHOD FOR GENERAL FLOW APPLICATIONS

Jack R. Edwards
North Carolina State University
Raleigh, North Carolina, USA

Jung-IL Choi
North Carolina State University
Raleigh, North Carolina, USA

Santanu Ghosh
North Carolina State University
Raleigh, North Carolina, USA

Daniel A. Gieseking
North Carolina State University
Raleigh, North Carolina, USA

Jeffrey D. Eischen
North Carolina State University
Raleigh, North Carolina, USA

ABSTRACT

The development of a direct-forcing immersed-boundary method for general flow applications is outlined in this paper. A cell-classification procedure based on a signed distance to the nearest surface is used to separate the computational domain into cells outside the immersed object ('field cells'), cells outside but adjacent to the immersed object ('band cells'), and cells within the immersed object ('interior cells'). Interpolation methods based on laminar / turbulent boundary layer theory are used to prescribe the flow properties within the 'band cells'. The method utilizes a decomposition of the velocity field near embedded surfaces into normal and tangential components, with the latter handled using power-law interpolations to mimic the energizing effects of turbulent boundary layers. A procedure for directly embedding sequences of stereo-lithography files as immersed objects in the computational is described, as are extensions of the methodology to compressible, turbulent flows. Described applications include human motion, moving aerodynamic surfaces, and shock / boundary layer interaction flow control.

INTRODUCTION

Immersed-boundary methods are a general class of technique that indirectly imposes the effects of a (possibly moving) solid surface on the surrounding flow. While the original immersed-boundary method dates from the work of Peskin (1), the technique was recast into a form more useful for conventional CFD strategies by Mohd-Yosuf (2), Verzicco et al. (3), Fadlun et al. (4) and others. A review article summarizing these and other techniques is that of Mittal and Iaccarino (5). A key to these newer immersed-boundary methods is the enforcement of fluid boundary conditions indirectly, through specification of the distribution of the fluid velocity in the vicinity of the immersed boundary.

This paper surveys some recent results obtained using a generalization of an immersed-boundary method developed for time-dependent, incompressible flows in Choi et al. (6). This approach is similar to that of Gilmanov et al. (7) in that a surface mesh consisting of structured or unstructured elements is embedded within a flow and that flow-property variations normal to the surface are reconstructed. The surface meshes may be closed (surrounding a volume of space) or zero-thickness (surfaces alone). The Navier-Stokes equations are solved in cells outside the body ('field cells'); a constant property condition is enforced for cells inside the body ('interior cells'); and boundary conditions are enforced through specifying distributions of fluid properties in a collection of 'band' cells just outside the immersed body ('band' cells). In contrast to many other IB techniques, the methods developed in Choi et al. (6) can be applied to turbulent flows at high Reynolds numbers by virtue of its use of power-law interpolation techniques to mimic the near-wall profile of an attached turbulent flow. The methods are also applicable to general curvilinear meshes as well as unstructured meshes. Since the publication of Choi et al. (6), extensions to particle-laden incompressible flows (8) and compressible, turbulent flows (9) have been developed. This report presents these developments as a unified framework and outlines a new way of embedding stereo-lithography files as immersed objects within a computational domain. Applications ranging from human motion activity to flow control of shock / boundary layer interactions are presented.

IMMERSED BOUNDARY METHOD

In the present work, an immersed surface is generated either as a cloud of arbitrarily-ordered points or as a

stereolithography (STL) file. The entire flow domain is then classified into three categories of cells. Cells sufficiently removed from the immersed boundary are termed as 'field' cells, cells very near but not inside the immersed object are 'band' cells, and cells inside the immersed body are 'interior' cells. To perform this classification, a distance function to the nearest surface location for all cells (within a 'bounding box' of the IB) is computed. The distance-function calculation differs depending on whether a point-cloud or an STL object is considered and is described later.

A 'direct forcing' approach is used to enforce the boundary conditions at the interior and band cells. This results in the residual form of the Navier-Stokes equation system shown below which is then solved implicitly, coupled with exterior cells, by use of sub-iteration techniques.

$$R_i^{n+1,l} = (1 - G(\Phi^{n+1}))R_{i,NS}^{n+1,l} + G(\Phi^{n+1})\left[\frac{V_i^{n+1,l} - V_{B,i}^{n+1,l}}{\Delta t}\right] = 0 \qquad (1)$$

This equation represents the blending of the Navier-Stokes residual with a source term that relaxes the primitive variable vector V to its band-cell values. The quantity $G(\Phi)$ is a sharp Heaviside function (set to 1 for 'band' and 'interior' cells and zero otherwise), Φ is the signed distance function, and l is a sub-iteration index. Field cells are defined as those with $G = 0$ and $\Phi > 0$, while band cells are those with $G = 1$ and $\Phi < 0$, and interior cells are those with $G = 1$ and $\Phi < 0$. The Navier-Stokes equations are solved in the field cells, fluid properties are generally held fixed in the interior cells, and analytic forms for the fluid properties (discussed later) are prescribed in the band cells.

CELL CLASSIFICATION PROCEDURE

3D Surface Definition in a Computational Domain

Most popular way to describe 3D objects in computer system is to construct triangle meshes. This can be done using a Computer Aided Design (CAD) format or though other means, but key is that triangle elements with outward pointing normal vector are created for each separate component of the objects, as different components may move at different rates. The next step is to define 3D surfaces using the *unsigned* distance and classification whether an arbitrary point in a background domain is inside or outside of the objects. The classification can be achieved to count the intersections of a ray going from the given point (outside point from the objects) to infinity since the number of intersections must be odd if the point is inside, which is called as *ray tracing method* (10). Another classification is to define a signed distance using the inner product between a pseudo normal vector and a distance vector to an arbitrary point from its closest point on the surface, which is called as *signed distance computation* (11, 12). While the former method needs to visit the parts of the triangle mesh along the ray tracing line, the latter algorithm needs to find the closest point on the mesh. We will apply the signed distance computation which is faster than ray tracing method in order to define 3D surfaces which will be incorporated with the present immersed boundary method.

The distance d from grid points $\boldsymbol{x}_g$ in a computational domain Ω_C to the closest surface point $\boldsymbol{x}_s$ on triangle meshes Γ^l for l^{th} component is simply defined as $d = \|\boldsymbol{x}_g - \boldsymbol{x}_s\|$ in Figure 1. Computation of the distance to 3D objects can be achieved by using brute force computation, Voronoi diagram (13) and hierarchical data structure (14, 15). Among these methods, we will use a *k*d-tree hierarchical data structure with a bounding box to accelerate finding the nearest triangle mesh. For simplicity, we consider the one component closed surface as shown in Figure 2. At first, we find a cloud of nearby points $\boldsymbol{x}_i^v$ from the given point $\boldsymbol{x}_g$ in a bounding box, in order of the closest distance, using approximate nearest-neighbor (ANN) searching algorithm (16). Next step is to search the closest point in the set of the neighbor triangle meshes $\Delta_i^j \in \Gamma_i$ which are sharing with a cloud of nearby vertex $\boldsymbol{x}_i^v$ since the closest vertex is typically different from the closest point on a triangle mesh. We can define the subset $\Gamma_s = \{\Gamma_i\}$ of the total triangle meshes Γ. Based on the subset Γ_s, minimum distance can be obtained using point-triangle, point-edge and point-vertex distance calculation.

In the search process, the subset Γ_s can be reduced using geometric restriction. The recent CAD programs enhance the uniformity of the triangle and control the edge distance. At a given edge distance d_e, we can get a restriction for the searching algorithm. As shown in Figure 2, the circles show the spheres with radius d_e and origin $\boldsymbol{x}_i^v$. The entire triangle neighbors Δ_i^j shared with the vertex $\boldsymbol{x}_i^v$ are included within the spheres. The distances d_i^j in the subset Γ_i are bounded as $|d_i^k - d_i| \le d_e$ with respect to the point-vertex distance d_i. Also the distance is $|d_1^j - d_1| \le d_e$ for the subset Γ_1 which is equivalent subsets for the minimum point-vertex distance. The difference between two point-vertex distances can be written as $d_i^j - d_1^k - 2d_e < d_i - d_1 < d_i^j - d_1^k + 2d_e$. If $d_i^j < d_1^k$, the difference should be bounded as $d_i - d_1 < 2d_e$. Therefore, the above nearest distance calculation should be repeated for the i^{th} nearest vertex point in ANN list which satisfies $d_i - d_1 < 2d_e$.

Signed Distance Computation

The signed distance function $\hat{d}$ can be obtained by multiplying the unsigned distance with the sign of the dot product of the distance vector with the outward normal vector $\boldsymbol{n}$:

$$\hat{d} = \mathrm{sgn}((\boldsymbol{x}_g - \boldsymbol{x}_s) \cdot \boldsymbol{n})\, d\,, \qquad (2)$$

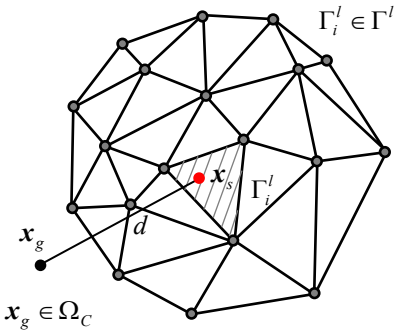

Figure 1: Schematics for a minimal distance from the points in a computation domain to surface points.

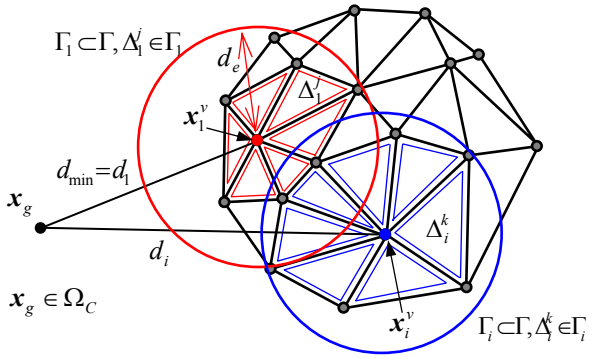

Figure 2: Schematics for nearest neighbors of tirangle elements.

where $\mathrm{sgn}(\phi)$ returns a value of 1 for each non-negative element and –1 for each negative element of ϕ, and $\|\ \|$ denotes the magnitude of the vector.

This simple procedure was found not to work properly for some very complex CAD objects (6). Usually, the CAD objects are defined as triangle surface elements that contain each vertex and face normal vector. If a nearest surface point at a given field point is located in an edge or vertex, the simple signed distance function may not be calculated correctly. Therefore, we consider an angle-weighted pseudo-normal vector (12), which is defined at surface nodes (vertices) or edges, rather than cell centers of surface triangles. For a given vertex $\boldsymbol{x}_v$, we can define the triangle elements shared with the vertex and calculate the incident angle α_i for each element with outward-pointing face normal vector $\boldsymbol{n}_i$ (6). The angle weighted pseudo-normal vector $\boldsymbol{n}_v$ at the vertex can be defined as,

$$\boldsymbol{n}_v = \frac{\sum_i \alpha_i \boldsymbol{n}_i}{\left\| \sum_i \alpha_i \boldsymbol{n}_i \right\|}, \qquad (3)$$

where i denotes the triangle elements that surround the vertex. Based on the pseudo-normal vector at the vertex and face normal vector $\boldsymbol{n}_i$ at the element center $\boldsymbol{x}_i$, we can determine an inside/outside decision using the same signed distance function in Eq. (2) with the data set of the vertices. This procedure essentially averages local fluctuations in the outward normal

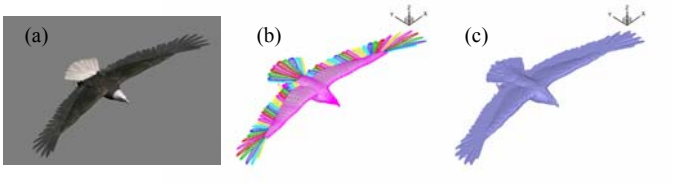

Figure 3: Illustrations of (a) Eagle image in 3DS Max®, (b) colored segments, and (c) rendering in computational domain.

that could result from small features in the CAD file.

To define a global signed distance function $\bar{d}$ at any given mesh point, a simple priority rule is exercised. First, the global distance function is initialized to a large number. Then, the global signed distance function at a particular point is taken as the minimum of the individual signed distance functions for each component l at that point:

$$\bar{d} = \min_l(\bar{d}_l) \qquad (4)$$

The collections of points that comprise the surfaces are allowed to move according to prescribed rate laws.

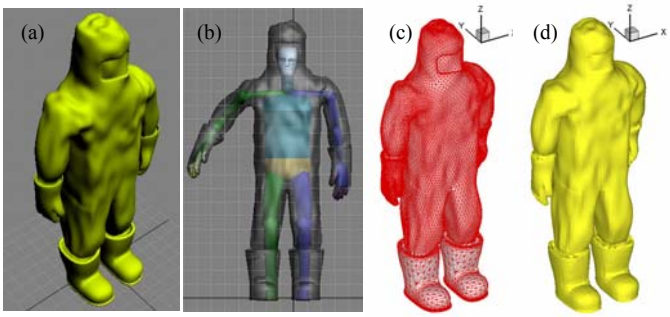

Figure 4: (a) Rendering in 3DS Max®, (b) skinned skeleton, (c) triangle elements in STL files, and (d) rendering in a computational domain

Embedding of CAD Objects as Immersed Surfaces

Common three-dimensional objects in a computer system can be written in stereo lithography (STL) file format. One of our major goals is to be able to incorporate general STL files as immersed surfaces in our flow solver without any additional user intervention. We used the 3DS Max® (Autodesk, Inc.) as a platform for generating STL files describing general objects, especially for generating sequential STL files for person (avatar) motions with inverse kinematics.

Following step-by-step procedure is used for identifying the multiple-objects or multiple-segments from one single STL file in a computational domain:

1. Import STL file; Store position vectors at three vertices and outward-pointing normal vector for each element.
2. Determine element-to-element connectivity; Make a list for the associated elements to each vertex. Define the connectivity where elements contain a common edge.

3. Find segments; Make a list for closed or open surfaces linked all elements based on the element-to-element connectivity
4. Calculate pseudo-normal vectors for all of the vertices, and edges and face-normal vectors for all of the elements using Eq.(3).
5. Given a computational cell, use ANN searching to determine a set of nearest vertices in the immersed objects.
6. Calculate signed distance based on the point-to-elements, point-to-edges, and point-to-point distance with the pseudo-normal and face-normal vectors.

We exercised the segmentation procedure for a CAD rendering of Eagle object (from 3DS Max) which contains 75 segments. Figure 3 shows that the present STL reader separates clearly the object into 75 segments and the immersed object is successfully retendered in a computational domain. Figure 4 shows an example of a person in hazmat suit, rendered using 3DS Max® and its skinned skeleton which is essential part for motion generation using inverse kinematics based on biped model. Figure 4 (c) and (d) show 63504 triangle elements representing the person and rendering in a computational domain with an iso-surface with zero value of signed distance function.

Band Cell Interpolations

The following first-order accurate closures are used for the fluid properties in the band cells, where the subscript 'I' indicates properties obtained at an interpolation point located along the normal line extending outward from the nearest surface location corresponding to the band cell in question (discussed later), and the subscript "B" indicates the band-cell.

$$
\begin{aligned}
&p_B = p(d_I)\\
&u_{B,i} - u_{S,i} = u_{T,i}(d_I)\left(\frac{d_B}{d_I}\right)^k + u_{N,i}(d_I)c(\rho, d_I, d_B),\\
&u_{N,i}(d_I) = (u_j(d_I) - u_{S,j})\mathbf{n}_j\mathbf{n}_i,\\
&u_{T,i}(d_I) = (u_i(d_I) - u_{S,i}) - u_{N,i}(d_I)
\end{aligned}
\tag{5}
$$

In these expressions, $\mathbf{n}_i$ is the normal vector at the closest point on the body surface, d is a distance from the nearest surface point, $u_{S,j}$ is the velocity at the nearest surface point, and k is a power-law. The choice of k allows the model to replicate a turbulent velocity profile (k=1/7 or 1/9) or a laminar profile (k=1). To obtain the temperature distribution near the surface, Walz's relation for the temperature distribution within a compressible boundary layer is used (17):

isothermal wall:

$$
\begin{aligned}
\frac{T_B}{T(d_I)} = \frac{T_w}{T(d_I)} &+ \left(1 - \frac{T_w}{T(d_I)} + \frac{r(\gamma-1)}{2\gamma RT(d_I)}[u_{T,i}(d_I)]^2\right)\left(\frac{d_B}{d_I}\right)^k\\
&- \frac{r(\gamma-1)}{2\gamma RT(d_I)}[u_{T,i}(d_I)]^2\left(\frac{d_B}{d_I}\right)^{2k}
\end{aligned}
\tag{6}
$$

adiabatic wall:

$$
\frac{T_B}{T(d_I)} = 1 + \frac{r(\gamma-1)}{2\gamma RT(d_I)}[u_{T,i}(d_I)]^2\left(1 - \left(\frac{d_B}{d_I}\right)^{2k}\right) \tag{7}
$$

In this, r is the recovery factor and $[u_{T,i}(d_I)]^2$ is the kinetic energy associated with the tangential velocity component at the interpolation point.

The function $c(\rho, d_I, d_B)$ that scales the normal velocity component in Eq. (5) is determined by enforcing a discrete form of the continuity equation at each band cell using a locally-parallel flow assumption. Details are given in Ghosh et al. (9). The result, for an adiabatic wall, is given as

$$
\begin{aligned}
&c(\rho, d_I, d_B) = \frac{1}{\tilde{\rho}}\frac{\frac{d_B}{d_I}d^-\tilde{\rho}^-}{\left(\frac{d_B}{d_I}d^-\tilde{\rho}^- + (1 - \frac{d_B}{d_I})\tilde{\rho}^+\right)},\\
&d^- = \left(\frac{d_B}{2d_I}\right)^k, \quad d^- = \left(\tfrac{1}{2}(1 + \frac{d_B}{d_I})\right)^k,\\
&\frac{1}{\tilde{\rho}} = 1 + \frac{r(\gamma-1)}{2\gamma RT(d_I)}[u_{T,i}(d_I)]^2\left(1 - \left(\frac{d_B}{d_I}\right)^{2k}\right),\\
&\frac{1}{\tilde{\rho}^-} = 1 + \frac{r(\gamma-1)}{2\gamma RT(d_I)}[u_{T,i}(d_I)]^2(1 - (d^-)^2),\\
&\frac{1}{\tilde{\rho}^+} = 1 + \frac{r(\gamma-1)}{2\gamma RT(d_I)}[u_{T,i}(d_I)]^2(1 - (d^+)^2),
\end{aligned}
\tag{8}
$$

Note that this procedure does not rigorously enforce mass conservation within the band cells, as the integral form of the continuity equation is not used. If precise mass conservation is required, the pressure interpolation in Eq. (5) can be replaced by the solution of the continuity equation in the band cells.

The turbulence variables in the band cells are defined as

$$k_B = u_\tau^2 / \sqrt{C_\mu}\ ,\ \omega_B = u_\tau / (\sqrt{C_\mu}\kappa d_B): \quad d^+ > 10.934$$

$$k_B = k(d_I)\left(\frac{d_B}{d_I}\right)^2,\ \omega_B = 60\nu_w / (0.075 d_B^2): \quad d^+ < 10.934 \qquad (9)$$

$$d^+ = u_\tau d_b / \nu_w, \quad u_\tau = |u_{T,i}(d_I)| / (\ln(d^+)/\kappa + 5.1)$$

(iterative solution)

To arrive at this form, we assume equivalence between the result provided by the power-law profile and the law of the wall within the band cells. For incompressible flows, the temperature distribution in Eq. (5) may be neglected, or if variable-temperature effects are important, the terms proportional to the kinetic energy in Eq. (5) can be neglected. The normal-velocity scaling for incompressible flows sets $\tilde{\rho}$, $\tilde{\rho}^-$, and $\tilde{\rho}^+$ to unity.

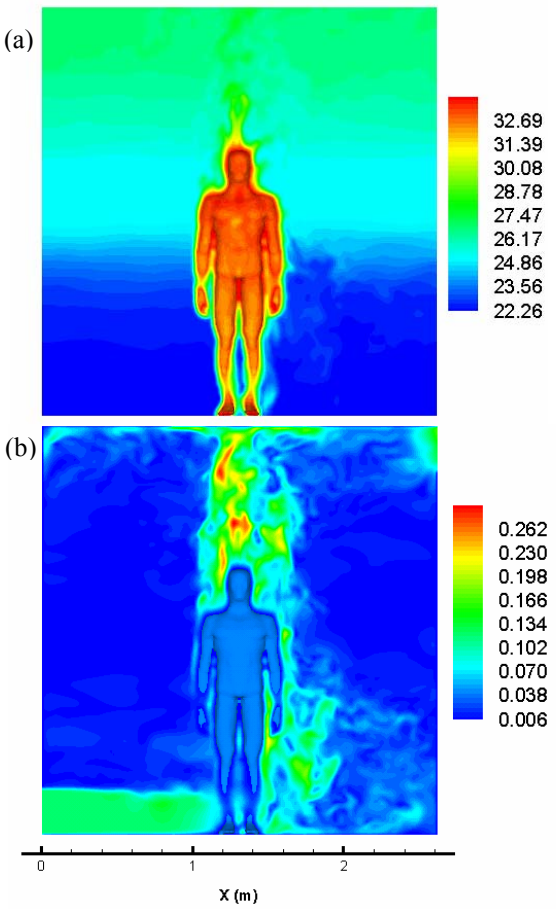

Figure 5: Velocity and temperature contours for forced convection over a human; (a) temperature and (b) vertical velocity

Determination of Information at the Interpolation Point

The preceding developments hinge on the determination of flow properties $\hat{q}(d_I)$ at a certain distance d_I away from the surface. Given a point within the band $\vec{x}_k$ and a list of nearest neighbors to that point $\vec{x}_l$, a merit function w_l is defined as

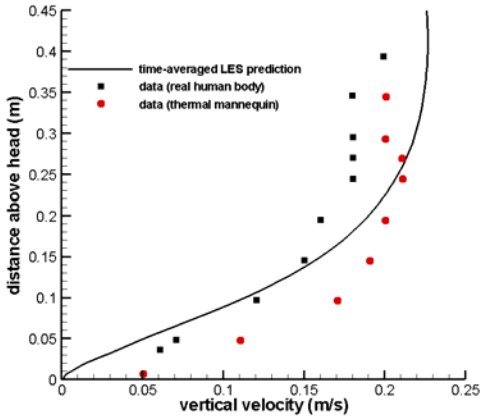

Figure 6: Vertical velocity versus distance above head – forced convection over a human body

$$w_l = \frac{1}{\sqrt{(|\vec{x}_l - \vec{x}_k|)^2 - ((\vec{x}_l - \vec{x}_k)\cdot\hat{n}))^2} + \varepsilon},\ (\vec{x}_l - \vec{x}_k)\cdot\hat{n} > 0, \qquad (10)$$

$$w_l = 0 \ \text{ otherwise}$$

In this, $(\vec{x}_l - \vec{x}_k)\cdot\hat{n}$ is the projection of the distance from $\vec{x}_k$ to $\vec{x}_l$ in the direction of the outward normal, and $|\vec{x}_l - \vec{x}_k|$ is the magnitude of the distance vector itself. If point $\vec{x}_l$ is located directly along the outward normal line corresponding to band point $\vec{x}_k$, and if $(\vec{x}_l - \vec{x}_k)\cdot\hat{n}$ is positive, meaning that point $\vec{x}_l$ is further away from the surface than point $\vec{x}_k$, then the merit function returns a very large value (~$1/\varepsilon$, where ε is 10^{-12})

The actual calculation of w_l is performed in three stages. First, only field points (those with $\Phi(\vec{x}_l, t) > 0$ and $G(\Phi(\vec{x}_l, t)) = 0$) are considered as members of the list of nearest neighbors. Then, w_l is calculated according to Eq. (9), and the sum of the weights $\sum_m w_m$ is calculated. If this sum is non-zero, then the actual weight function for each nearest-neighbor is determined as

$$\omega_l = \frac{w_l}{\sum_m w_m}. \qquad (11)$$

Otherwise, the process is repeated, now considering both field points and other band points as members of the list of nearest neighbors. If this application also results in no viable interpolation points being found, then the band point $\vec{x}_k$ is effectively set to an interior point.

The location at which interpolated properties are defined, d_I, is calculated for a particular field point as

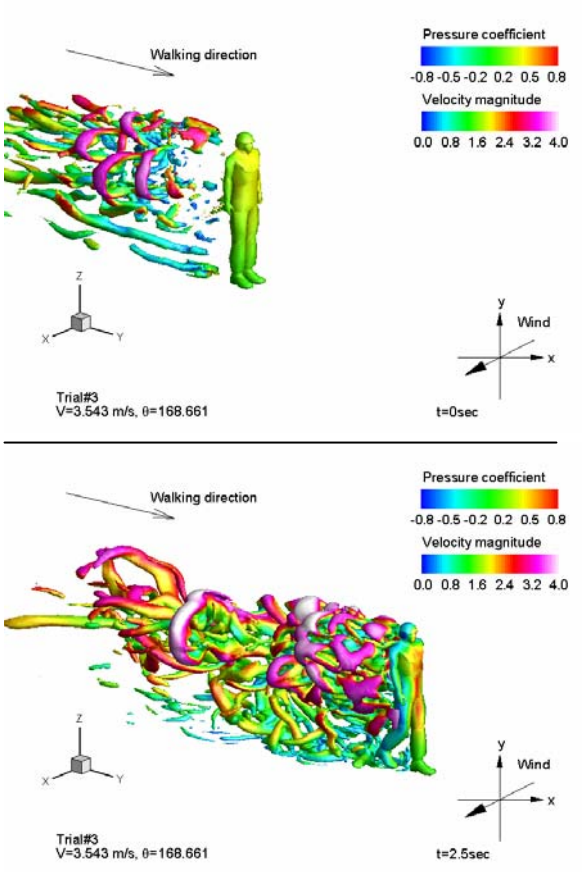

Figure 7: Vortical structures (colored by velocity magnitude) and surface pressure distributions for Trial 1 (top: initial; bottom: after 2.5 seconds of walking

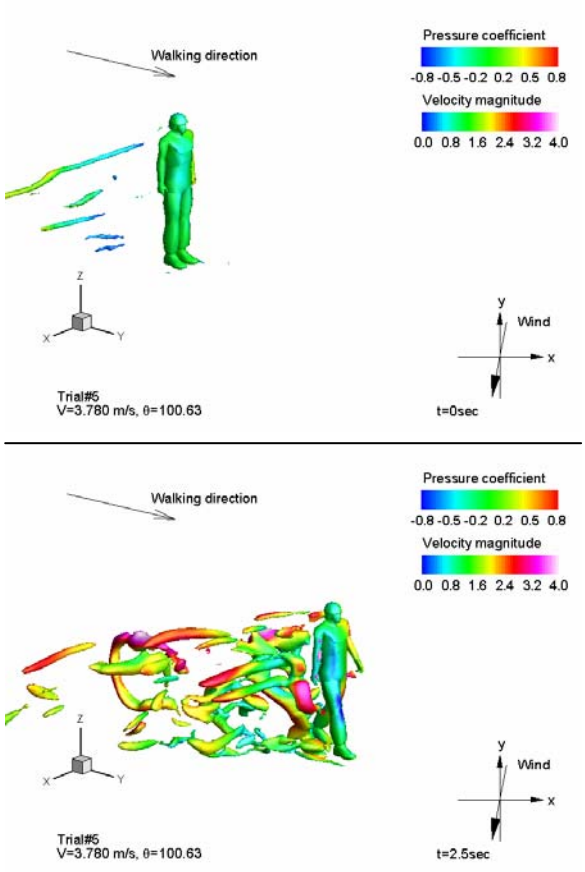

Figure 8: Vortical structures (colored by velocity magnitude) and surface pressure distributions for Trial 2 (top: initial; bottom: after 2.5 seconds of walking

$$d_I = \sum_l \omega_l (\vec{x}_l - \vec{x}_k) \cdot \hat{n} \,. \tag{12}$$

Note that this distance is in the direction of the normal coordinate. With this, the fluid properties $\hat{q}(d_I)$ are found by applying the weighting functions

$$\hat{q}(d_I) = \sum_m \hat{q}_m \omega_m \,. \tag{13}$$

APPLICATIONS

Human Motion

The IB method described in the earlier sections has been applied routinely to calculate the effects of human activity on the surrounding flow. Large-eddy simulation techniques have been used in these simulations, with a general metric of mesh resolution being ~(2 cm)^3 in the vicinity of a moving person. Applications have focused primarily on understanding the effects of the human wake in entraining and transporting fine particles or tracer gases. Several examples are described in the following paragraphs. Figures 5-7 correspond to simulations

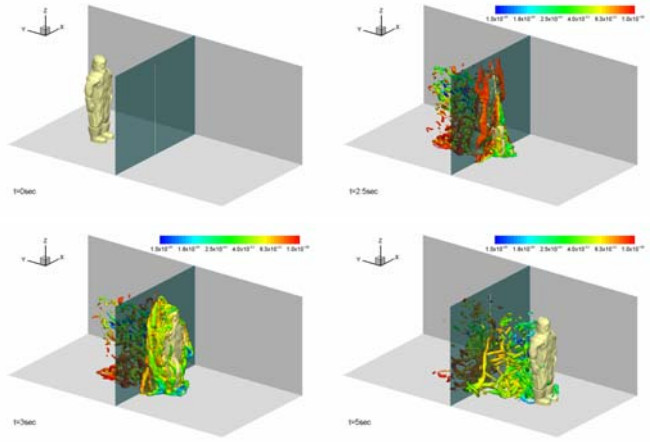

Figure 9: Snapshots of vortical structures generated by the interaction between the person and fabric model during an entry event.

of an experiment of Hayashi, et al. (18) involving forced convection in a 2.6 x 2.2 x 2.2 m^3 room. This calculation solves a decoupled form of the energy equation to provide the temperature field and uses the Boussinesq approximation to account for buoyancy due to thermal gradients. In the experiment, a small vent forces cooler (22 °C) air into the room at a rate of 0.12 m/s. The jet impinges upon a stationary human

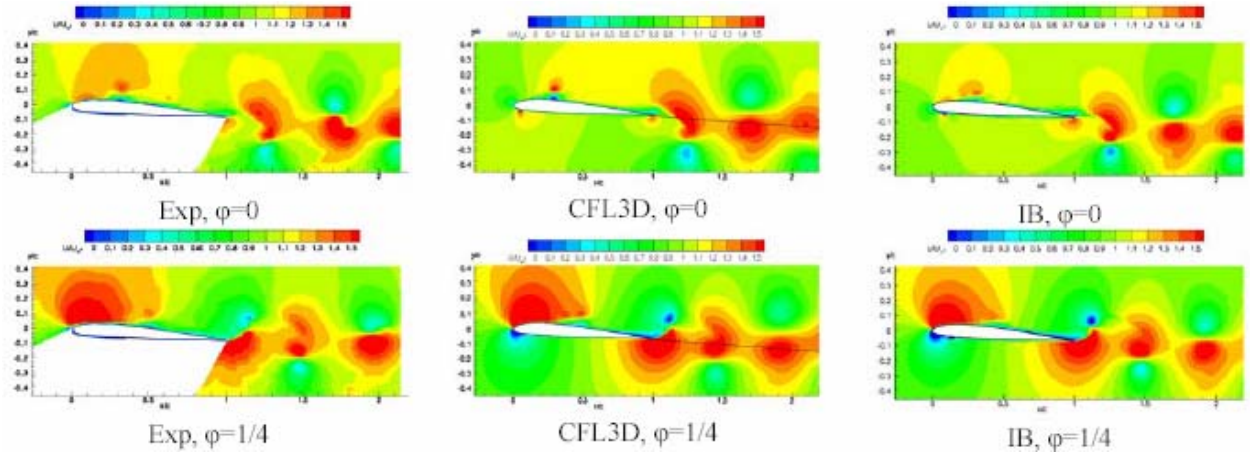

Figure 10: Axial velocity comparisons at two phase angles within a plunge sequence (from McGowan et al. (19))

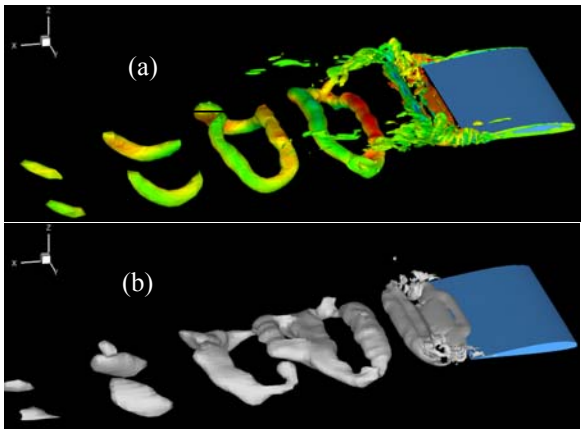

Figure 11: Snapshots of flow field generated by plunging motion of a rectangular wing; (a) 5 s^{-1}swirl-strength iso-surface and (b) 0.4 m/s velocity magnitude iso-surface.

standing in the center of the room and exits through an exhaust port in the ceiling. Experimental data consists of vertical velocity measurements above the head of the human. The updraft is induced by the combined effects of the vent system and the buoyant thermal plume of the human. In the calculations (performed on a mesh with ~4.4 million cells), the surface temperature of the human was set to 34 C, and the computations were continued for 540 s (2/3 of one air exchange time). Figures 5 shows X-Z slices of the instantaneous velocity and temperature distributions, while Figure 6 shows a prediction of the average vertical velocity induced above the head of the human.

Another example involves flow over a moving human in a wind field. The avatar's motion (a soldier) is obtained as a sequence of STL files from the 3DSmax® (Autodesk, Inc.) software package. Two sets of wind speeds and directions are considered. (Figure 7 - Trial 1 and Figure 8 - Trial 2) The top components of the figures show the initial wakes generated as the wind flows over the static body. The bottom components illustrate the wake development as the avatar reaches its maximum walking speed of 1 m/s.

The IB method has also been loosely coupled with various structural response solvers. In the example illustrated in Figure 9, a computation of the avatar (a person in a HazMat suit) interacting with a flexible sheet with a slit is modeled off-line using LS-DYNA. The output is a sequence of STL files for the moving avatar and the fabric. The immersed surfaces corresponding to the avatar are treated as closed bodies, while those corresponding to the fabric are treated as zero-thickness surfaces. In the case of zero-thickness surfaces, there is no 'outside' and 'inside', and the direction of the surface normal vectors is arbitrary. This does not affect the calculation of properties within the band cells, but there are no interior cells to be classified for a thin body. The properties of the fabric were chosen to mimic tent canvas. Figure 9 shows the response of the fabric motion and wakes induced by person and fabric interaction. The iso-surfaces represent vortical structures colored by the velocity magnitude. The person's impact forces the fabric pieces apart. Once he moves through, the fabric returns to its nominal position. LS-DYNA uses a penalty method to enforce the contact constraint between the avatar and flexible wall. This in effect places a stiff spring and an associated force between avatar and wall at any contact points. A scale factor is employed that determines the magnitude of the force. Too small a force results in interpenetration (and potential leaking flow) of the avatar through the wall, while too large a force results in numerical instability. For a scale factor of 800 the enforcement of the contact constraint is near perfect with no penetration or instability observed.

Moving Aerodynamic Surfaces

One potential advantage of an IB method in aerodynamics is the ease in which it can be adapted to handle both general body motion and the relative motion of other components (such as slats or flaps) without the need for expensive re-gridding. Figure 10 (from McGowan et al. (19)) shows an example of the use of the IB technique in computing unsteady plunging

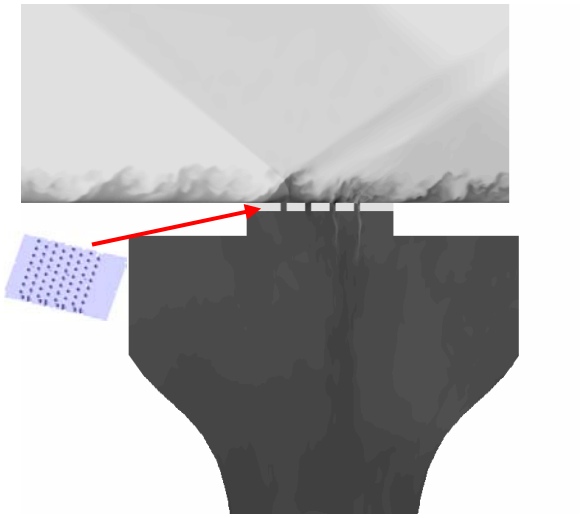

Figure 12: Snapshot of temperature contours with inset bleed plate: shock / boundary layer interaction with bleed (from Ghosh et al. (20))

motion of an airfoil at a Reynolds number of 60000. The motion is sinusoidal, and the time-averaged results at different phase angles are compared with phase-locked PIV

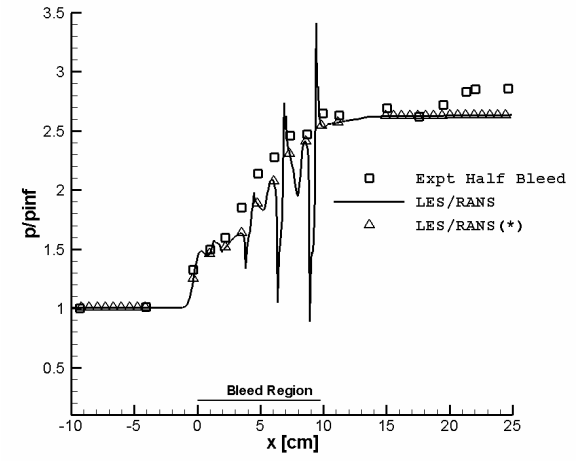

Figure 13: Wall temperature distributions for shock/boundary layer interaction with bleed.

measurements and with results from the CFL3D flow solver, which used a body-fitted, moving grid approach. Good agreement between the sets of results is indicated. Snapshots of vortices induced during a 3-D finite-span simulation (aspect ratio = 2) at the same conditions (Figure 11) provides evidence of a complex merging of tip-generated vortices with those generated due to the plunging motion. Vortical structures within the airfoil boundary layer have been blanked out to delineate the geometry more clearly. This simulation was performed on a mesh containing upwards of 14 million cells – the resolution increase being primarily due to the need to resolve the region of the flow affected by the wing's motion in a nearly isotropic manner.

Compressible-Flow Applications

The IB method described herein has been used to investigate the effects of various types of devices used to control shock-wave / boundary layer interactions. Devices considered in these studies include micro vortex generators (9), arrays of bleed holes (20), and deformable meso-flaps (21). The primary advantages of an IB method in this scope include the ease in which different kinds of geometric devices can be embedded in the domain and their effects assessed, reduced resolution requirements near the control device (provided that the band-cell interpolations function as desired), and the possibility of incorporating aero-elastic effects without mesh adaptation. For some devices (e.g. bleed hole arrays), it is critical to ensure mass conservation in the band cells located just outside the immersed object. In these cases, we replace the pressure interpolation to the surface with the direct solution of the continuity equation in the band cells. Calculations of these high-speed flowfields have either used Reynolds-averaged turbulence modeling methods (Menter's SST model (22)) or a hybrid large-eddy / Reynolds-averaged Navier-Stokes method (23). In both cases, the implementation of the IB for turbulent flow is as described in the Section "Band Cell Interpolation" Figure 12 (from Ghosh et al. (20)) shows an example of a calculation of an oblique shock / turbulent boundary layer interaction at Mach 2.75. The bleed array consists of 64 hollow cylinders along with the supporting flat-plate structure and is rendered as an immersed object in the computational domain. Instantaneous temperature contours of along the centerline are shown at a bleed-rate condition ½ of the maximum considered in the experimental study (24). Time-averaged surface pressure distributions (Figure 13) are in good agreement with experimental data.

CONCLUSIONS

An immersed boundary method suitable for general flow simulations has been presented. The model is grid-topology independent and is based on the decomposition of a computational domain into cells inside an immserd body ('field' cells), cells outside but adjacent to an immersed body ('band' cells), and cells far away from an immersed body ('field' cells). Interpolation methods based on turbulent boundary layer theory are used to connect the flow solution in 'band' cells to specified surface boundary conditions and to the solution of the Navier-Stokes equations in the field cells. The approach differs from others in the literature in its use of power-law forms for the near surface velocity field,. This enables the method to mimic the energizing effect of a turbulent boundary layer without excessive near-surface resolution. Results for a variety of compressible and incompressible flow problems involving both static and moving immersed object have been presented. Current efforts are focused on resolving a particular issue with the method that relates to the need (in some cases) of enforcing rigorous mass conservation in all cells exterior to an embedded surface. While this can be easily accomplished by integrating the

continuity equation in band cells, the resulting pressure field is often oscillatory and in some cases, can corrupt the external flow solution. Other research issues relate to the performance of the method for zero-thickness immersed surfaces. While the current procedures extend easily to these situations, the performance of the model is not yet ideal, particularly for object motion in three dimensions.

ACKNOWLEDGMENTS

This work has been supported by DARPA (HR0011-04-0057, HR0011-05-C-0157), U.S. EPA (4C-R138-NAEX, EP06C000195), and AFOSR (FA9550-07-1-0191), and is currently supported by the Naval Surface Warfare Center, Dahlgren Division (N001178-08-C-3030) and by the Air Force Office of Scientific Research (FA9550-10-1-0120).

REFERENCES

(1) Peskin, C.S., (1972) "Flow Patterns Around Heart Valves: A Numerical Method", *Journal of Computational Physics*, Vol.10, pp. 220-252., 1972

(2) Mohd-Yosuf, J. "Combined Immersed Boundary / B-Spline Methods for the Simulation of Flow in Complex Geometries" *Annual Research Briefs of the Center for Turbulence Research*, pp. 317-328, 1997.

(3) Verzicco, R., Mohd-Yusof, J., Orlandi, P., and Haworth, D. "LES in Complex Geometries Using Boundary Body Forces" *AIAA Journal*, Vol. 38, pp. 427-433. 2000.

(4) Fadlun, E.A., Verzicco, R., Orlandi, P., and Mohd-Yusof, J. "Combined Immersed-Boundary / Finite-Difference Methods for Three-Dimensional Complex Flow Simulations," *Journal of Computational Physics*, Vol. 161, pp.35-60, 2000.

(5) Mittal, R., and Iaccarino, G. "Immersed Boundary Methods", *Annual Review of Fluid Mechanics*, Vol.37, pp. 239-261, 2005.

(6) Choi, J.-I., Oberoi, R.C., Edwards, J.R. and Rosati, J.A. "An Immersed Boundary Method for Complex Incompressible Flows", *Journal of Computational Physics*, Vol.224, pp.757-784, 2007.

(7) Gilmanov, A., Sotiropoulus, F., and Balaras, E. "A General Reconstruction Algorithm for Simulating Flows with Complex 3-D Immersed Boundaries on Cartesian Grids" *Journal of Computational Physics*, Vol. 191, pp. 660-669, 2003.

(8) Choi, J-I. and Edwards, J.R. "Large Eddy Simulation and Zonal Modeling of Human-Induced Contaminant Transport" *Indoor Air*, Vol. 18, pp. 233-249, 2008.

(9) Ghosh, S., Choi, J.-I., Edwards, J.R. "Numerical Simulation of Effects of Micro Vortex Generators Using Immersed Boundary Methods", AIAA Journal, Vol. 48, No. 1, pp. 92-103, 2010.

(10) J. Linhart, "A Quick Point-in-Polyhedron Test" *Computers & Graphics*, Vol.14, No.3, pp.445-448, 1990.

(11) H. Gouraud, "Continuous Shading of Curved Surfaces", *IEEE Trans. Computers*, Vol.20, No.6, pp.623-629, 1971.

(12) J. A. Bærentzen and H. Aanæs, "Signed Distance Computation Using the Angle Weighted Pseudonormal", *IEEE Trans. Visualization & Computer Graphics*, Vol.11, No.3, pp.243-253, 2005.

(13) K.E. Hoff, T. Culver, J. Keyser, M. Lin and D. Manocha, "Fast Computation of Generalized Voronoi Diagrams using a Graphics Hardware", *SIGGRAPH'99 Proc*., pp.277-285, 1999.

(14) B.A. Payne and A.W. Toga, "Distance Field Manipulation of Surface Models", *Computer Graphics & Applications*, Vol.12, No.1, pp.65-71, 1992.

(15) A. Guéziec, "Meshsweeper: Dynamic Point-to-Polygonal-Mesh Distance and Applications", *IEEE Trans. Visualization & Computer Graphics*, Vol.7, No.1, pp.47-61, 2001.

(16) S. Arya and D.M. Mount, N.S. Netanyahu, R. Silverman, A.Y. Wu, "An Optimal Algorithm for Approximate Nearest-Neighbor Searching", *Journal of the ACM*, Vol. 45, pp.891-923, 1998.

(17) Walz, A. Boundary Layers of Flow and Temperature (English translation), MIT Press, 1969.

(18) Hayashi, T., Murakami, S., and Sakuma, K., "Study on Computational Thermal Mannequin Part 14 Measurement of Flow and Temperature Fields Around Real Human Body and Thermal Mannequin" *Technical Papers of Annual Meeting*, The Society of Heating, Air-Conditioning, and Sanitary Engineers of Japan, Sapporo, Japan, pp. 985-988, 1998.

(19) McGowan, G.Z., Gopalarathnam, A., Ol, M.V., Edwards, J.R., and Fredberg, D. "Computation vs. Experiment for High-Frequency Low Reynolds Number Airfoil Pitch and Plunge" AIAA Paper 2008-0653, 2008.

(20) Ghosh, S., Choi, J-I., and Edwards, J.R. "Simulation of Shock / Boundary Layer Interactions with Bleed using Immersed Boundary Method", *Journal of Propulsion and Power*, Vol. 26, No. 2, pp. 203-214, 2010.

(21) Ghosh, S., Choi, J.-I., and Edwards, J.R. "Numerical Simulation of the Effects of Mesoflaps in Controlling Shock / Boundary Layer Interactions", AIAA Paper 2010-4465, 2010.

(22) Menter, F.R. "Two Equation Eddy Viscosity Turbulence Models for Engineering Applications," *AIAA Journal*, Vol. 32, No. 8, pp. 1598-1605, 1994.

(23) Edwards, J.R., Choi, J-I., and Boles, J.A. "Hybrid Large-Eddy / Reynolds-Averaged Navier-Stokes Simulation of a Mach-5 Compression Corner Interaction", *AIAA Journal,* Vol. 46, No. 4, pp. 977-991, 2008.

(24) Willis, B.P., Davis, D.O., and Hingst, W.R. "Flowfield Measurements in a Normal-Hole Bled Oblique Shock Wave and Turbulent Boundary Layer Interaction", AIAA Paper 95-2285, 1995.