HEFAT2010
7th International Conference on Heat Transfer, Fluid Mechanics and Thermodynamics
19-21 July 2010
Antalya, Turkey

A Comparison Between Blasco-Chorda And Haselbacher-Najjar Algorithms In Computation Of Particle Tracking In a Flow Field

Emami M.D.*, Saidi M.S. and Safaei H.
*Author for correspondence
Department of Mechanical Engineering,
Isfahan University of Technology,
Isfahan, 84156-83111,
Iran
E-mail: mohsen@cc.iut.ac.ir

ABSTRACT

Numerical Simulation of multiphase flows may be performed in the framework of Eulerian, Lagrangian, or a hybrid formulation. The present paper adopts the hybrid scheme to numerically simulate particle tracking in fluid flow. The governing equations of mass and momentum for the fluid phase are written in the Eulerian frame, and the particle equations are described in a Lagrangian frame. The numerical descretization is performed, using the finite volume approach in an unstructured grid. Particle equations are solved through standard ODE solvers. The current paper focuses on the assessment of two efficient tracking algorithms. Results show that the Blasco-Chorda algorithm is more suitable in terms of computational time when fine grids (both triangular and quadrilateral) are used for the simulation.

NOMENCLATURE

r	[m]	Location
r(t)	[m]	Location in time t
x, y	[m]	*Component of location*
x(t), y(t)	[m]	Component of location in time t
P	[-]	Vertex of polygon
n	[-]	Outward unit normal of the face
tr	[-]	Trajectory
t, Δt		Time and time step
dt	[m]	Trajectory distance
d	[m]	diameter
ρ	[kg/m^3]	Density
u,v	[m/s]	Components of Velocity
U	[m/s]	Dimensionless velocity
V	[m/s]	Dimensionless velocity component perpendicular to the main coordinate direction
p	[pa]	Static pressure
I		Unit tensor
g	[m/s^2]	Gravitational force
k	[m^2/s^2]	Kinetic energy
S	[m/s^2]	Mean strain rate
T	[-]	Dimensionless time
D	[-]	Dimensionless diameter
Stk	[-]	Stokes number
Rho	[-]	Density ratio
Fr	[-]	Froude number
G	[1/s]	Perpendicular gradient
C_{df},C_s, Csgn	[-]	Coefficient terms
R		main difference between the standard k–ε model and the RNG k–ε model
Special characters		
τ	[pa]	Stress tensor
μ	[pa.s]	Dynamic viscosity
ε	[m^2/s^3]	*Dissipation rate*
Subscripts		
I		Vertex number
p		particle
f		fluid
c		Centroid of the face
eff		Effective viscosity
t		turbulance
o		Mean Inlet velocity

INTRODUCTION

An important issue in the numerical solution of the hybrid Eulerian-Lagrangian model is to predict the particle location, especially in complex geometries and unstructured grids. The particle-locating algorithm plays an important role in improving computational efficiency for Eulerian-Lagragian computations of two-phase flows. The Lagrangian trajectory solver may become very time consuming as a result of the overloaded particle-locating work. Generally speaking, there are four approaches proposed to locate particles in irregular grids, i.e. Cartesian background grid method [1], mapping techniques [2,

3], tetrahedral walk method [4, 5], and face-by-face search method [6-11]. The first two schemes are based on locating particles in a regular grid instead of the original irregular (physical) grid. However, there are various deficiencies associated with these methods. For example, inaccuracy and inefficiency may rise in the calculation of particle trajectory on the basis of the Cartesian background grid method due to large variations in grid size, and the mapping method is limited to structured grids [12]. It has been recognized that such deficiencies do not exist in the tetrahedral-walk and face-by-face search methods, which perform point location directly in a physical grid. The tetrahedral walk method, which locates a particle based on a tetrahedron enclosed by triangular plane surfaces, involves a sophisticated tetrahedral decomposition in structured curvilinear grids containing hexahedral cells or unstructured hybrid grids containing mixed element types [13]. On the contrary, the face-by-face search method approximately treats non-planar faces as plane surfaces. With such approximation, the point-locating scheme is simple and easy to implement, but it may "force" particles to be trapped near non-planar faces. In order to obtain a good CPU-time performance with enough robustness, as summarized by Lohner [4] and Vaidya et al. [10], various approaches have been proposed to optimize the search path of point location.

Among the most prominent particle-locating methods are the ones proposed by Zhou and Leschziner [7] (ZL), Chen and Pereira [6] (CP), Blasco-chorda [8] and Haselbacher et al.[11]. The most interesting feature of the ZL algorithm is that it can be implemented very straightforwardly, and requires very small CPU times [14]. However, due to its simplicity, the ZL locating algorithm traverses through a larger number of cells than strictly required, and can even get trapped in an infinite circular search around the particle cell. The method introduced by Chen and Pereira [6], also referred as to the directed search, only examines the cells passed through by a considered particle, and is thus very beneficial to the determination of some crucial information for modeling multiphase flow, such as particle–wall interaction and particle residence time [6, 10, 14]. However, it suffers from some important disadvantages, such as large CPU times in a general particle-locating context, and not too-clear extension to 3D grids due to the non-planar faces, as pointed out by Chordá et al. [8].

Regarding the aforementioned points about the advantages and drawbacks of these particle-tracking methods, we consider only the Blasco-chorda and Haselbacher Najjar algorithms [8,11].

PARTICLE-SEARCH ALGORITHMS

This section presents a brief summary of the two selected particle-tracking algorithms.

1. The Blasco-Chorda Algorithm (BC Algorithm)

The performance of the proposed particle-locating strategy is compared with existing ones, and is evaluated on two tests.

1.1. particle-to-the-left test

The algorithm defines an arbitrary 2D convex polygon P by giving the Cartesian coordinates (x, y) of its vertices ordered anticlockwise.

The particle-to-the-left condition, which will be referred to as P2L, can be checked for each cell face by looking at the z component of the cross product between the face vector P_IP_{I+1} and the particle vector $P_Ir_P(t+\Delta t)$, see Fig1.

$$\Omega_I = (x_{I+1} - x_I)(y_p(t+\Delta t) - y_I) - (x_p(t+\Delta t) - x_I)(y_{I+1} - y_I) \quad (1)$$

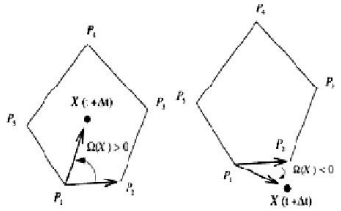

Fig .1. Examples of the P2L test

- $\Omega_I > 0$ indicates that the point P_I is on the left-hand side of the cell face.
- $\Omega_I < 0$ indicates that the point is on the right-hand side of the face.
- $\Omega_I = 0$ indicates that the point is on the face.

1.2 Trajectory-to-the-left test

The face–trajectory intersections detect by the z component of the cross product of the vectors $r_P(t)P_I$ and $r_P(t)r_P(t+\Delta t)$.

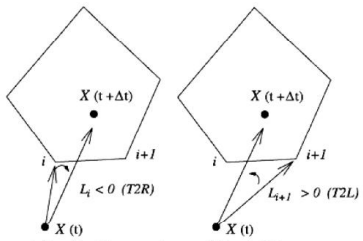

Fig. 2. Illustration of the T2L test

With refer to Fig.2, the expression of the z component of vertex I is:

$$L_I \equiv (x_I - x_p(t))(y_P(t+\Delta t) - y(t)) - (x_P(t+\Delta t) - x_P(t))((y_I - y_P(t)) \quad (2)$$

if L>0 the particle trajectory lies to the left of a given vertex and if L<0 otherwise. For this reason, the computation of the value of L_I will be termed as trajectory-to-the-left (T2L) test. The BC algorithm has the following steps:

1. Check if the particle trajectory $r_P(t)r_P(t+\Delta t)$ crosses one face of the current cell. This is done by applying the T2L test (Eq. (2)) to the two face vertices and comparing the sign of the T2L test.
2. If it does, check the P2L test on that face. If the particle lies to the right of the face (P2L<0), then we have found the appropriate (i.e., exit) crossing face. Exit the loop and move to the neighbouring cell that shares that face.
3. Move to the next face and go to step 1.

4. If the loop over all the cell faces is finished without fulfilling step 2 then the particle lies within the current cell.
The face–trajectory intersection would be impossible to compute in the event of having a particle trajectory crossing through a cell vertex. This would make the T2L test fail in the case of having an L = 0 value for I or I + 1.
In order to account for this (highly improbable) situation, the following modification to the face-intersection is suggested: assume that the face I-(I+1) is crossed if $L_I L_{I+1} \leq 0$.

2. The Haselbacher-Najjar Algorithm (HN Algorithm)
The algorithm is based on tracking a particle along its trajectory by computing the intersections of the trajectory and the cell faces.

2.1 Particle-Localization Algorithm
The basic idea of the present particle-localization algorithm is the following: Assume that the particle is known to be located in cell c_1 and to move along a given trajectory. Assume further that we can determine which face of cell c_1 is intersected by the particle trajectory. If the cell adjacent to the intersected face is c_2, the particle must pass from cell c_1 into cell c_2. By applying this idea repeatedly, we can determine the cell c_n which contains the predicted new particle position. A cell is said to contain a particle location $\mathbf{r}_p$ if this position satisfies the so-called "in-cell test," i.e., if for each face of the cell,
$(\mathbf{r}_c-\mathbf{r}_p).n \geq 0$ (3)

The whole tracking steps may be summarized as:
1 Compute trajectory distance and trajectory:

$$dt = \left\| \mathbf{r}_p(t+\Delta t) - \mathbf{r}_p(t) \right\| \quad (4)$$

$$\mathbf{tr} = (\mathbf{r}_p(t+\Delta t) - \mathbf{r}_p(t))/dt . \quad (5)$$

2 Compute trajectory-face intersections and then the minimum intersection of them and the corresponding face (α_I, α_{min}, f_{min}) in the current cell. Note that it is not necessary to calculate intersections for faces that $\mathbf{tr}.\mathbf{n}<0$:

3 $$\alpha_I = \frac{(\mathbf{r}_c - \mathbf{r}_p(t)).\mathbf{n}}{\mathbf{tr}.\mathbf{n}} \quad (6)$$

4 $$dt \leftarrow dt - \alpha_{min}$$

5 If $dt \geq 0$ then if f_{min} is not a boundary face get adjacent cell, else if this face is a boundary face reflect for solid wall. Go to step 2
6 If $dt<0$ then this cell containing the new position of particle.
7 Go to step 1

3.1 Simulation of the flow field
The model used for an isothermal flow simulation solves the conservation equations for mass (or continuity) and momentum. The turbulence in the system is solved through a modified form of the two-equation model. Under steady-state conditions, the equations for mass and momentum in a general form are as follows:

$$\nabla . \rho_f \vec{u}_f = 0 \quad (14)$$

$$\nabla . (\rho_f \vec{u}_f \vec{u}_f) = -\nabla p + \nabla . (\vec{\tau}) + \rho \vec{g} = 0 \quad (15)$$

where

$$\vec{\tau} = \mu_{eff} [(\nabla u_f) - \frac{2}{3} \nabla . \rho \vec{u}_f I] \quad (16)$$

where
$\mu_{eff}=\mu+\mu_t$.

The renormalization group (RNG) k–ε model is used to model turbulence effects [13]. This method is similar in the form to the standard k–ε model but includes additional terms for ε development that significantly improve the accuracy, especially for rapidly strained flows.
The kinetic energy and dissipation rate are obtained from the transport equations given below:

$$\frac{\partial(\rho_f k)}{\partial t} + \frac{\partial(\rho_f k u_{fi})}{\partial x_i} = \frac{\partial}{\partial x_j} \left[((\mu + \frac{\mu_t}{\sigma_k}) \frac{\partial k}{\partial x_j} \right] + G_k - \rho_f \quad (18)$$

$$\frac{\partial(\rho_f \varepsilon)}{\partial t} + \frac{\partial(\rho_f k \varepsilon)}{\partial x_i} = \frac{\partial}{\partial x_j} \left[((\mu + \frac{\mu_t}{\sigma_\varepsilon}) \frac{\partial k}{\partial x_j} \right] \quad (19)$$

$$+ C_{1\varepsilon} \frac{\varepsilon}{k} (G_k) - C_{2\varepsilon} \rho_f \frac{\varepsilon^2}{k} (G_k) - R$$

$$G_k = -\overline{u'_{fi} u'_{fj}} \frac{\partial \overline{u}_{fi}}{\partial x_j} \quad (20)$$

The model constants $C_{1\varepsilon}$, $C_{2\varepsilon}$, C_μ, σ_k and σ_ε were assumed to have the following values [13]:

$\sigma_\varepsilon = 1.3$, $\sigma_k = 1.0$, $C_{2\varepsilon} = 1.92$, $C_{1\varepsilon} = 1.44$, $C_\mu = 0.0845$.

The term R is defined as:

$$R_{ij} = \frac{\rho_f C_\mu \eta_{ij} \left(1 - \frac{\eta_{ij}}{\eta_0}\right) \frac{\varepsilon^2}{k}}{1 + \beta \eta^3} \quad (21)$$

where η_0, β are constants assumed to have the following values [13]: η_0 –4.38, β–0.012

The term η_{ij} is defined as:

$$\eta_{ij} = S_{ij} k / \varepsilon \quad (22)$$

where S_{ij} is expressed as:

$$S_{ij} = \left(\frac{\partial u_{fi}}{\partial x_j} + \frac{\partial u_{fj}}{\partial x_i} \right) \quad (23)$$

The turbulence viscosity is then calculated as:

$$\mu_t = \rho C_\eta \frac{k^2}{\varepsilon} \quad (24)$$

where C_η =0.09 is a model constant.

3.2. The particle equations
The equation of motion for a particle in a Lagrangian framework in dimensionless form may be written as:

$$\frac{dU_p}{dT} = C_{df} \frac{(U_f - U_p)}{Stk} + (1 - \frac{1}{Rho}) \frac{1}{Fr^2} + \frac{0.72 C_{sgn} C_s}{(Rho\ Stk)^{1/2}} (U_f - U_p) \left| \frac{\partial V_f}{\partial X_b} \right| \quad (25)$$

where the dimensionless parameters are calculated using u_{fo}, the mean inlet fluid velocity and a length scale h, which depends on the problem dimensions:
$U_{f,p} = u_{f,p} / u_{fo}$.

$T = tu_{fo} / 2h$.

$D_p = d_p / 2h$.

The force terms of equation (25) on the right-hand side are drag, gravity (and buoyancy) and Saffman lift, respectively.
The coefficient terms in equation (25) are:

$$C_{df} = 1 + \mathrm{Re}_p^{0.687}$$

where C_{df} coefficient used by Clift et al. [15] to modify the drag term for ultra-Stokesian drag and the Reynolds number accounts for the velocity-slip between the particle and fluid .

$$\mathrm{Re}_p = \frac{d_p \rho_f |u_f - u_p|}{\mu_f} = D_p |U_f - U_p| \mathrm{Re}_f$$

Equation (25) has three important particle dimensionless parameters the Stokes number, the density ratio, and the Froude number defined as:

$$Stk = \frac{\rho_p d_p^2 u_{fo}}{18\mu_f (2h)} = Rho_p D_p^2 \mathrm{Re}$$

Small Stokes number particles represent particles with low inertia which tend to follow the flow.

$$Rho = \frac{\rho_P}{\rho_f}$$

Generally, most studies which integrate equation (25) consider a solid-fluid flow which has the condition and all the force terms except drag and gravity can be ignored.

$$Fr = \frac{u_{fo}}{(2hg)^{1/2}} \qquad (26)$$

The Froude number in the present study assumes a constant acceleration of gravity and length scale similar to most experimental studies.
The Saffman lift force was originally derived by Saffman [16] for low Reynolds numbers. In equation (25) the lift force has been simplified and is treated simply as an aerofoil lift effect. The coefficient term C_{sgn} is $G/|G|$ where G is the perpendicular gradient term. Many studies have improved and investigated the Saffman lift force, in view of a study by McLaughlin [17] an additional coefficient has been added C, described as follows:

$$C = \begin{cases} 0 & s < 0.25 \\ s^{5/2} & 0\ s < 0.25 \\ 1 & s > 1 \end{cases}$$

where the parameter s is .

$$s = (Gv_f)^{1/2} / |u_f - u_P|$$

VERIFICATION

To verify the correct implementation of the aforementioned algorithms, a particle is tracked within a carrier-flow executing solid-body rotation with the angular velocity, ω, fixed at unity. The same problem has been studied by Barton [18] and Ruetsch and Meilburg [19], and its attractiveness lies partly in the availability of an analytic solution for the particle's outward spiralling motion associated with its inertia. The particle is released at (x_P, y_P)=(0,1)and has the velocity components(u_P, v_P)=(1,0). The fluid velocity is known at any given position (x, y) in the flow field and can be described as (ω_x, ω_y) The exact solution for the particle path as a function of time is an exponentially increasing spiral whose shape depends on the particle's relaxation time t [18].
Fig.3 shows the analytical solutions of three different particle trajectories for the respective values of relaxation times t =100.0 [s], 1.0 [s] and 0.01 [s] within the time range $0 < t < 2\pi$ [s]. As appears in this figure, both algorithms show accurate predictions compared to the analytical method.

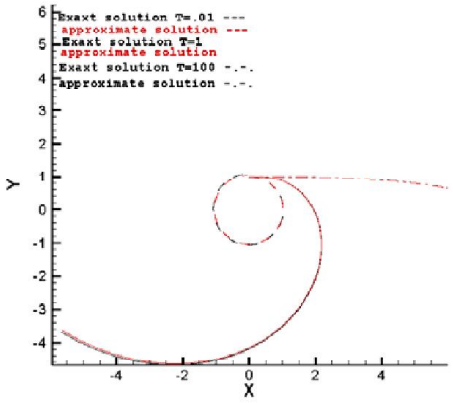

Fig. 3. Trajectories of three particles with different relaxation times within solid-body rotation between t=0 and t=2 π (s)

THE PRESENT CASE

Particle tracking in a backward-facing step flow is considered to assess the two tracking algorithms. The geometry has been studied by Barton [20] and Armaly [21]. In the present study, both quadrilateral and triangular computational cells are used for the simulations. The following non-dimensional parameters are used for the simulation:
Re=547, Rho=10, Stk=0.13, Fr=0.9

A representative particle is injected at point(0, 3/2 h) with the same velocity as that of fluid.
Figure 4 shows the particle trajectory of a sample particle. As appears in the particle trajectory, the recirculation zone behind the step changes the particle track from its straight path.

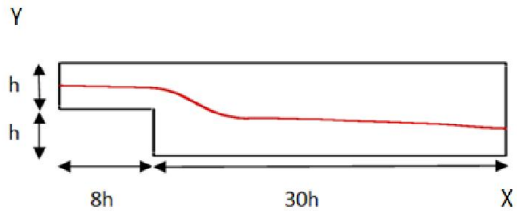

Figure 4. A sample particle trajector

In order to compare the two tracking approaches, 11 different simulations have been performed for quadrilateral and triangular grids, ranging non-dimensional grid sizes from 0.0002 to 0.003. It should be mentioned that a small modification has been done in the implementation of the HN algorithm to increase its performance, namely the face corresponding to minimum distance is evaluated after the fourth step of the algorithm.

As both algorithms predict the same trajectory for a particle, the tracking time is used to compare these two algorithms. Figure 5 compares the ratio of BC to HN computation time for several quadrilateral meshes. The horizontal coordinate shows the typical cell length scale. It is observed that for larger cells, values greater than 0.0003, the tracking time for the BC algorithm is higher than that of HN. But as the cell size becomes smaller, the BC algorithm takes less time to track the particle. This may be attributed to time consumed to perform the internal test, and time consumed for the intersection of the particle and the CV faces in algorithms. The same comparison has been performed for triangular cells, which is depicted in Figure 6.

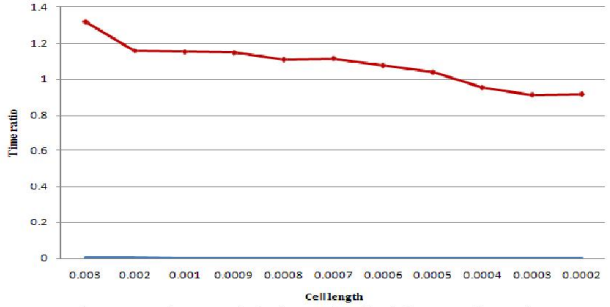

Figure 5.Time ratio(Blasco/Najjar)for quad mesh

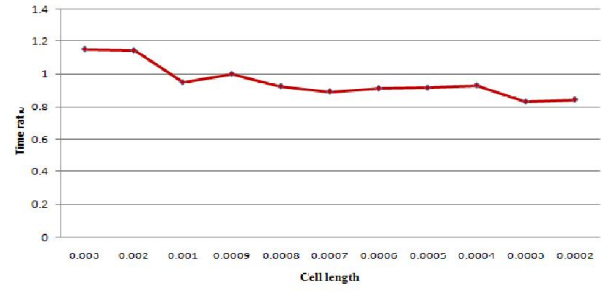

Figure 6.Time ratio(Blasco/Najjar)for triangle mesh

Part of the ratio of calculation time of BC to HN is a function of the ratio of the number of intersections of the particle with CV faces (m) to the total number of steps (N) used for the tracking simulation. By recording these parameters for the numerical simulations, it is found that a ratio of m/N=0.45 prevails for all simulations. We seek to define a non-dimensional parameter to have estimation for this ratio for a general tracking problem. For this reason, we note that the number of intersections between the particle path and the CVs faces is proportional to the characteristic length of particle to the mesh characteristic length. Therefore,

$$m \propto \frac{Ndt}{l}$$

Dividing the above expression by (NΔt):

$$\frac{m}{N} \propto \frac{Ndt}{N\Delta t} \times \frac{1}{l}$$

in which is a measure of the mean velocity of the particle. So it may be written as:

$$\frac{m}{N} \propto \frac{\overline{u}_P}{\frac{l}{\Delta t}}$$

Therefore:

$$\frac{m}{N} = C_1 \frac{\overline{u}_P}{\frac{l_{equad}}{\Delta t}} \qquad \text{for quad mesh} \qquad (24)$$

$$\frac{m}{N} = C_2 \frac{\overline{u}_P}{\frac{l_{etri}}{\Delta t}} \qquad \text{for tiangle mesh} \qquad (25).$$

Parameters C1 and C2 are model constants. Work is underway to find appropriate values for these constants.

CONCLUSIONS

Both tracking algorithms that are implemented show good accuracy and robustness during the computations. It was found from the results that the Blsco-Chorda algorithm may be more time-consuming for tracking in coarse cells, but as the grid is refined, the algorithm seems to work better that the Haselbacher-Najjar. A criterion was set for *a priori* compare these two algorithms, but needs further work to suitably define the model constants.

REFERENCES

[1] Muradoglu M., and Kayaalp A.D., An Auxiliary Grid Method for Computations of Multiphase Flows in Complex Geometries, *Journal of Computational Physics*, Vol. 214, 2006, pp. 858-877.

[2] Schafer F., and Breuer M., Comparison of C-Space and P-Space Particle Tracing Schemes on High-Performance Computers: Accuracy and Performance., *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS*, Vol. 39, 2002, pp. 277-299.

[3] Hægland H., Dahle H.K., Eigestad G.T., Lie K.-A., and Aavatsmark I., Improved Streamlines and Time-of-Flight for Streamline Simulation on Irregular Grids, *Advances in Water Resources*, Vol. 30, 2007, pp. 1027-1045.

[4] Löhner R., Robust, Vectorized Search Algorithms for Interpolation on Unstructured Grids, *Journal of Computational Physics*, Vol. 118, 1995, pp. 380-387.

[5] Sadarjoen I.A., Boer A.J.D., Post F.H., and Mynett A.E., Particle Tracing in Σ-Transformed Grids Using Tetrahedral 6-Decomposition, *Paper presented at the Proceedings of the*

Eurographics Workshop, Blaubeuren, Blaubeuren, Germany 1998.

[6] Chen X.Q., and Pereira J.C.F., A New Particle-Locating Method Accounting for Source Distribution and Particle-Field Interpolation for Hybrid Modeling of Strongly Coupled Two-Phase Flows in Arbitrary Coordinates, *Numerical Heat Transfer, Part B: Fundamentals: An International Journal of Computation and Methodology*, Vol. 35, 1999, pp. 41 - 63.

[7] Zhou Q., and Leschziner M.A., An Improved Particle-Locating Algorithm for Eulerian-Lagrangian Computations of Two-Phase Flows in General Coordinates, *International Journal of Multiphase Flow*, Vol. 25, 1999, pp. 813-825.

[8] Chordá R., Blasco J.A., and Fueyo N., An Efficient Particle-Locating Algorithm for Application in Arbitrary 2d and 3d Grids, *International Journal of Multiphase Flow*, Vol. 28, 2002, pp. 1565-1580.

[9] Apte S.V., Mahesh K., Moin P., and Oefelein J.C., Large-Eddy Simulation of Swirling Particle-Laden Flows in a Coaxial-Jet Combustor, *International Journal of Multiphase Flow*, Vol. 29, 2003, pp. 1311-1331.

[10] Vaidya A.M., Subbarao P.M.V., and Gaur R.R., A Novel and Efficient Method for Particle Locating and Advancing over Deforming, Nonorthogonal Mesh, *Numerical Heat Transfer, Part B: Fundamentals: An International Journal of Computation and Methodology*, Vol. 49, 2006, pp. 67 - 88.

[11] Haselbacher A., Najjar F.M., and Ferry J.P., An Efficient and Robust Particle-Localization Algorithm for Unstructured Grids, *Journal of Computational Physics*, Vol. 225, 2007, pp. 2198-2213.

[12] Löhner R., and Ambrosiano J., A Vectorized Particle Tracer for Unstructured Grids, *Journal of Computational Physics*, Vol. 91, 1990, pp. 22-31.

[13] Dompierre J., Labbé P., Vallet M.G., and Camarero R., How to Subdivide Pyramids, Prisms and Hexahedra into Tetrahedra, *Paper presented at the Proceedings of the 8th International Meshing Roundtable*, Sandia National Laboratories, Albuquerque 1999.

[14] Chen X.Q., Efficient Particle Tracking Algorithm for Two-Phase Flows in Geometries Using Curvilinear Coordinates, *Numerical Heat Transfer, Part A: Applications: An International Journal of Computation and Methodology*, Vol. 32, 1997, pp. 387 - 405.

[15] Clift R., Grace J.R., and Weber M.E. *Bubbles, Drops and Particles*: Academic Press, 1978.

[16] Saffman P.G., The Lift on a Small Sphere in a Slow Shear Flow, *Journal of Fluid Mechanics*, Vol. 22, 1965, pp. 385-398.

[17] McLaughlin J.B., Intertial Migration of a Small Sphere in Linear Shear Flows, *Journal of Fluid Mechanics*, Vol. 224, 1991, pp. 261-274.

[18] Barton I.E., Exponential-Lagrangian Tracking Schemes Applied to Stokes Law, *Journal of Fluids Engineering*, Vol. 118, 1996, pp. 85-89.

[19] Ruetsch G.R., and Meilburg E., On the Motion of Small Spherical Bubbles in Two-Dimensional Vortical Flows, *Physics of Fluids A*, Vol. 5, 1993, pp. 2326-2341.

[20] Barton I.E., Computation of Particle Tracks over a Backward-Facing Step, *Journal of Aerosol Science*, Vol. 26, 1995, pp. 887-901.

[21] Armaly B.F., Durst F., Pereira J.C.F., and Schdnung B., Experimental and Theoretical Investigation of Backward-Facing Step Flow., *journal of Fluid Mechanics*, Vol. 127, 1983, pp. 473-496.