# Some Advances in Development and Applications of Immersed Boundary Method (IBM)

C. Shu

Department of Mechanical Engineering
National University of Singapore

# Outline

- Introduction
- Conventional Immersed Boundary Method (IBM)
- Boundary Condition-enforced IBM
- Some Recent Progresses
- Conclusions

# 1. Introduction

Current numerical methods can be roughly classified as

- **Body-fitted solver**
  - Conventional methods; strong coupling, tedious mesh generation
- **Sharp interface non-body-fitted solver**
  - Cartesian mesh; Cut-cell, Ghost fluid, Immersed interface method; loose coupling; tedious treatment at boundary
- **Diffuse interface non-body-fitted solver**
  - Cartesian mesh; Immersed boundary method, decoupling; equal treatment for simple and complex geometry

# 2. Conventional Immersed Boundary Method (IBM)

- Proposed by C. S. Peskin in 1972 to study fluid dynamics of heart valves (2D model)
- Effect of boundary to surrounding fluids is through the body force

# Idea of IBM:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \upsilon\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + f_x$$

**Body force density at Eulerian points**

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \upsilon\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + f_y$$

$$\boldsymbol{f}(\boldsymbol{x},t) = \oint \boldsymbol{F}(s,t)\,\delta\left(\boldsymbol{x} - \boldsymbol{X}(s,t)\right)ds$$

**Boundary force density at Lagrangian points**

# Calculation of Force:

## Hooke's law (Peskin 1972, JCP):

$$\boldsymbol{F}\left(X,t\right)=-k\Delta\xi=-k\left(\mathbf{V}_{fluid}\,\Delta t-\mathbf{V}_{wall}\,\Delta t\right)$$

$$\boldsymbol{f}\,(x,t)=\sum_{j}\boldsymbol{F}(X_{j},t)D_{j}(x-X_{j})\Delta s_{j}$$

**Idea is simple;**
**Needs a user-specified parameter**

## Direct forcing method (Fadlun et al 2000, JCP):

$$\boldsymbol{f}(\boldsymbol{x}=\boldsymbol{X}(s,t),t)=\rho\left(\frac{\partial \boldsymbol{u}}{\partial t}+\boldsymbol{u}\cdot\nabla\boldsymbol{u}\right)+\nabla p-\mu\Delta\boldsymbol{u}$$

$$\boldsymbol{f}(\boldsymbol{x},t)=\sum_{j=1}^{m-1}\boldsymbol{f}(\boldsymbol{X}(s_j,t),t)D_j(\boldsymbol{x}-\boldsymbol{X}(s_j,t))\Delta sh$$

**Needs to compute derivatives and interpolation**

**Improved Version (Predictor-Corrector Scheme):**

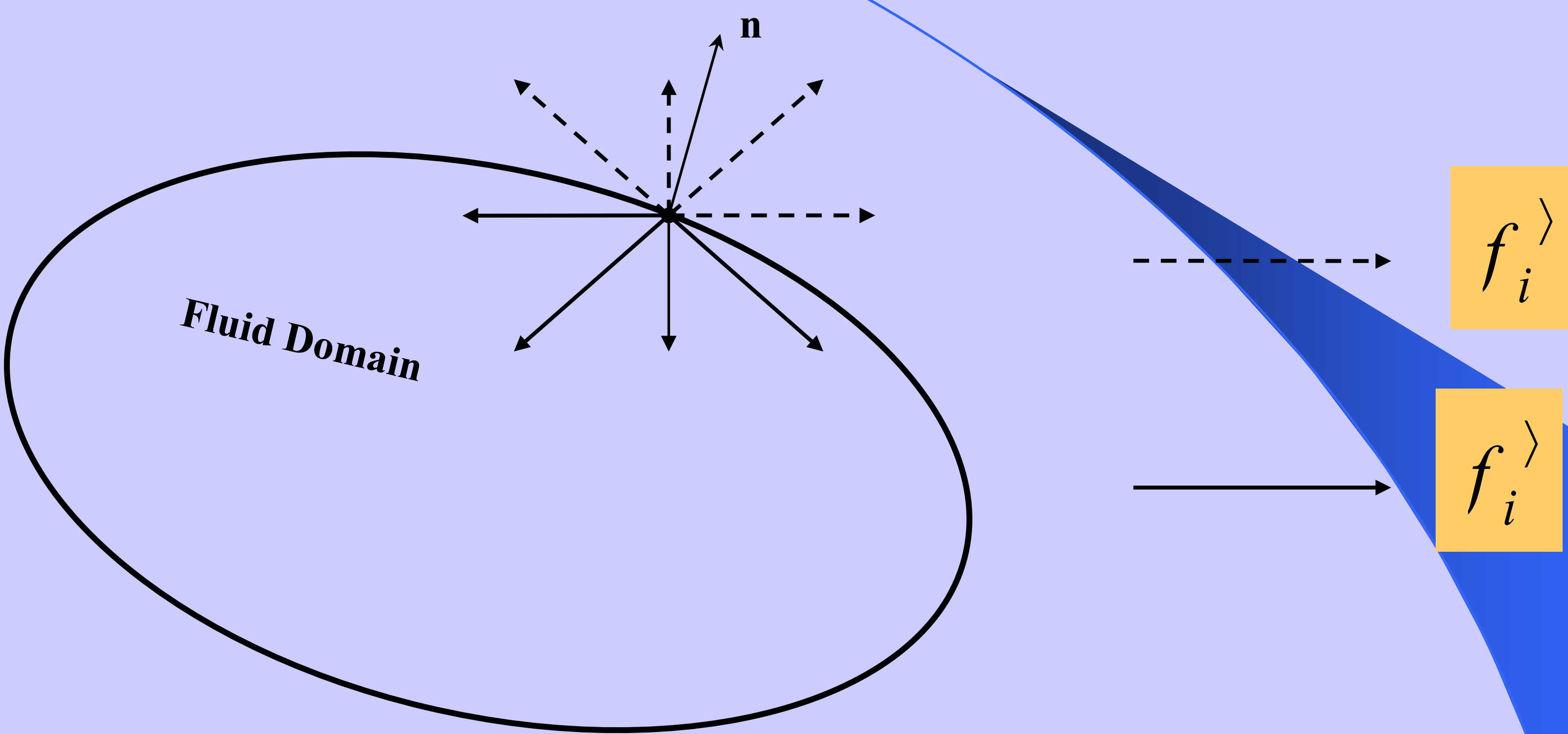
**Predictor: Get u* from**

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t}+\boldsymbol{u}\cdot\nabla\boldsymbol{u}\right)+\nabla p-\mu\Delta\boldsymbol{u}$$

**Corrector: compute f from**

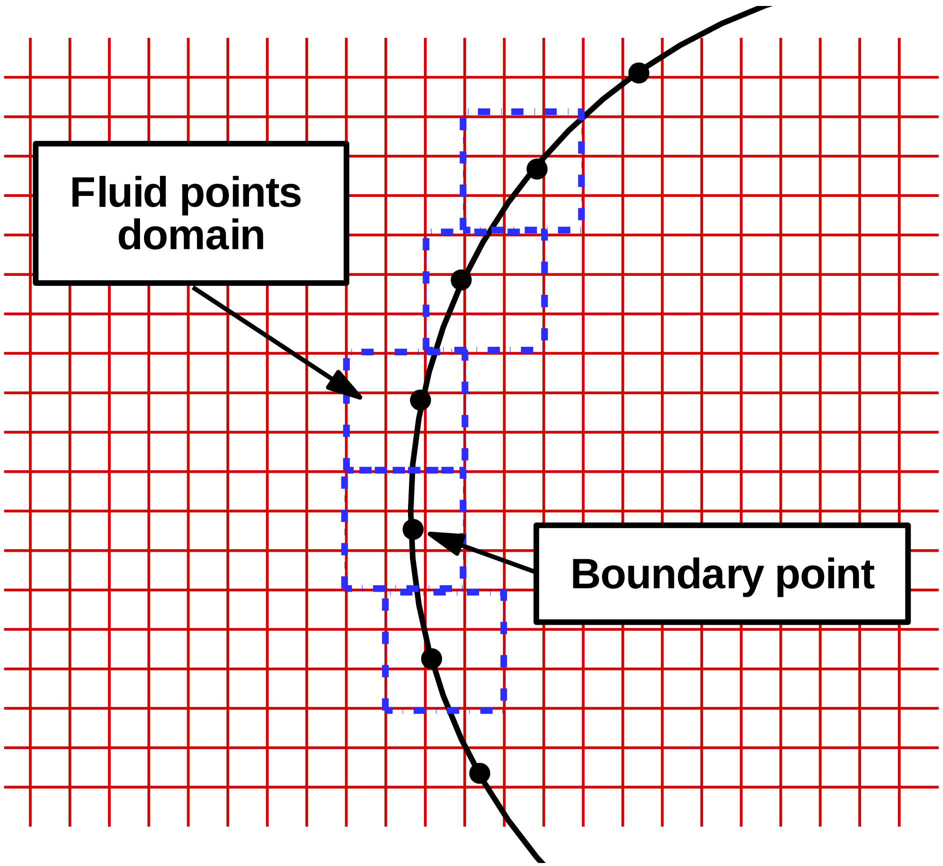
$$\boldsymbol{f}(\boldsymbol{x}=\boldsymbol{X}(s,t),t)=\rho\frac{\boldsymbol{u}_{wall}-\boldsymbol{u}^*}{\Delta t}$$

# Momentum exchange method
## (Niu, Shu and Chew, 2006, Physical Letters A)




**Momentum exchange equals to impulse**

# Insight of IBM:




**Physical View**

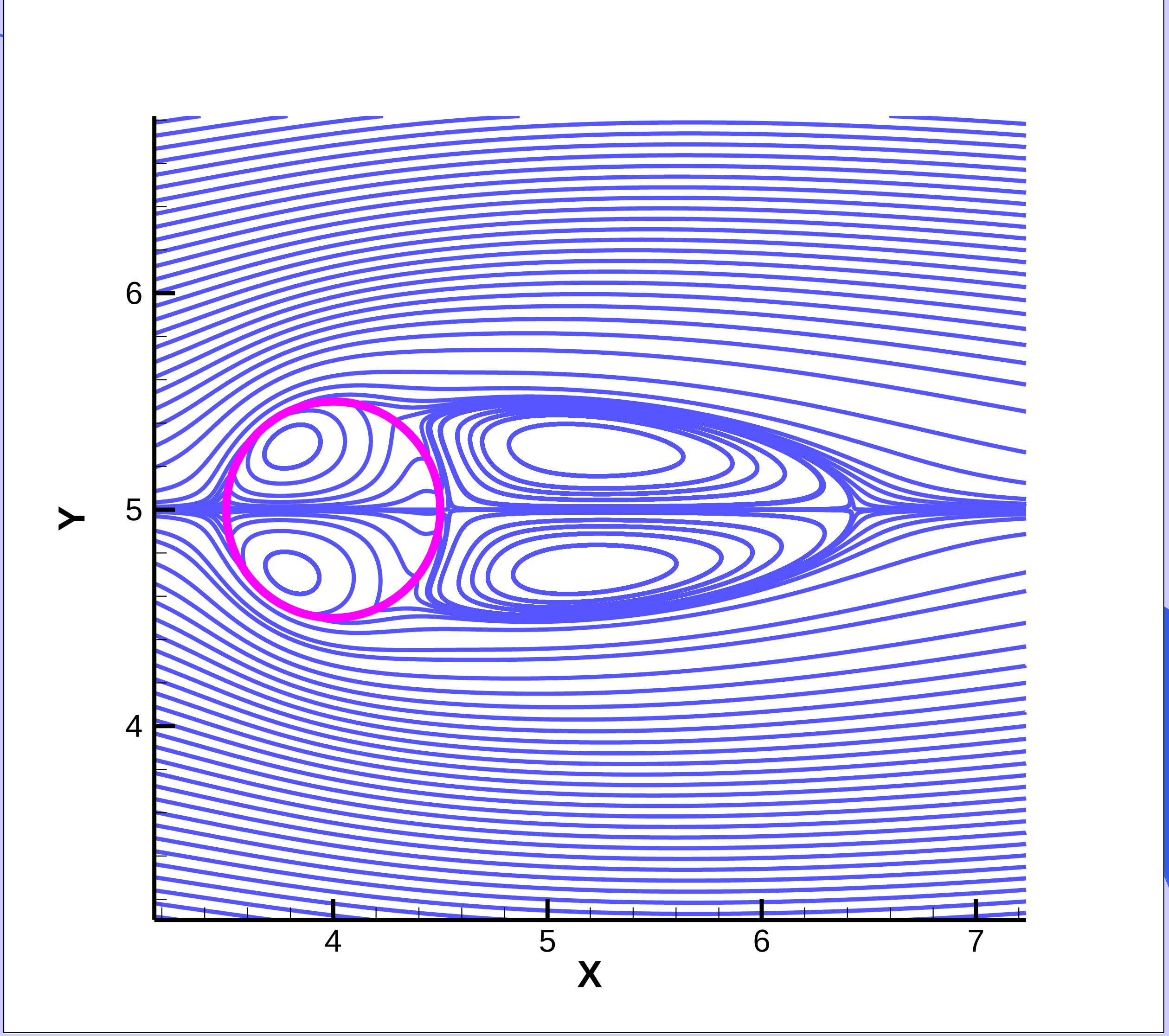
(**Momentum flux at a surface would affect velocity in a control cell**)

# Computational Sequence:

- Eulerian points for flow field while Lagrangian points for boundary;
- Solve N-S equations without body force *f*;
- Compute the force at boundary (Lagrangian points);
- Distribute the boundary force to surrounding fluid points;
- Re-solve N-S equations with body force;
- Carry on computation until convergence is reached.

# Features of IBM:

- Very simple implementation;
- Iterative procedure to satisfy both governing equations and boundary conditions;
- Boundary conditions are approximately satisfied;
- Streamlines may penetrate the solid boundary;
- Calculation of forces is not accurate;
- Use of $\delta$-function just has 1st order of accuracy near boundary.




**Simulation of Flow past a Circular Cylinder by IBM, Re=40**

# 3. Boundary Condition-enforced IBM

**Shu, Liu & Chew (JCP, 2007)**

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \upsilon\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + f_x$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \upsilon\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + f_y$$

**Introduction of body force is equivalent to make velocity correction**

## Fractional step approach:

**Predictor step:** Solve the normal Navier-Stokes equations (setting f=0) and obtain predicted velocity components u*, v*

**Corrector step:** Correct the velocity field

$$\frac{\partial u}{\partial t} = f_x \qquad \frac{\partial v}{\partial t} = f_y$$

$$u^{n+1} = u^* + \Delta t \cdot f_x = u^* + u'$$ ($f_x$ only affects u)

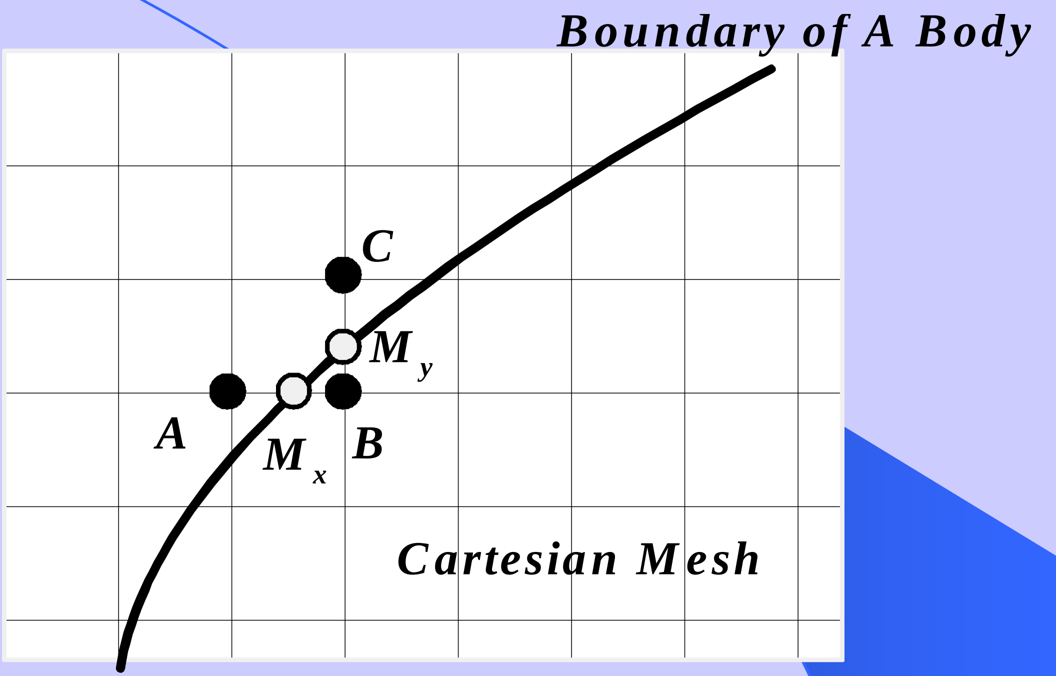
$$v^{n+1} = v^* + \Delta t \cdot f_y = v^* + v'$$ ($f_y$ only affects v)

## Conventional IBM:

**f calculated ➡ u', v' fixed ➡ $u^{n+1}$, $v^{n+1}$ not satisfy B. C.**

## Interpolation:

$$u_{ix} = u_A + \frac{\Delta_1}{\Delta x}(u_B - u_A)$$



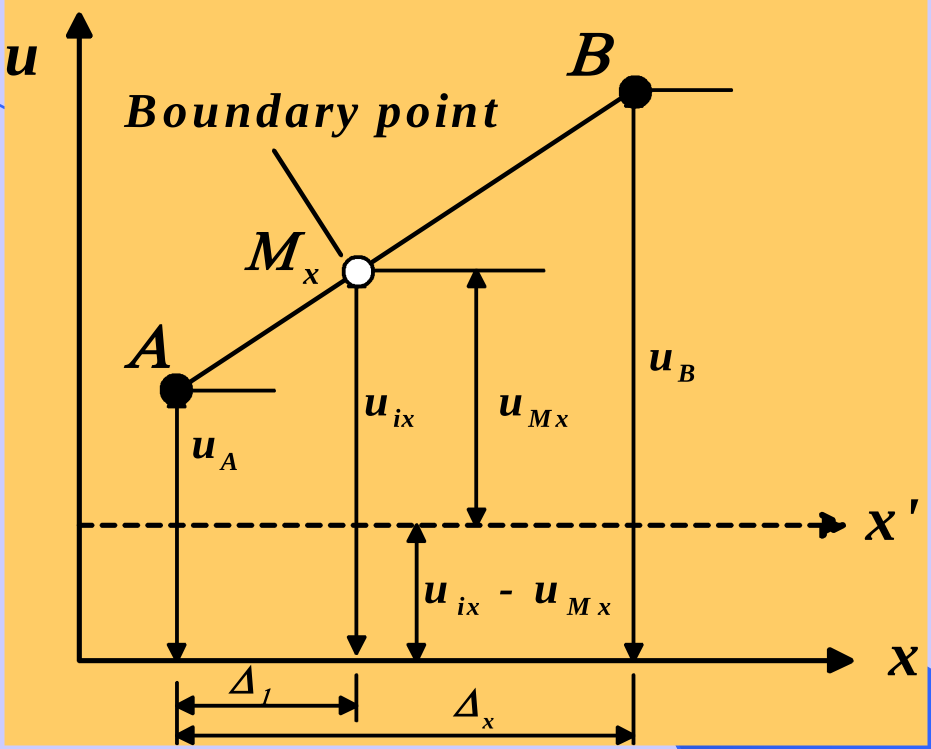

**$u_A$ and $u_B$ are fluid velocity**

**Cartesian mesh lines, immersed boundary and their intersection points**

**Requirement:**
**$u_{ix}$ + u' satisfies B. C.**

## Velocity correction:

$$u' = u_{Mx} - u_A - \frac{\Delta_1}{\Delta x}(u_B - u_A)$$




**$u_{Mx}$ – velocity at boundary**

**Linear velocity distribution between two mesh points**

**u' is applied at A and B**

**Flow field can be obtained by Navier-Stokes solver or Lattice Boltzmann solver**

# Features of New Scheme:

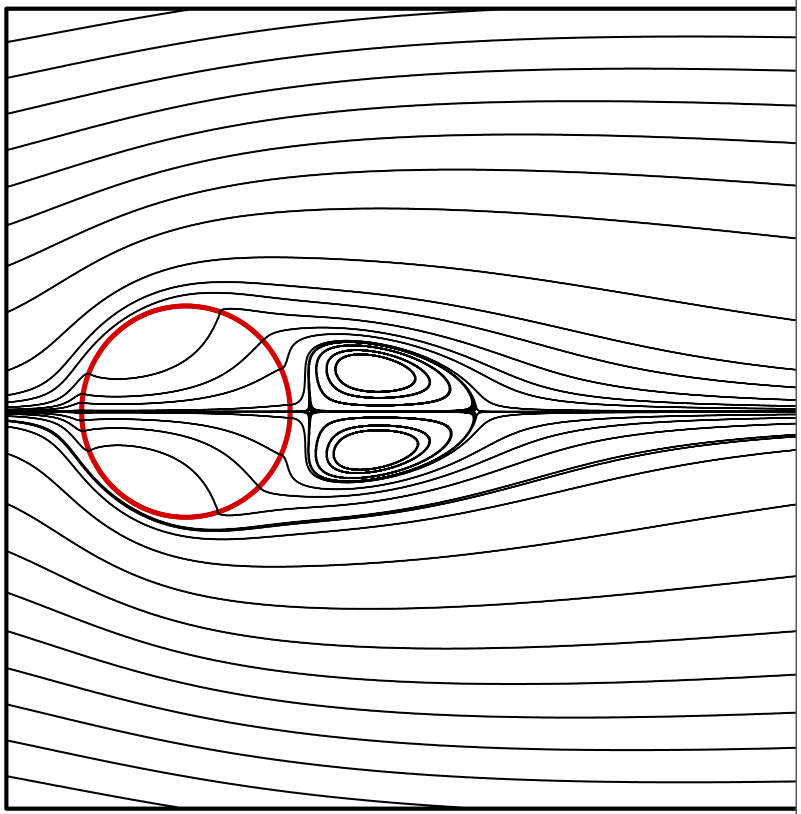
Non-slip boundary condition can be accurately satisfied

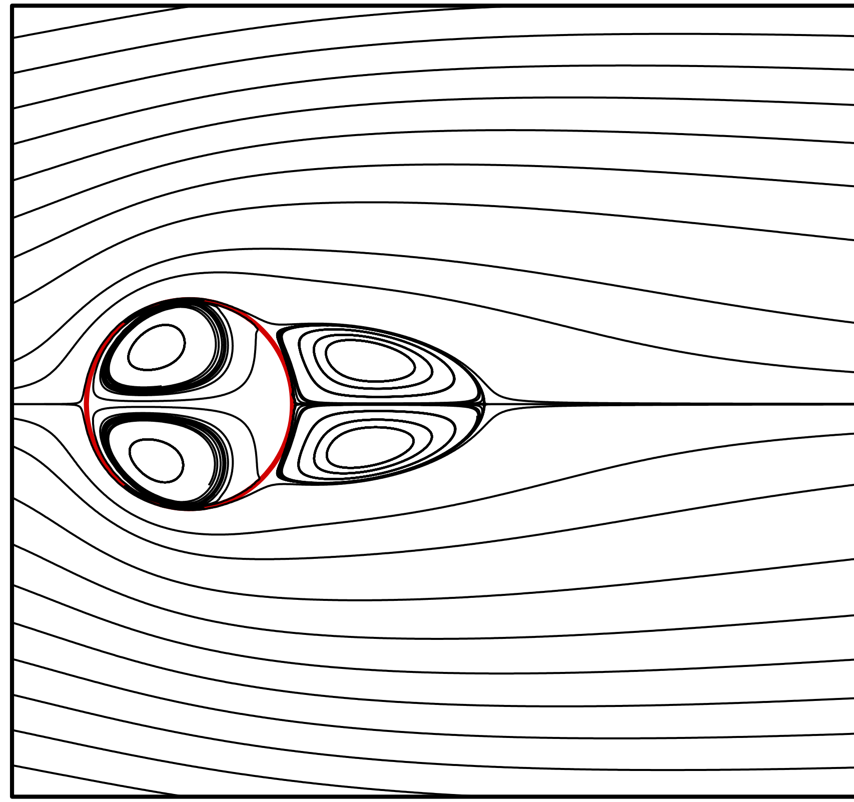
No $\delta$-function is involved

Need to get intersection points between boundary and mesh lines

Force calculation is from velocity correction

# Flow around a cylinder

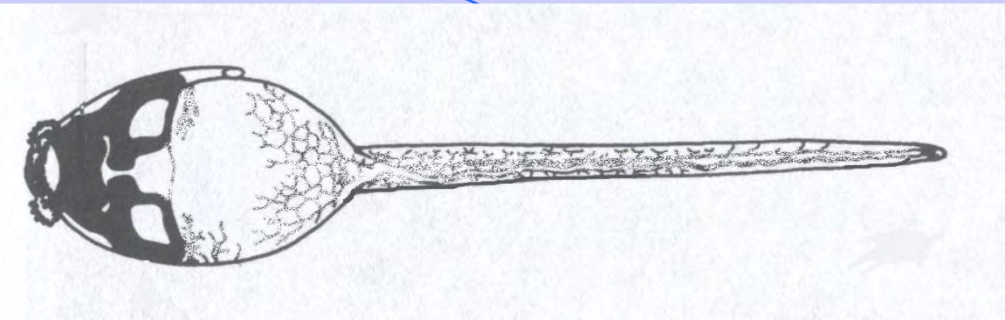




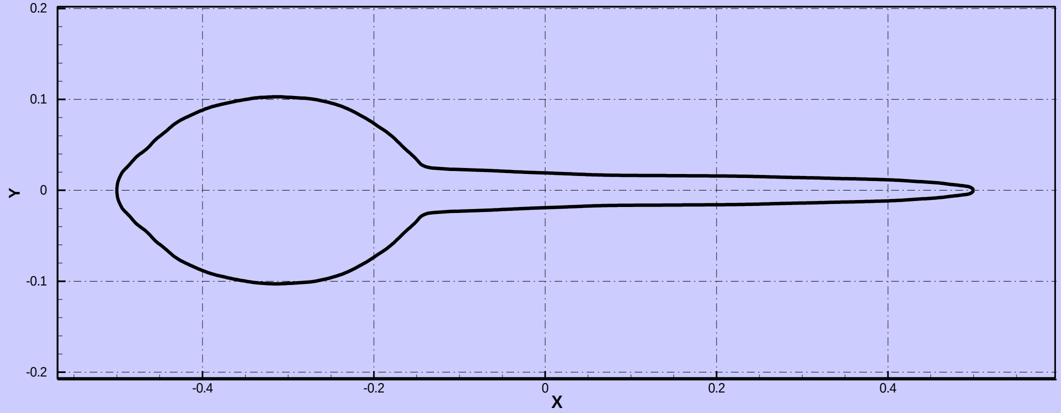
Conventional IBM

Re = 20

# Simulation of Fish Motion



**Physical Model**



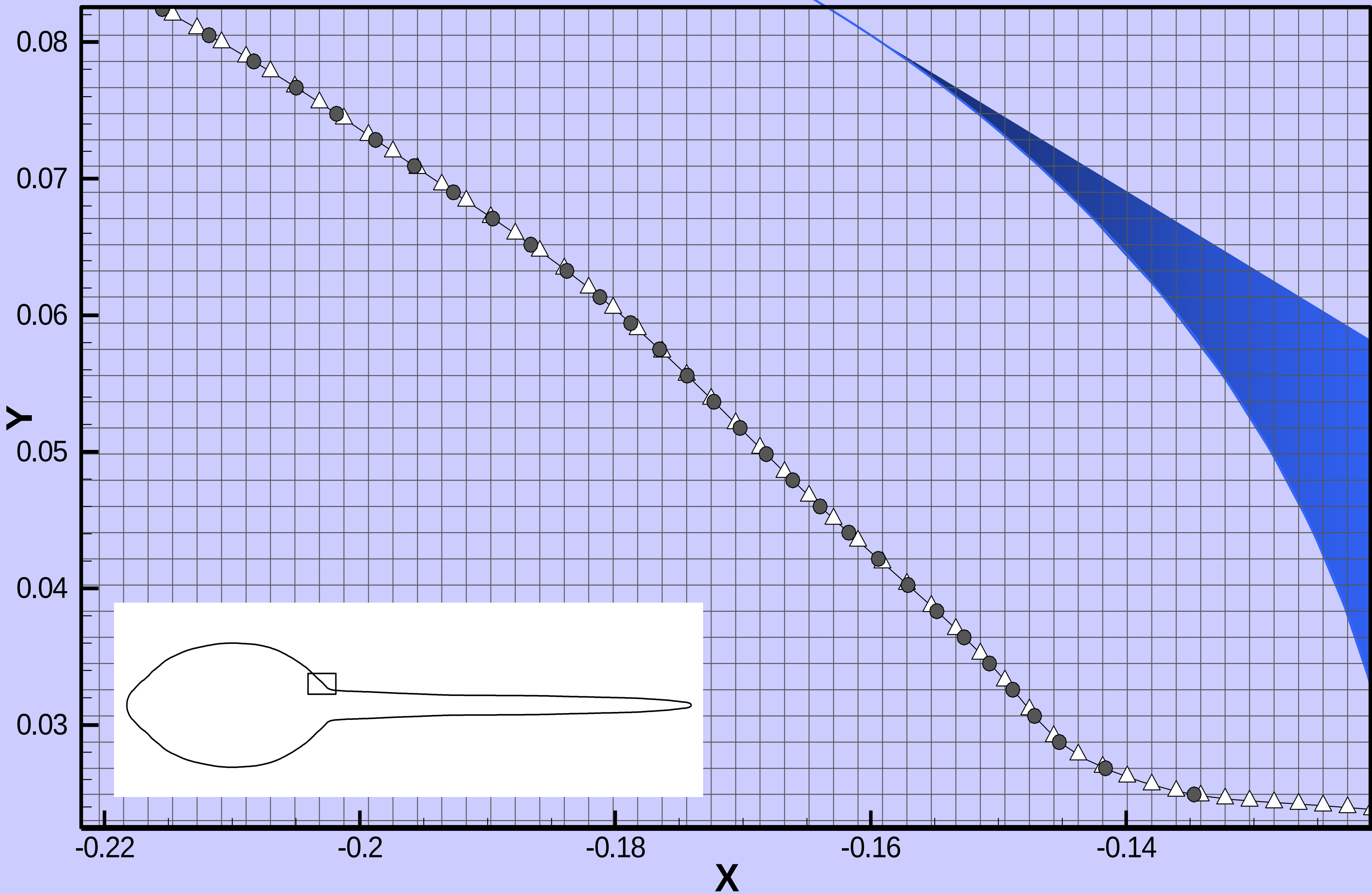

**Mathematical Model**

$$y_m(x,t) = a_m(x)\cos(2\pi x/\lambda - 2\pi ft)$$

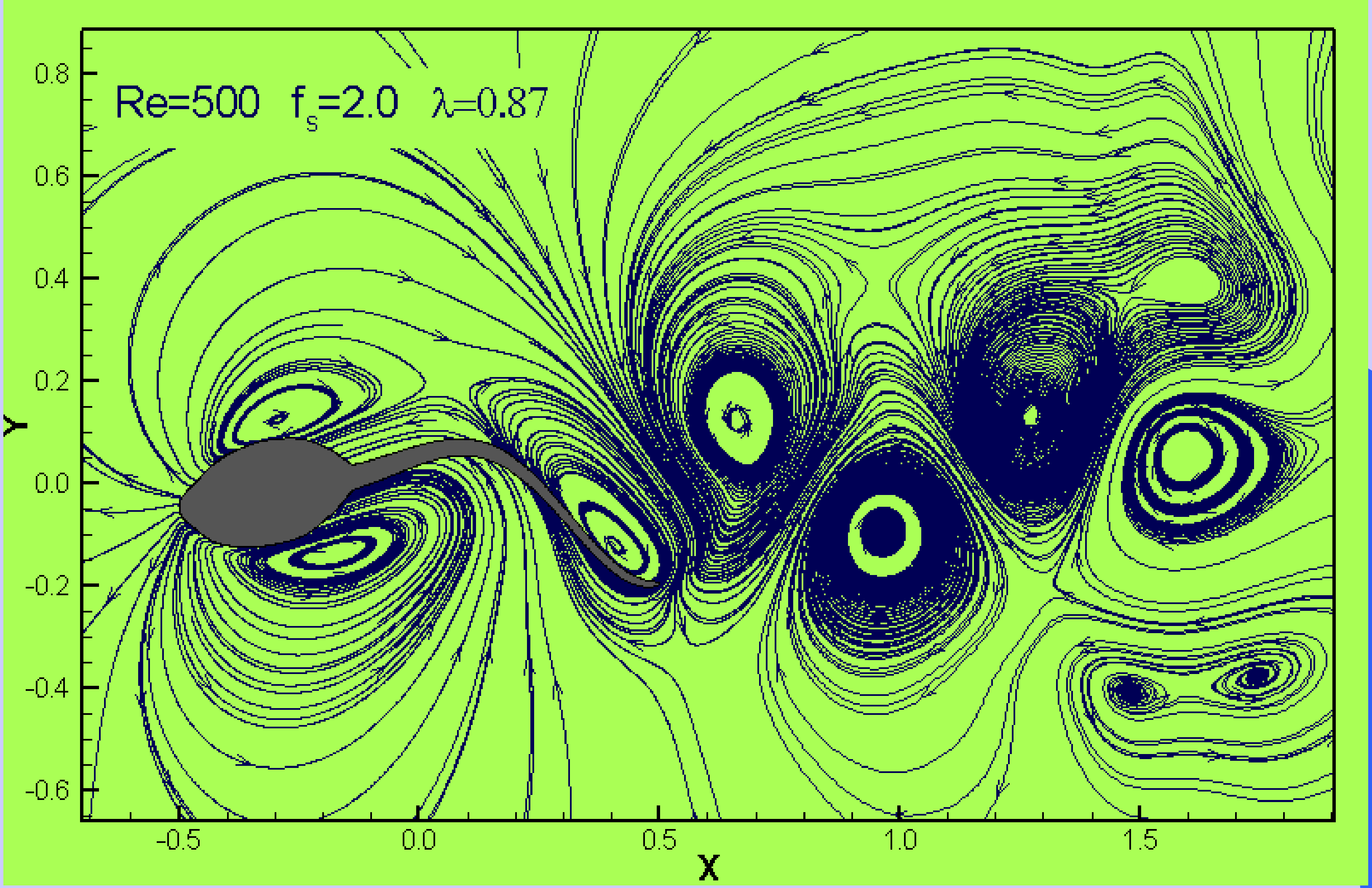
**f: frequency** **λ: wavelength**

**Amplitude can be approximated by a polynomial with curve fitting**

# The mesh line-boundary crossing points

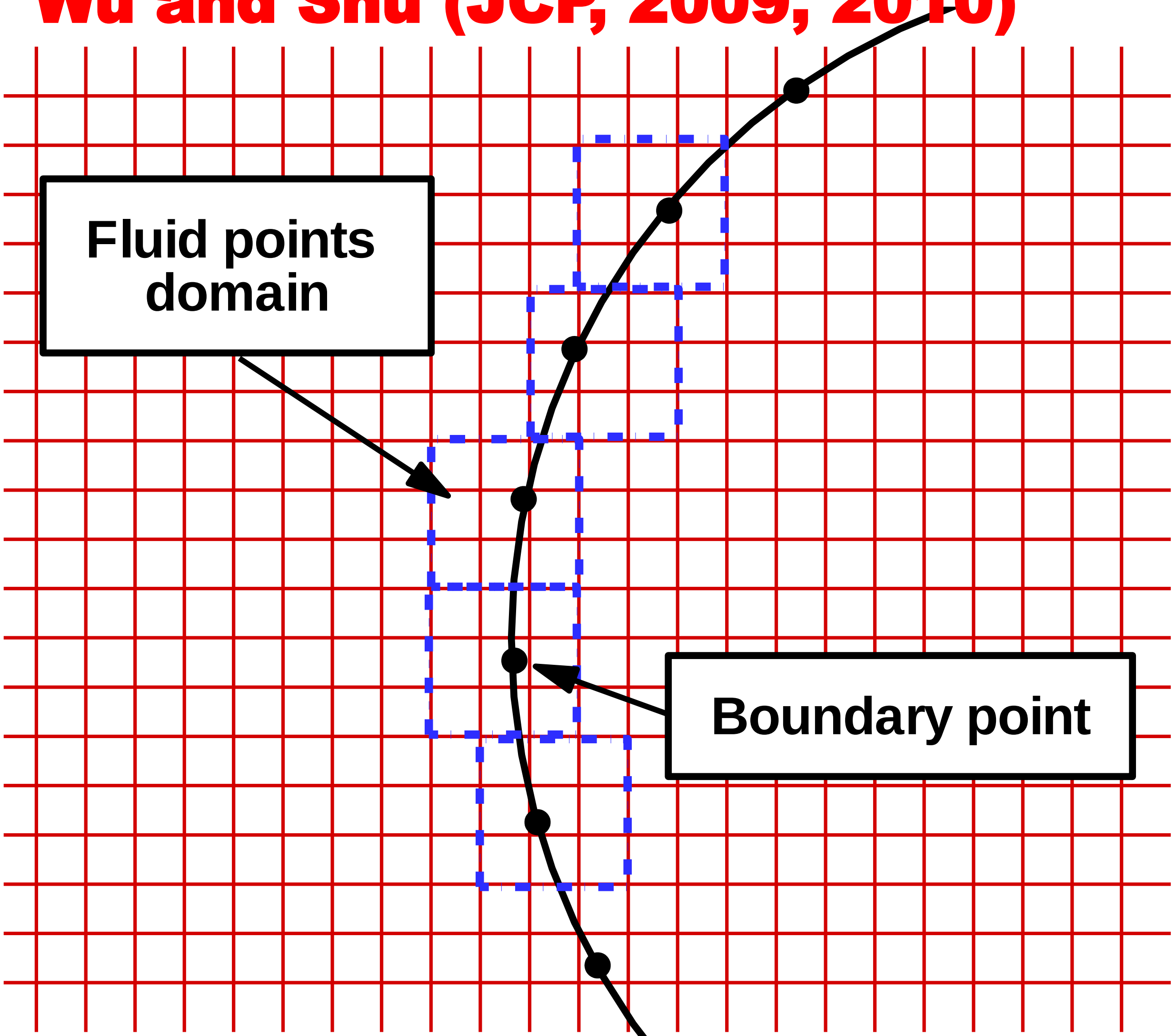


# The streamlines for ($U-U_0$, V) for the case with thrust



# Extension of Idea of Boundary Condition Enforcement to Conventional IBM

**Wu and Shu (JCP, 2009, 2010)**




**Information exchange is through δ-function interpolation**

# As an example, use lattice Boltzmann method as a solver for flow field

**Guo, Zheng & Shi (PRE, 2002)**

$$f_\alpha\left(\mathbf{x}+\mathbf{e}_\alpha\delta t, t+\delta t\right) - f_\alpha\left(\mathbf{x}, t\right) = -\frac{1}{\tau}\left(f_\alpha\left(\mathbf{x}, t\right) - f_\alpha^{eq}\left(\mathbf{x}, t\right)\right) + F_\alpha \delta t$$

$$F_\alpha = \left(1-\frac{1}{2\tau}\right) w_\alpha \left(\frac{\mathbf{e}_\alpha - \mathbf{u}}{c_s^2} + \frac{\mathbf{e}_\alpha \cdot \mathbf{u}}{c_s^4}\mathbf{e}_\alpha\right) \cdot \mathbf{f}$$

$$\rho \mathbf{u} = \sum_\alpha \mathbf{e}_\alpha f_\alpha + \frac{1}{2}\mathbf{f}\delta t$$

**Velocity correction**

**Can properly consider the discrete lattice effects to the density and momentum**

$$\mathbf{U}_B^l\left(\mathbf{X}_B^l,t\right)=\sum_{i,j}\mathbf{u}\left(\mathbf{x}_{ij},t\right)D_{ij}\left(\mathbf{x}_{ij}-\mathbf{X}_B^l\right)\Delta x\Delta y$$

$$\mathbf{u}\left(\mathbf{x}_{ij},t\right)=\mathbf{u}^*\left(\mathbf{x}_{ij},t\right)+\delta\mathbf{u}\left(\mathbf{x}_{ij},t\right)$$

unknown

$$\delta\mathbf{u}\left(\mathbf{x},t\right)=\int_\Gamma\delta\mathbf{u}_B\left(\mathbf{X}_B,t\right)\delta\left(\mathbf{x}-\mathbf{X}_B\left(s,t\right)\right)ds$$

$$\delta\mathbf{u}\left(\mathbf{x}_{ij},t\right)=\sum_{l}\delta\mathbf{u}_B^l\left(\mathbf{X}_B^l,t\right)D_{ij}\left(\mathbf{x}_{ij}-\mathbf{X}_B^l\right)\Delta s_l$$

$\delta$**u is related to the force density**

$$\mathbf{U}_B^l\left(\mathbf{X}_B^l, t\right) = \sum_{i,j} \mathbf{u}^*\left(\mathbf{x}_{ij}, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right) \Delta x \Delta y$$

$$+ \sum_{i,j} \sum_{l} \delta \mathbf{u}_B^l\left(\mathbf{X}_B^l, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right) \Delta s_l D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right) \Delta x \Delta y$$

$$\mathbf{A}\mathbf{X} = \mathbf{B}$$

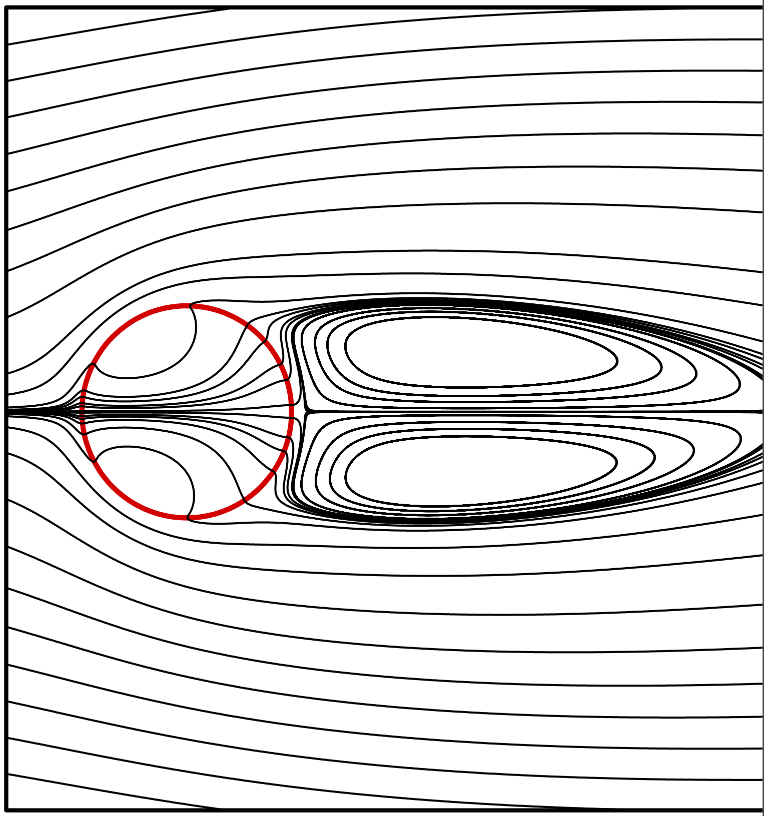
$$\mathbf{X} = \left\{\delta \mathbf{u}_B^1, \delta \mathbf{u}_B^2, \cdots, \delta \mathbf{u}_B^m\right\}$$

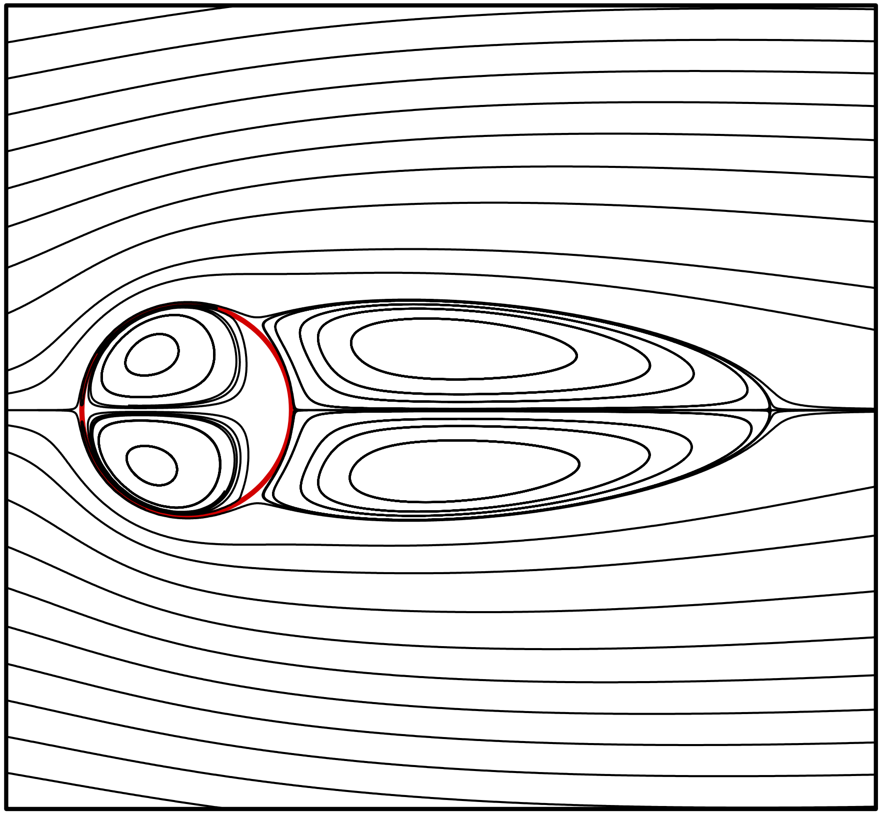
$$\mathbf{B} = \left\{\Delta \mathbf{u}_1, \Delta \mathbf{u}_2, \cdots, \Delta \mathbf{u}_m\right\}^T$$

$$\Delta \mathbf{u}_l = \mathbf{U}_B^l\left(\mathbf{X}_B^l, t\right) - \sum_{i,j} \mathbf{u}^*\left(\mathbf{x}_{ij}, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right) \Delta x \Delta y$$

# Features of Improved IBM:

- Easy Application (same as conventional IBM)
- Accurate satisfaction of non-slip boundary condition
- Easy and accurate calculation of force (directly from velocity correction on the boundary)



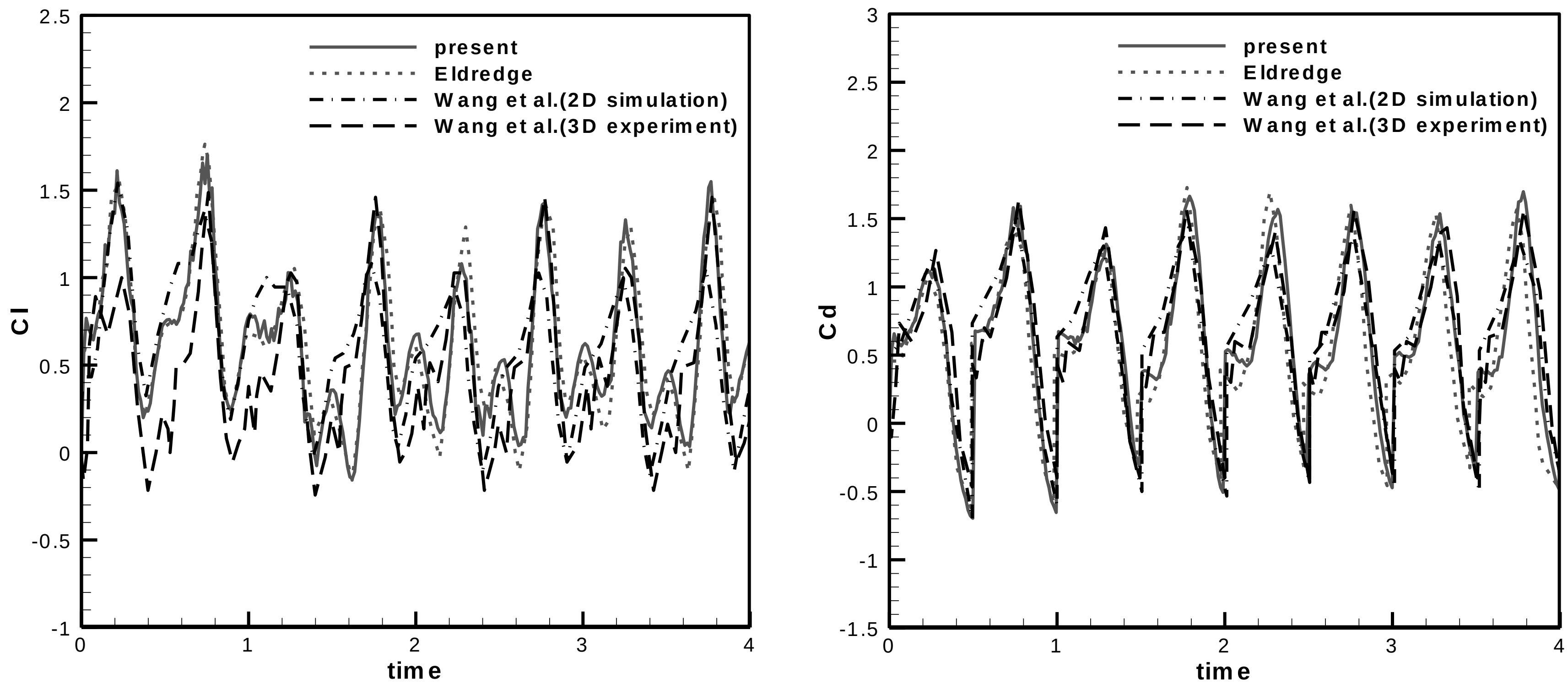


**Conventional IBM**

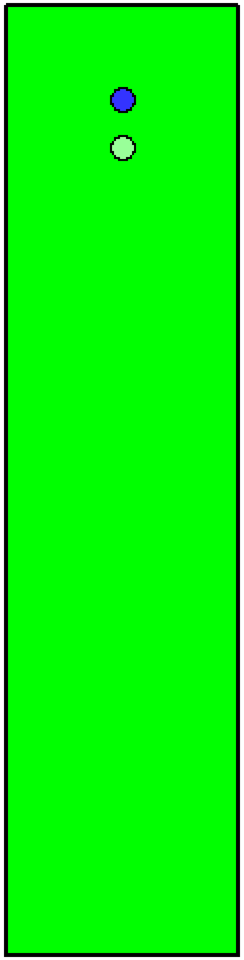
**Boundary condition-enforced IBM**

**Re = 40**

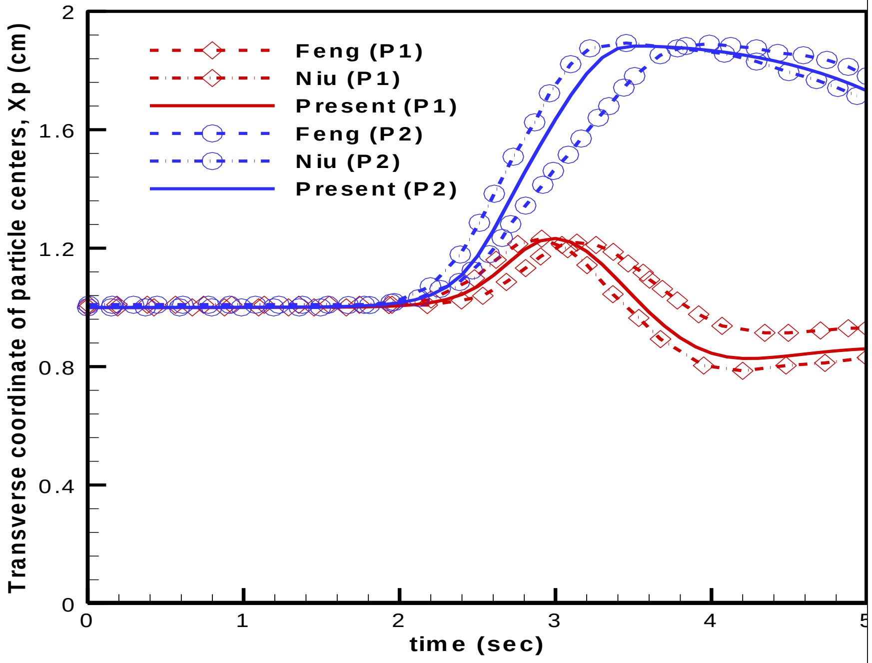
**Unsteady Flows around a moving body**

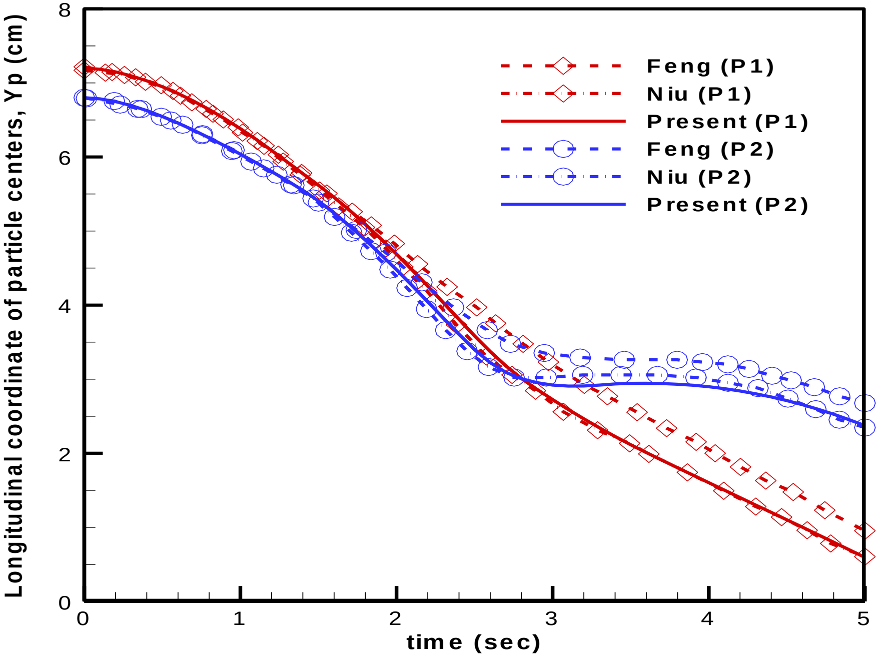



**Time histories of lift and drag coefficients for $\phi = \pi/4$**



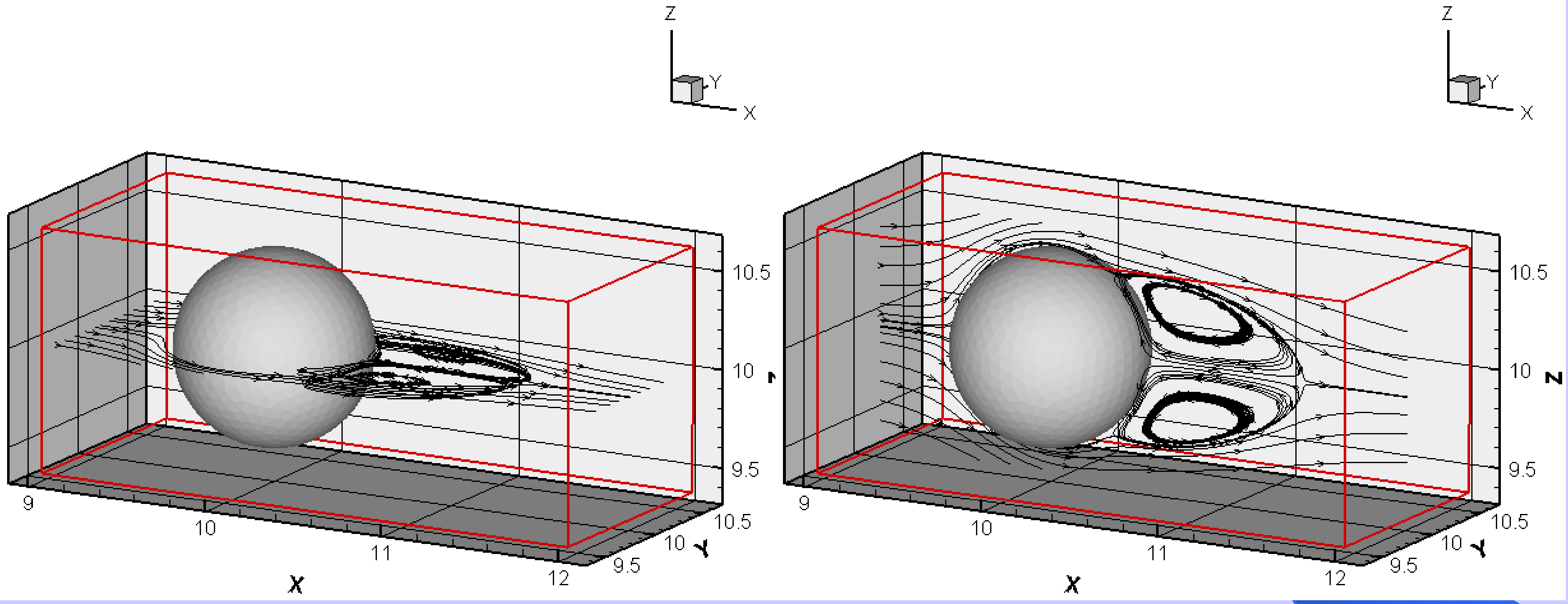
**Sedimentation of two particles in a channel at different time stages**




**Transverse coordinates of the centers of two particles**

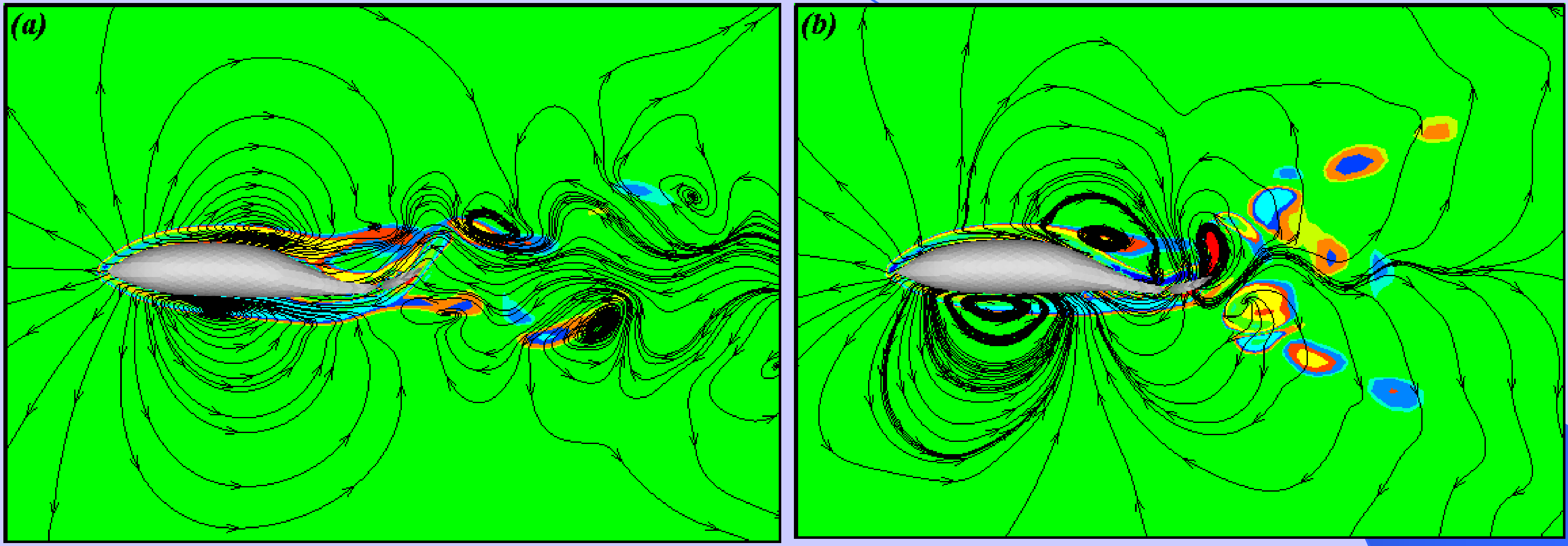
**Longitudinal coordinates of the centers of two particles**




**Steady axisymmetric flow past a sphere**
***Re*** **= 100**

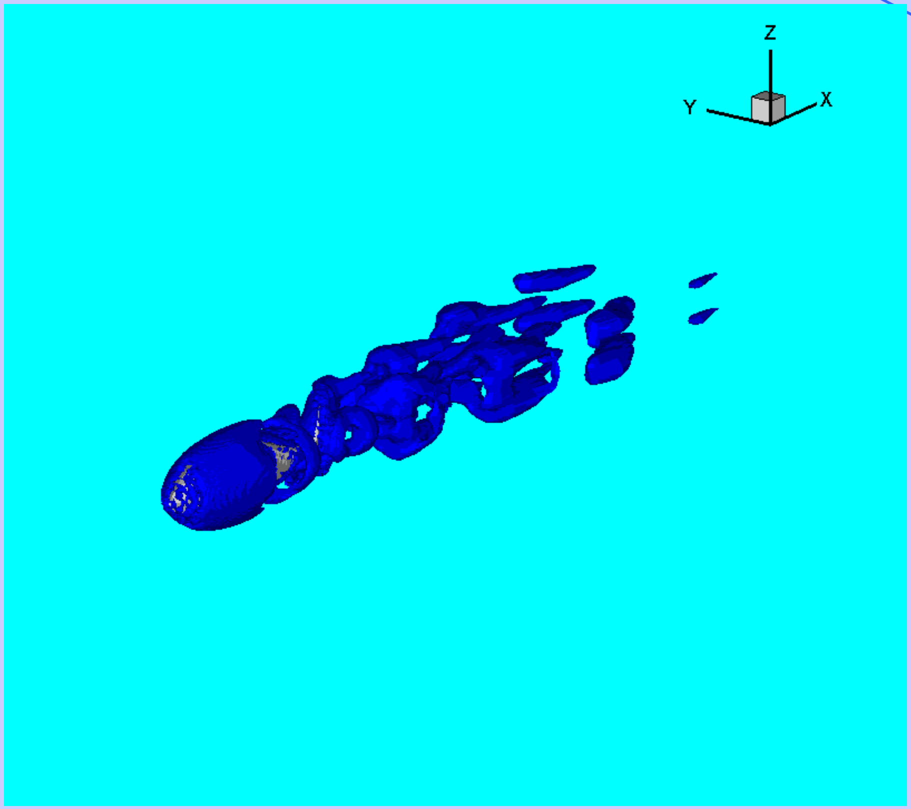
**Cd = 1.13 (present), 1.10 (Johnson et al.)**

# Simulation of 3D Fish Motion

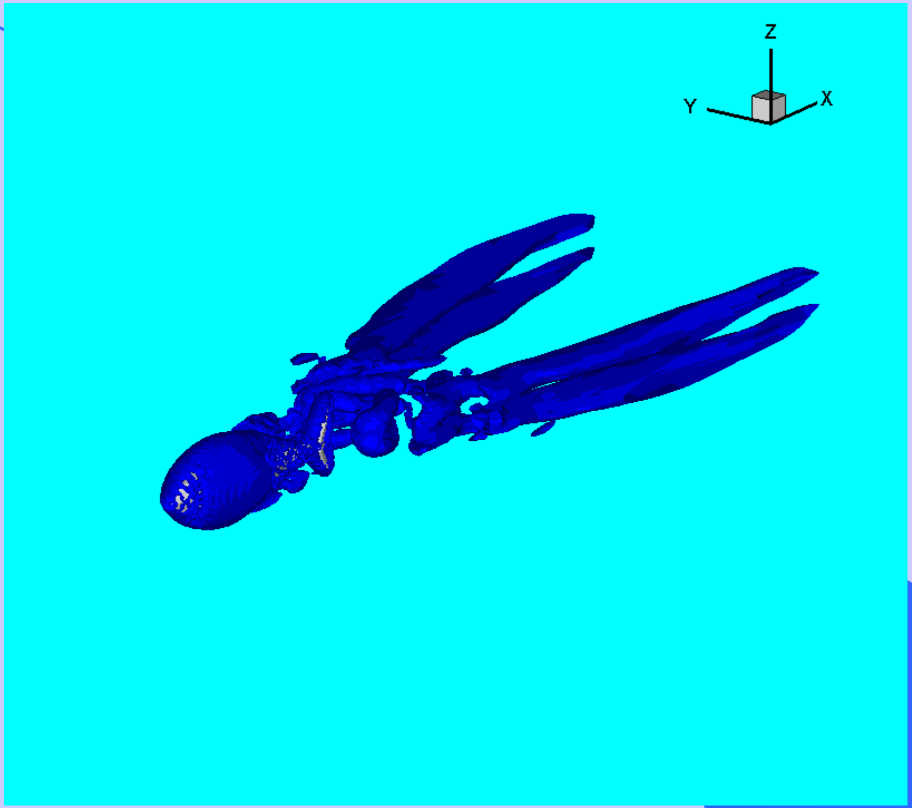



**St=0.3 drag type**
**Karman vortex street**

**St=0.7 thrust type**
**Reverse Karman vortex street**

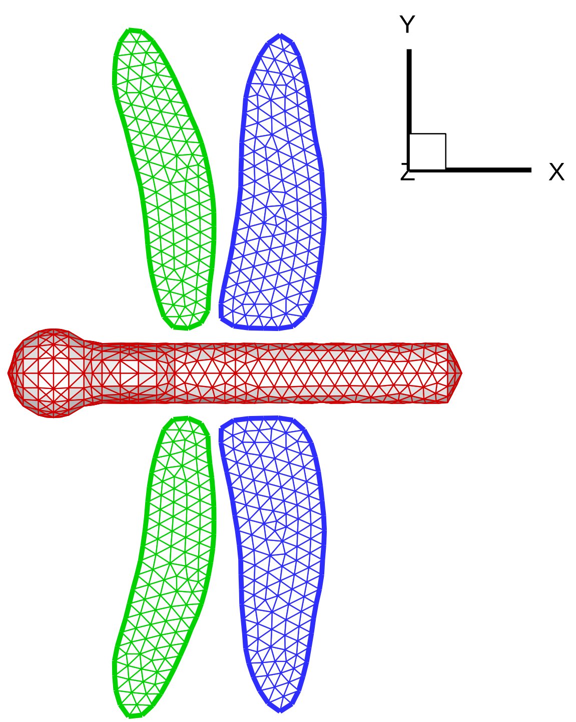



**St=0.3**
**Single row wake**

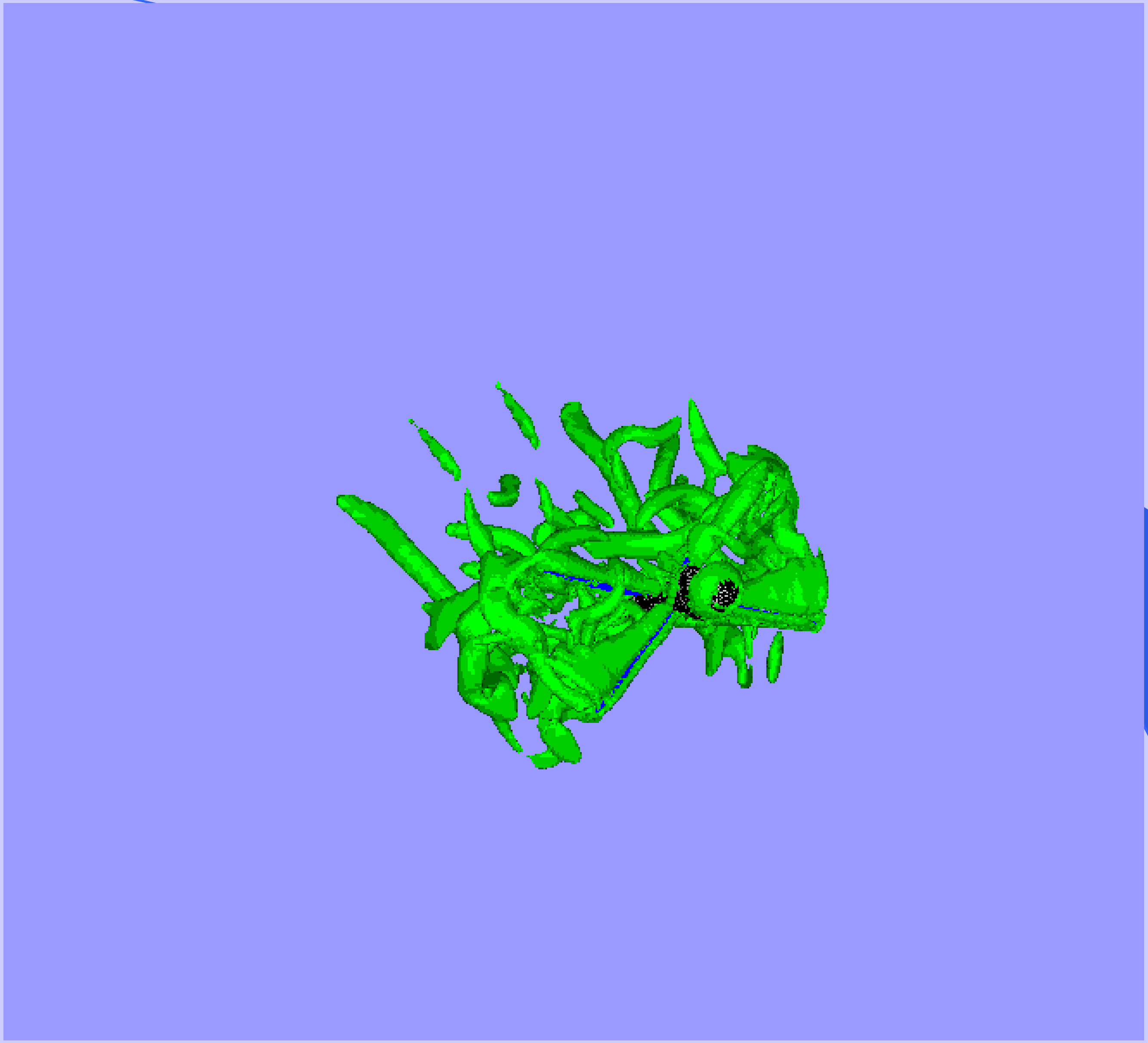



**St=0.7**
**Double row wake**

Physical model



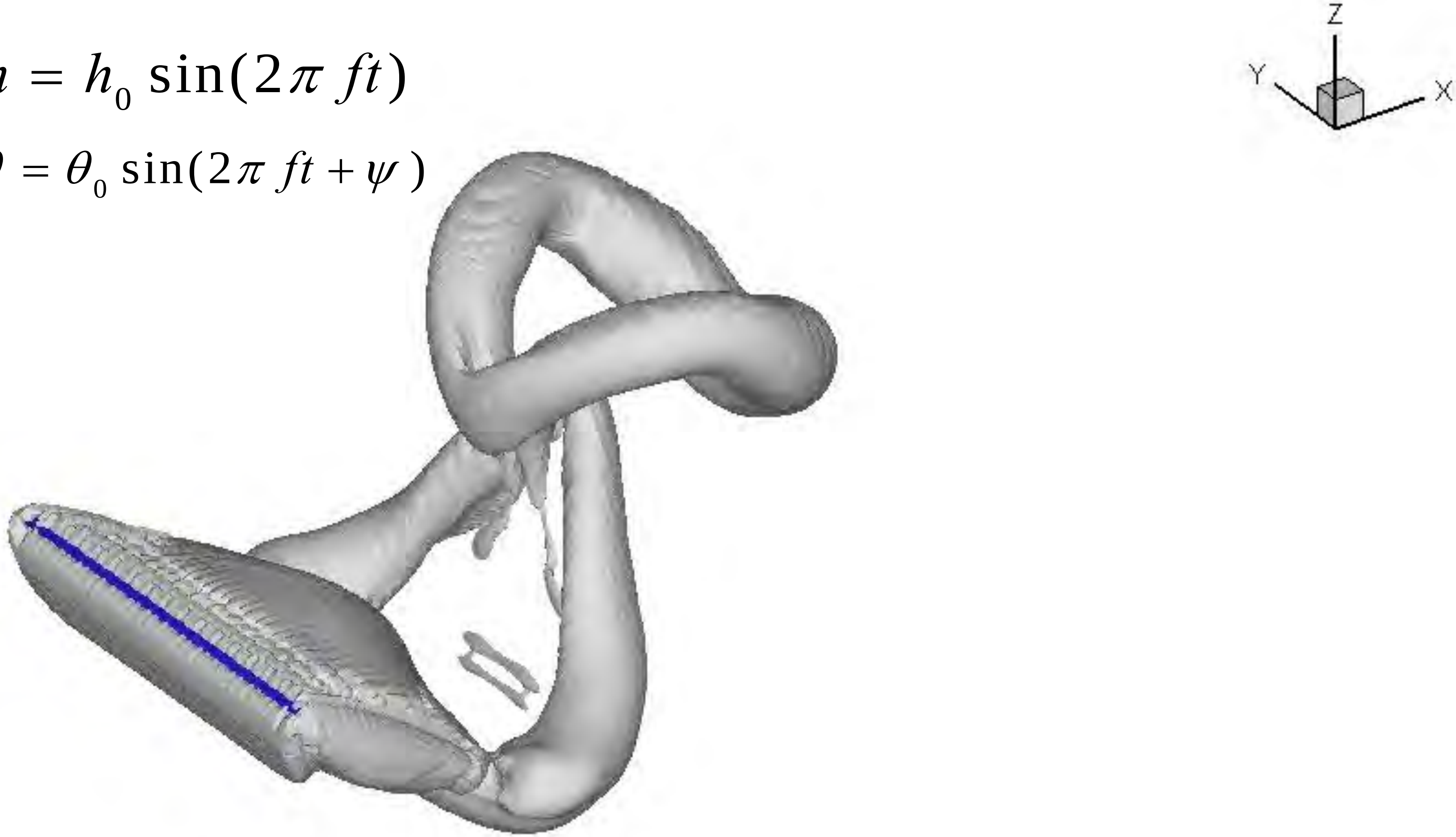

Mathematical model

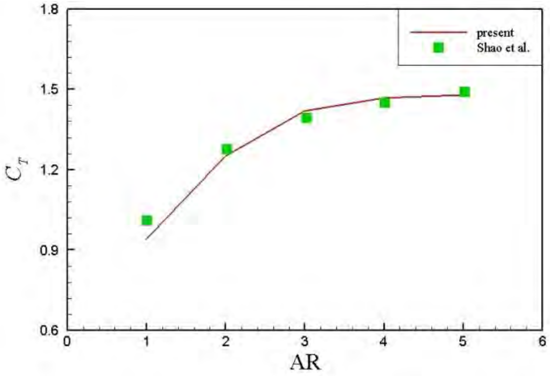


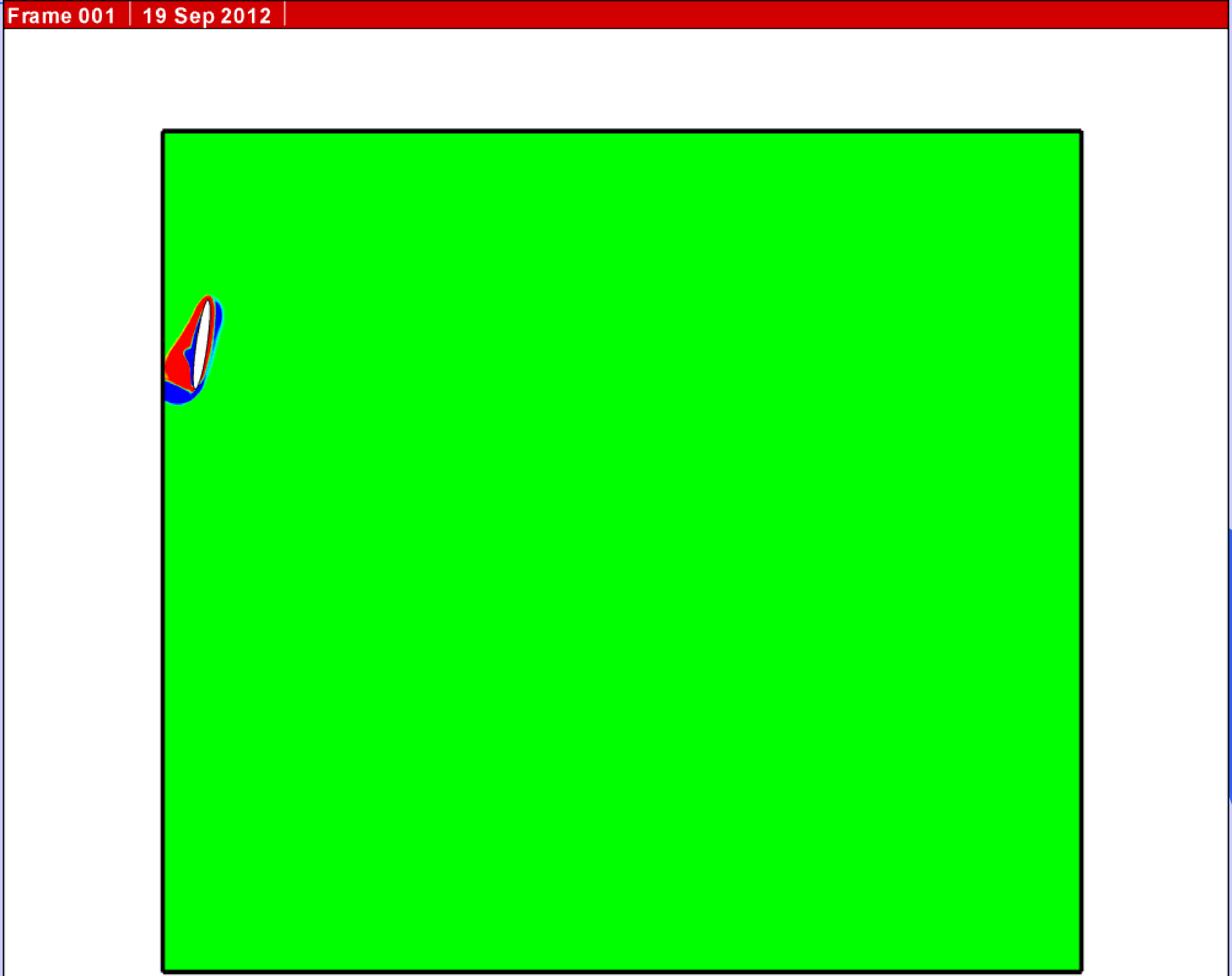
# 3D dragonfly

$$h = h_0 \sin(2\pi f t)$$

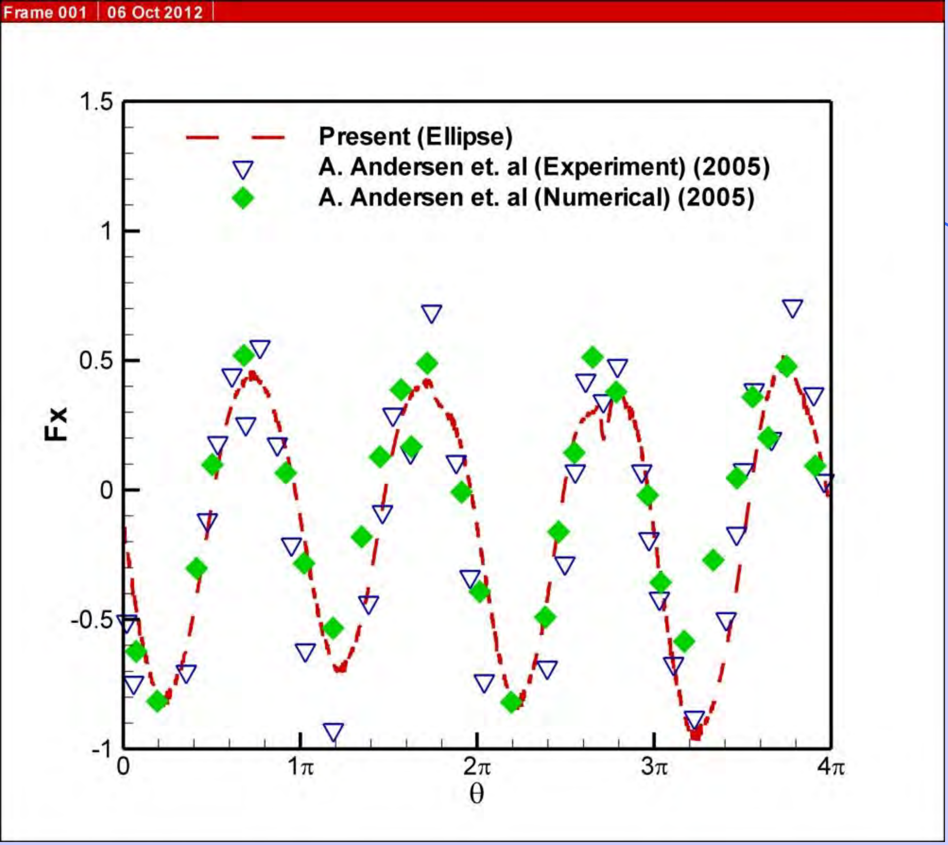
$$\theta = \theta_0 \sin(2\pi f t + \psi)$$

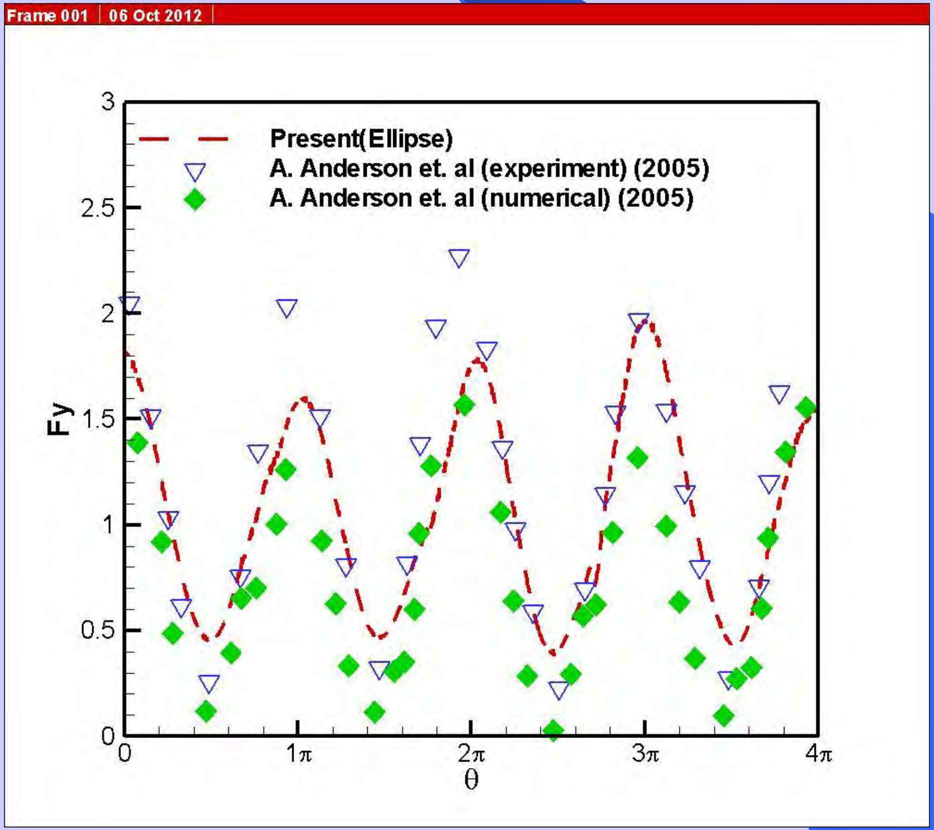


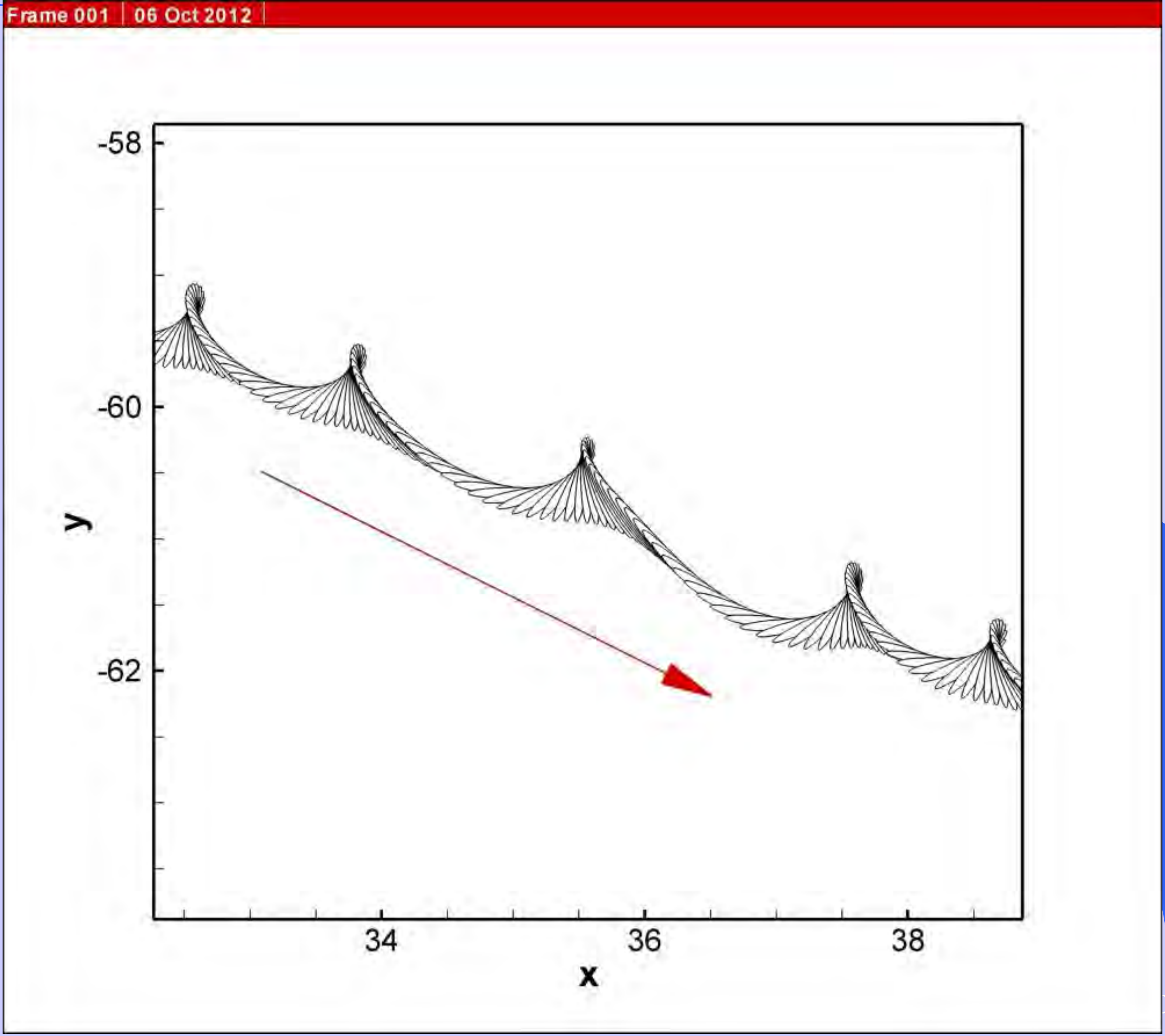






**Free falling of an ellipse**


Frame 001 | 06 Oct 2012
Present (Ellipse)
A. Andersen et. al (Experiment) (2005)
A. Andersen et. al (Numerical) (2005)
Fx
θ
0
1π
2π
3π
4π
1.5
1
0.5
0
-0.5
-1


Frame 001 | 06 Oct 2012
Present(Ellipse)
A. Anderson et. al (experiment) (2005)
A. Anderson et. al (numerical) (2005)
Fy
θ
0
1π
2π
3π
4π
3
2.5
2
1.5
1
0.5
0



Frame 001 | 06 Oct 2012
-58
-60
-62
y
34
36
38
x


| | Fernandes et al (2007, Experiment) | Shenoy et al (2010, Numerical) | Present |
|---|---|---|---|
| $\theta_{y\max}$ | 22.72° | 27.51° | 21.48° |
| Phase difference | 195.3° | - | 191.8° |

# 4. Some Recent Progresses

## 4.1 Extension to Thermal Flow Problems

Ren et al (C&F, 2002)

- Velocity field is treated using the same way as shown early on
- Energy equation is solved by adding a heat source term

$$\rho c_p\left(\frac{\partial T}{\partial t} + (\mathbf{u} \bullet \nabla)T\right) = k\nabla^2 T + q$$

Heat source/sink

$$q(\mathbf{x},t) = \int_\Gamma Q(\mathbf{X}(s),t)\delta(\mathbf{x} - \mathbf{X}(s,t))ds$$

# Conventional way to evaluate Heat Source term

- Directly from conventional IBM

  **E.g. from direct forcing scheme**

$$\boxed{q = \rho c_p\left(\frac{\partial T}{\partial t} + (\mathbf{u} \bullet \nabla)T\right) - k\nabla^2 T}$$

Applied at boundary points

Wall temperature

$$q^{n+1} = \rho c_p\left(\frac{T_B^{n+1} - T^n}{\Delta t} + (\mathbf{u}^{n+1} \bullet \nabla)T^n\right) - k\nabla^2 T^n$$

- Tedious
- q is pre-calculated
- Satisfying of boundary condition is not guaranteed

# Boundary condition-enforced IBM for energy equation

- Predictor step

$$\rho c_p\left(\frac{\partial T^*}{\partial t} + (\mathbf{u} \bullet \nabla) T^*\right) = k\nabla^2 T^*$$

$\longrightarrow$ **Get T***

- Corrector step

$$\rho c_p \frac{\partial T}{\partial t} = q$$

$$\rho c_p \frac{\delta T}{\delta t} = q \quad \text{with} \quad \delta T = T^{n+1} - T^*$$

$$\delta T = \frac{\delta t}{\rho c_p} q = \frac{\delta t}{\rho c_p} \int_{\Gamma} Q(\mathbf{X}(s), t)\delta(\mathbf{x} - \mathbf{X}(s,t))ds$$

$$\delta T\left(\mathbf{x}_{ij}, t\right) = \frac{\delta t}{\rho c_p} \sum_l Q_B\left(\mathbf{X}_B^l, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right) \Delta s_l$$

$$T_B\left(\mathbf{X}_B^l, t\right) = \sum_{i,j} T\left(\mathbf{x}_{ij}, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right)\Delta x \Delta y$$

$$T\left(\mathbf{x}_{ij}, t\right) = T^*\left(\mathbf{x}_{ij}, t\right) + \delta T\left(\mathbf{x}_{ij}, t\right)$$

unknown

$$\delta T\left(\mathbf{x}_{ij}, t\right) = \frac{\delta t}{\rho c_p} \sum_{l} Q_B\left(\mathbf{X}_B^l, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right)\Delta s_l$$

$$T_B\left(\mathbf{X}_B^l, t\right) = \sum_{i,j} T^*\left(\mathbf{x}_{ij}, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right)\Delta x \Delta y$$

$$+\frac{\delta t}{\rho c_p}\sum_{i,j}\sum_{l} Q_B\left(\mathbf{X}_B^l, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right)\Delta s_l D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right)\Delta x \Delta y$$

$$\mathbf{A}\mathbf{X} = \mathbf{B}$$

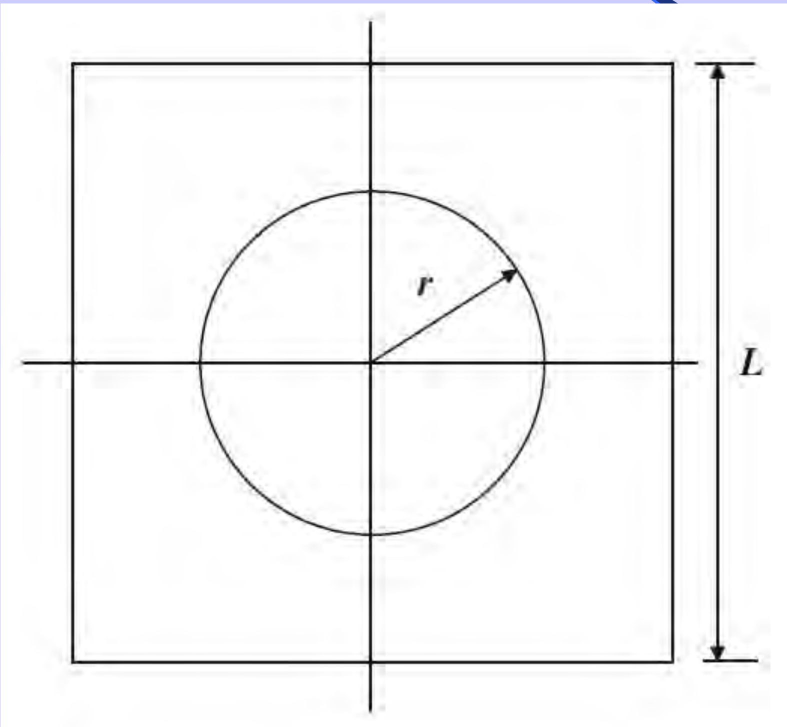
$$\mathbf{X} = \left\{Q_B^1, Q_B^2, \cdots, Q_B^m\right\}$$

$$\mathbf{B} = \left\{\Delta T_1, \Delta T_2, \cdots, \Delta T_m\right\}^T$$

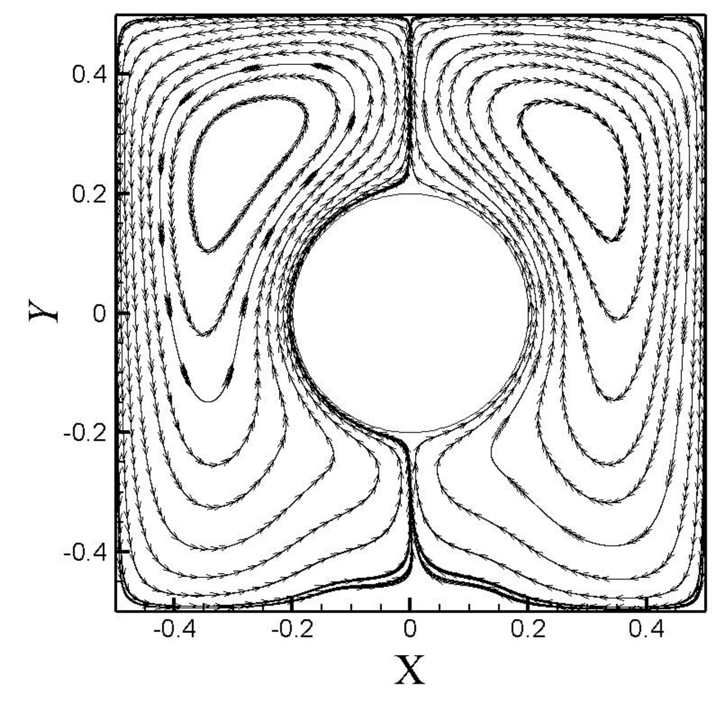
$$\Delta T_l = T_B^l\left(\mathbf{X}_B^l, t\right) - \sum_{i,j} T^*\left(\mathbf{x}_{ij}, t\right) D_{ij}\left(\mathbf{x}_{ij} - \mathbf{X}_B^l\right)\Delta x \Delta y$$

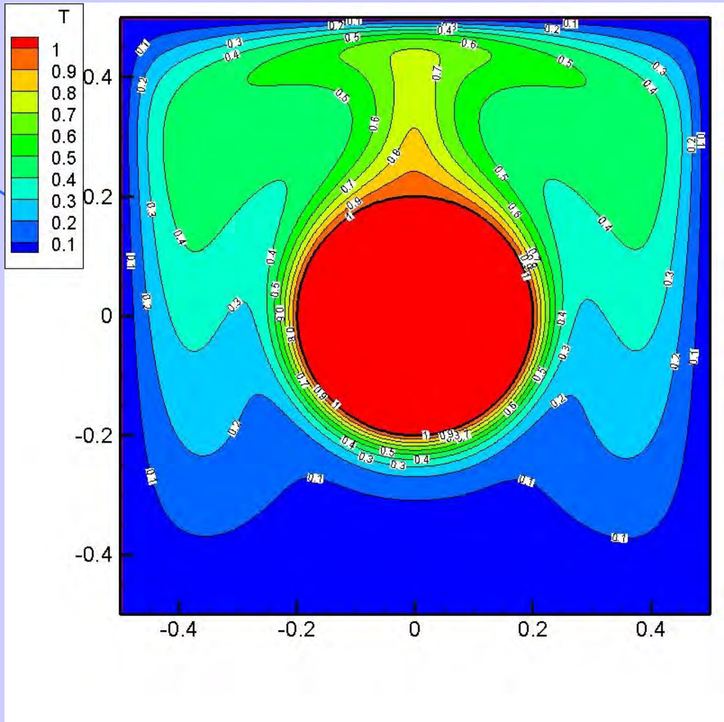
**Solution of system gives heat flux at boundary points**

# Natural convection in a concentric *isothermal* annulus between a square outer cylinder and a circular inner cylinder




$$L/(2r) = 2.5$$




**Streamlines** **Isotherms**

$$Ra = 1 \times 10^{6}$$

# 4.2 Extension of IB-LBM for Neumann boundary condition

**Shu et al (IJNMF, 2013)**

$\dfrac{\partial T}{\partial n}$ is given **or** $Q_B = -k\dfrac{\partial T}{\partial n}$ is given

- **Very little work on this part**
- **Possible reason: Delta function may not be appropriate for derivative approximation at boundary points**
- **Use energy equation to illustrate method**

# Recall early work

$$\rho c_p \left(\frac{\partial T}{\partial t} + (\mathbf{u} \bullet \nabla) T\right) = k \nabla^2 T + q$$

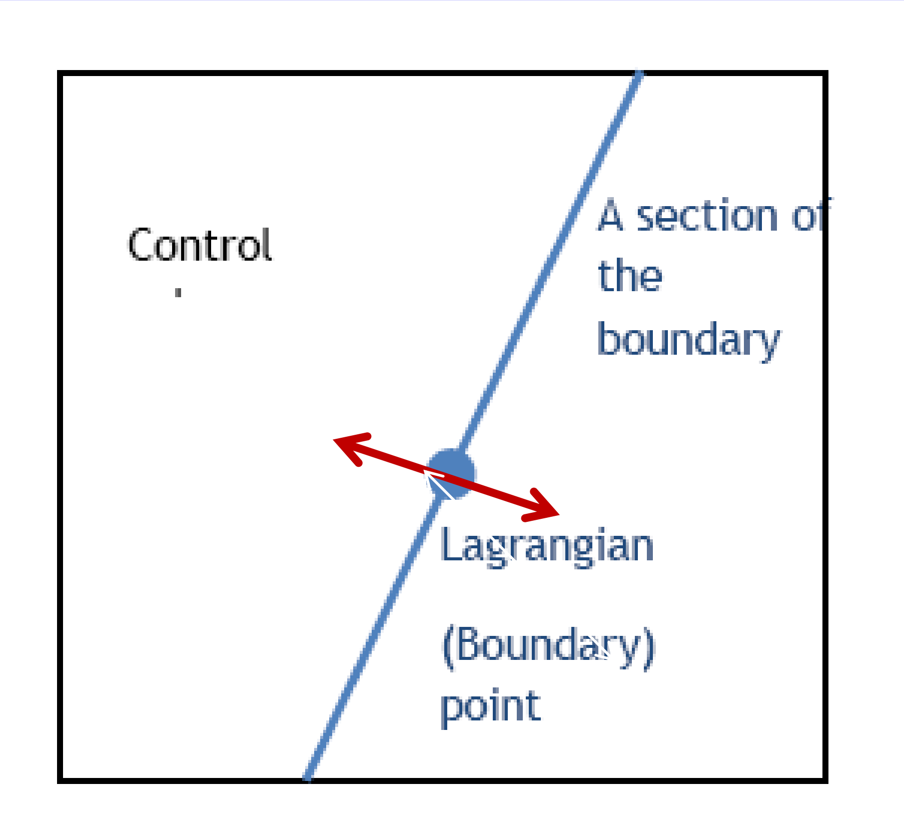
$$q(\mathbf{x}, t) = \int_{\Gamma} Q_B(\mathbf{X}(s), t)\, \delta(\mathbf{x} - \mathbf{X}(s,t))\, ds$$

## Predictor step:

$$\rho c_p \left(\frac{\partial T^*}{\partial t} + (\mathbf{u} \bullet \nabla) T^*\right) = k \nabla^2 T^*$$

**but** $$-k \frac{\partial T^*}{\partial n}(\mathbf{X}) \neq Q_B(\mathbf{X})$$

**Their difference will contribute to $\delta Q_B$ to affect surrounding temperature field**




**At each direction:** $Q_B(\mathbf{X}) + k\dfrac{\partial T^*}{\partial n}(\mathbf{X})$

**Both directions contribute to flow field in a control volume**

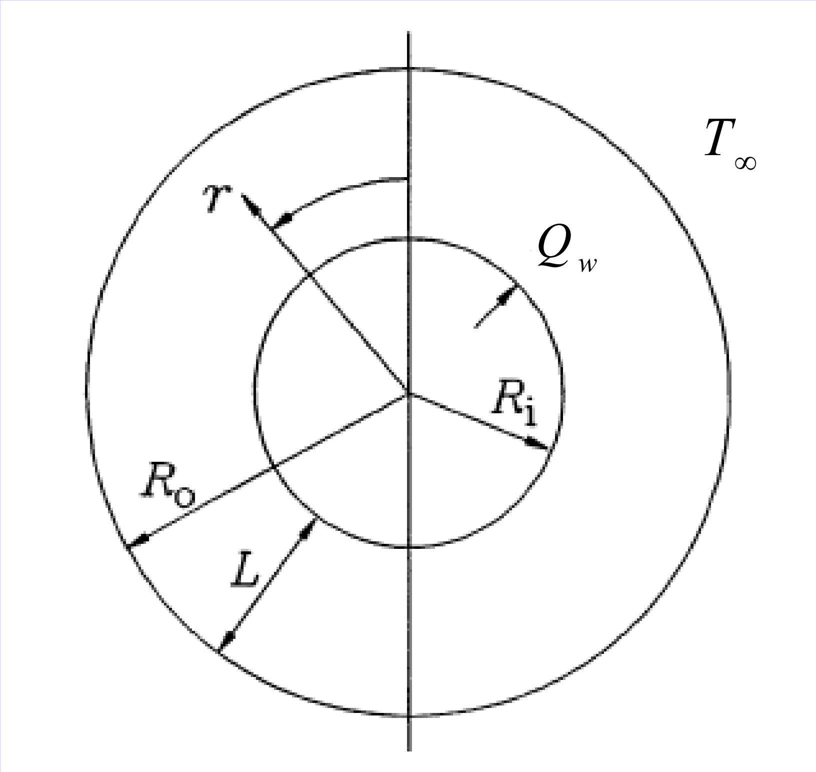
$$q(\mathbf{x},t) = \int_{\Gamma} \delta Q_B(\mathbf{X}(s),t)\delta(\mathbf{x} - \mathbf{X}(s,t))ds$$

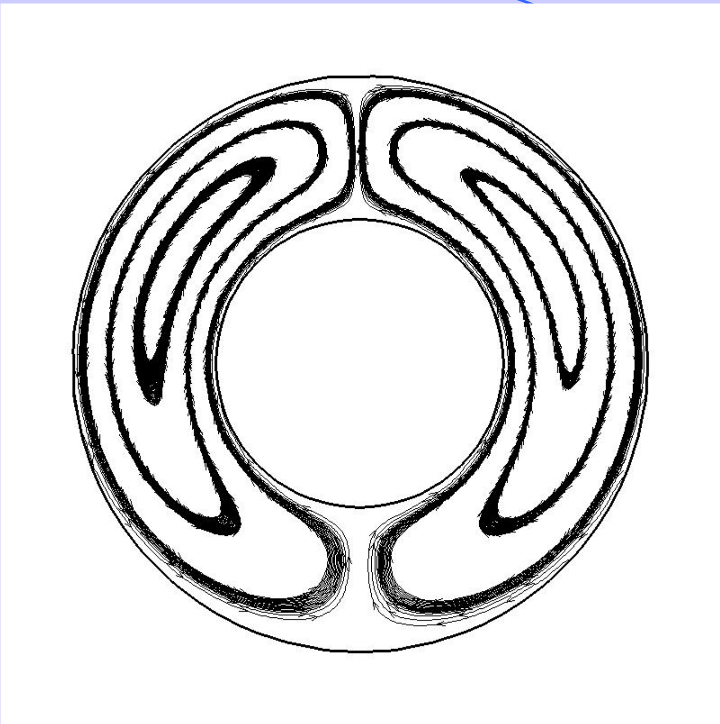
**Corrector step:** $$\rho c_p \frac{\partial T(\mathbf{x},t)}{\partial t} = q(\mathbf{x},t)$$

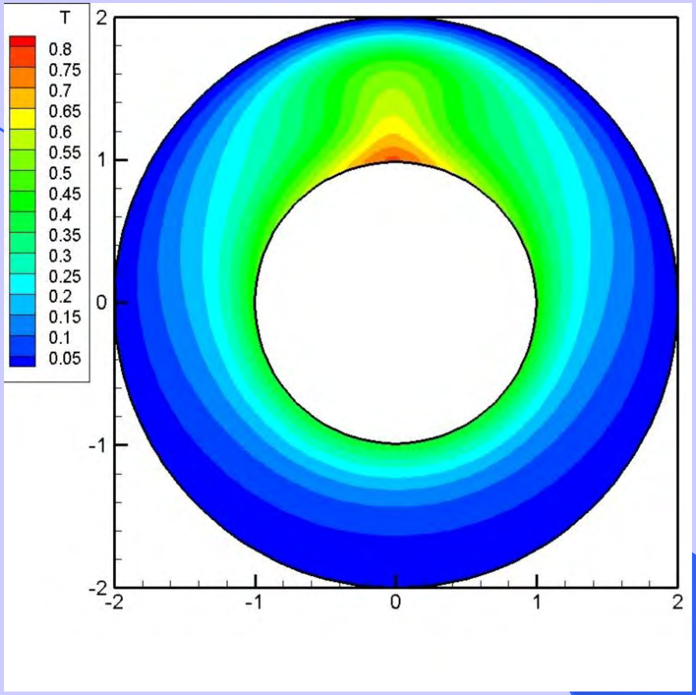
$$\rho c_p \frac{\delta T(\mathbf{x},t)}{\delta t} = q(\mathbf{x},t) \quad \text{with} \quad T^{n+1}(\mathbf{x},t) = T^*(\mathbf{x},t) + \delta T(\mathbf{x},t)$$

**Temperature at boundary point is obtained by delta function interpolation**

# Natural convection in a concentric cylindrical annulus between an outer isothermal cylinder and an inner *isoflux* cylinder

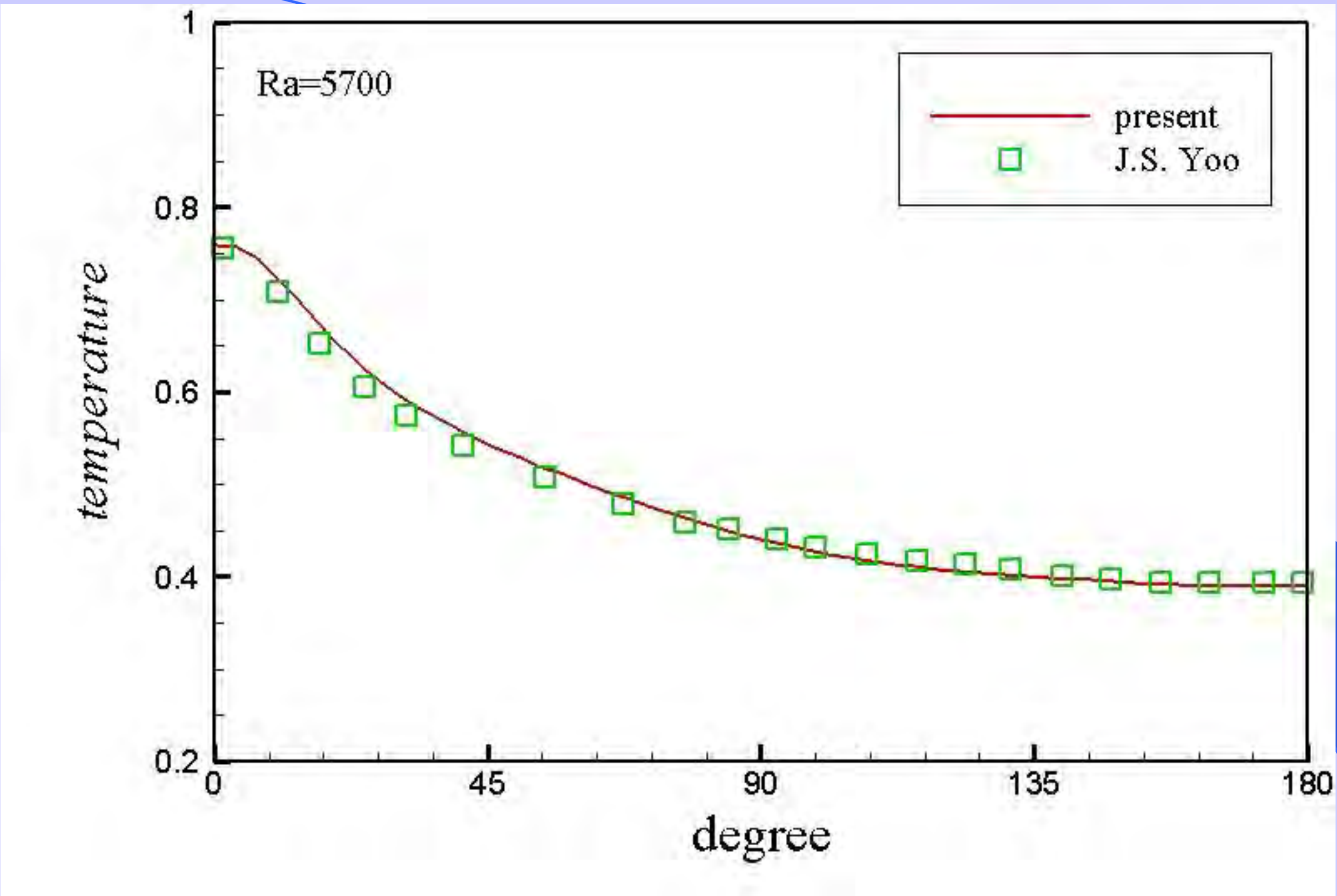







**Streamlines** **Isotherms**

$$Ra = 5700$$




**Comparison of local temperature distribution on the Inner cylinder surface**

## 4.3 Extension to simulate multiphase flows by phase field IB-LBM

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0$$

**Shao,Shu,Chew (JCP, 2012)**

$$\frac{\partial (\rho \boldsymbol{u})}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u}) = -\nabla \cdot P + \mu \nabla^2 \boldsymbol{u} + \boldsymbol{F}_b$$

$$\frac{\partial \phi}{\partial t} + \boldsymbol{u} \nabla \cdot \phi = M \nabla^2 \mu_\phi$$

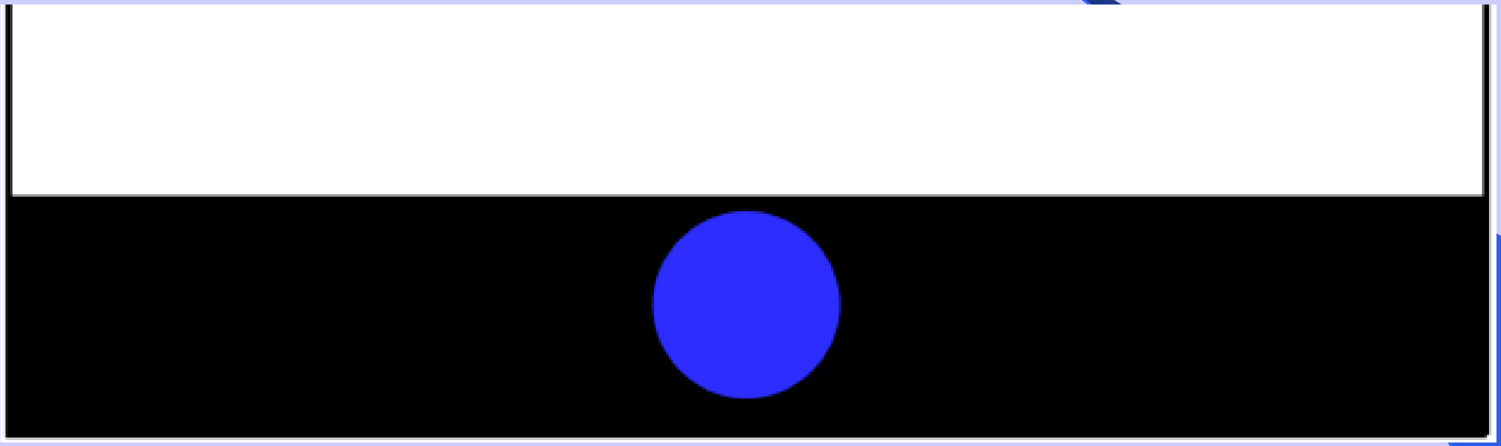
**(C-H Equation)**

**2 Neumann conditions for C-H equation:**

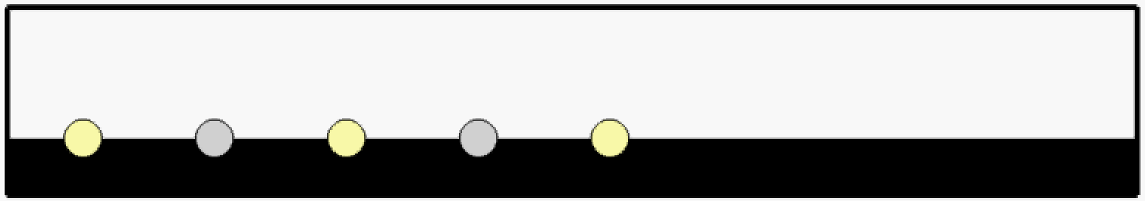
$$\kappa \boldsymbol{n} \cdot (\nabla \phi)_s = -\tilde{\omega}_1$$

$$\boldsymbol{n} \cdot (\nabla \mu_\phi)_s = 0$$

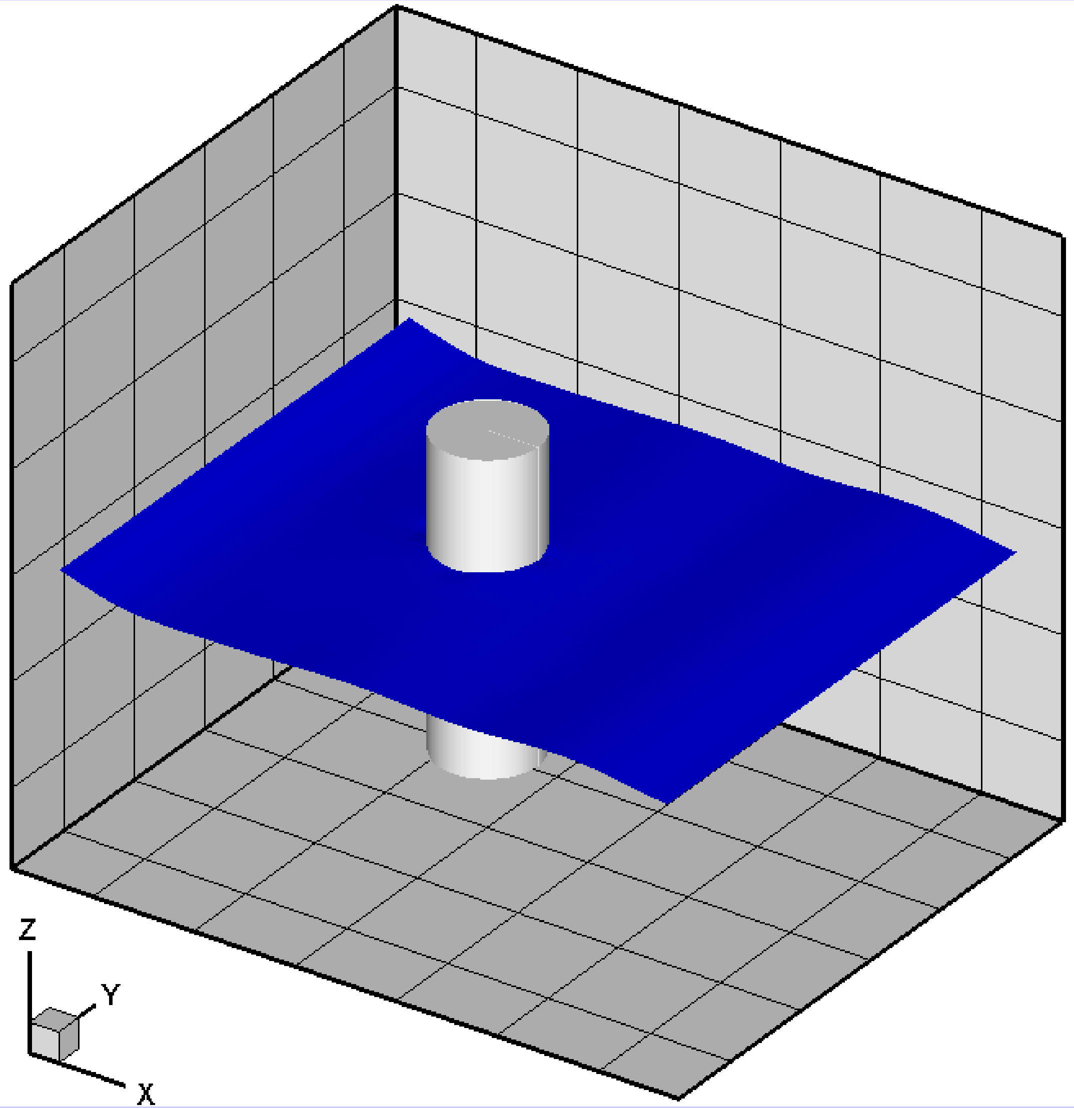
**2 distribution functions used for flow and phase fields**



**A solid particle moves up from water**

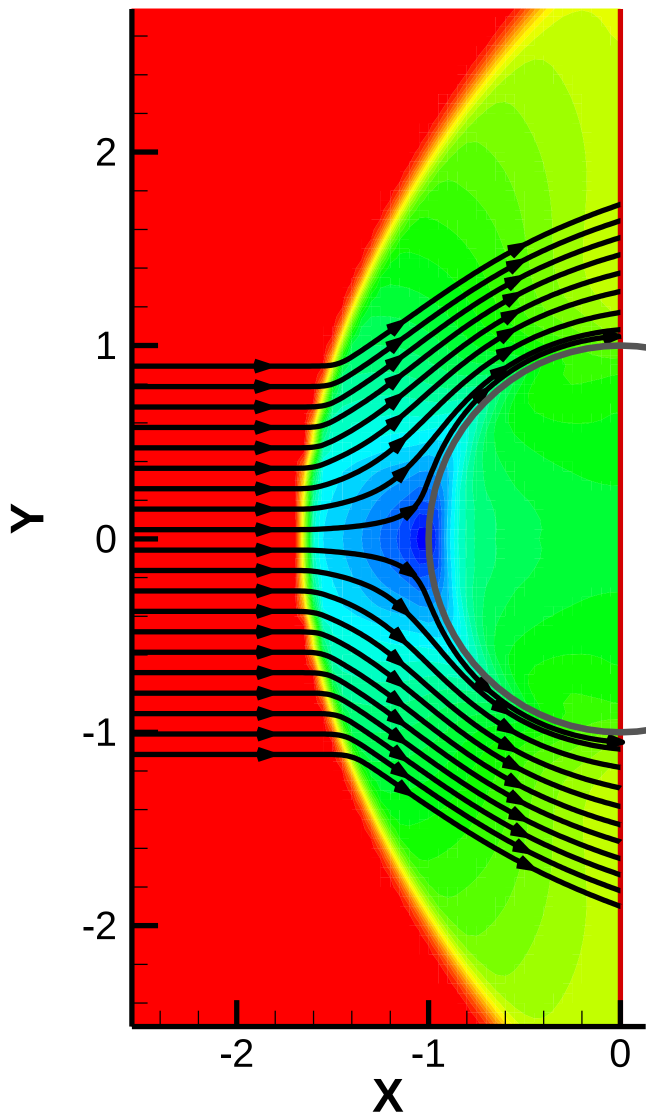


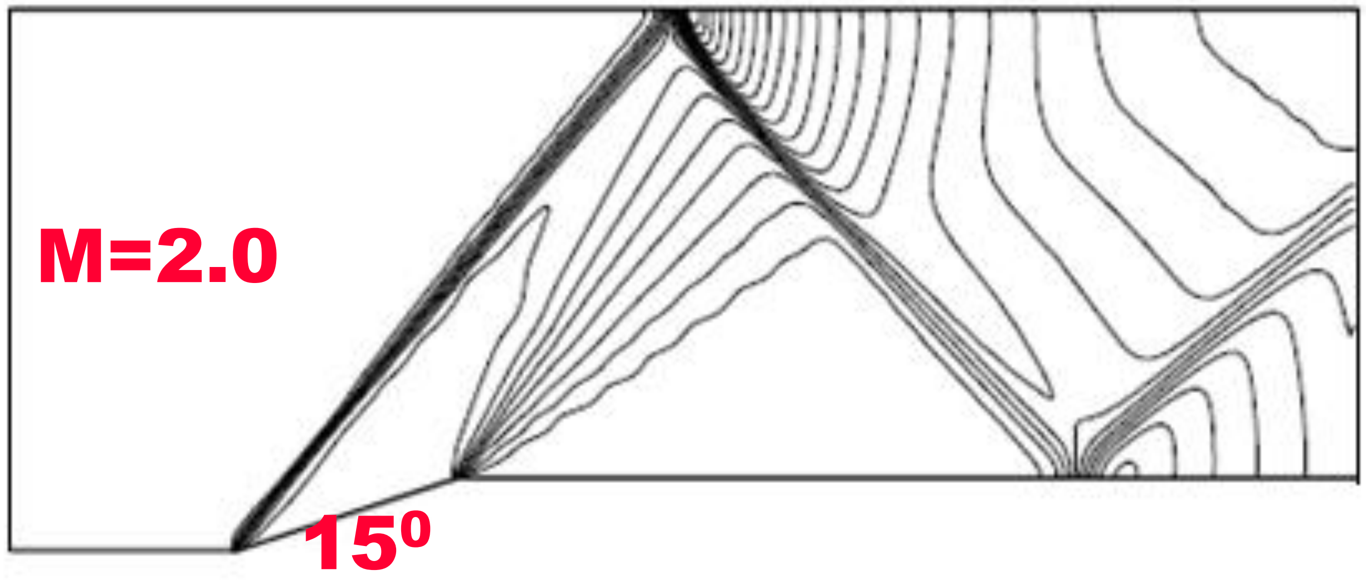
**Interaction of moving particles with slug flows**



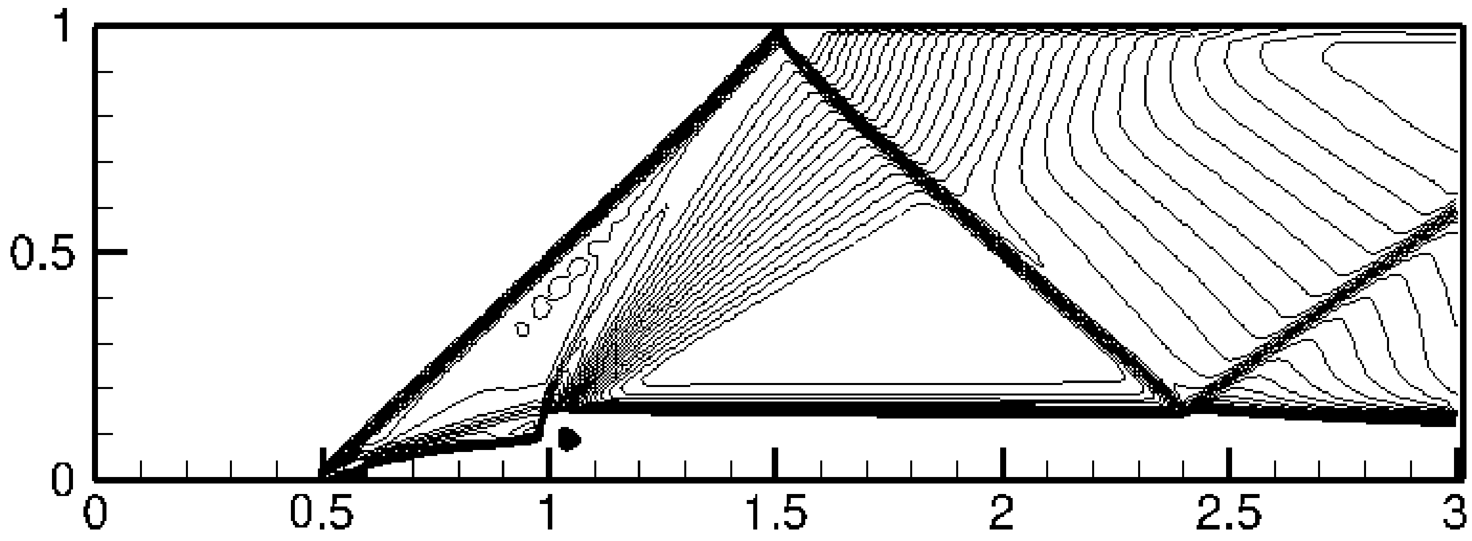

**Simulation of wave-current interaction in a tank**

# 4.3 Extension to simulate compressible Inviscid flows

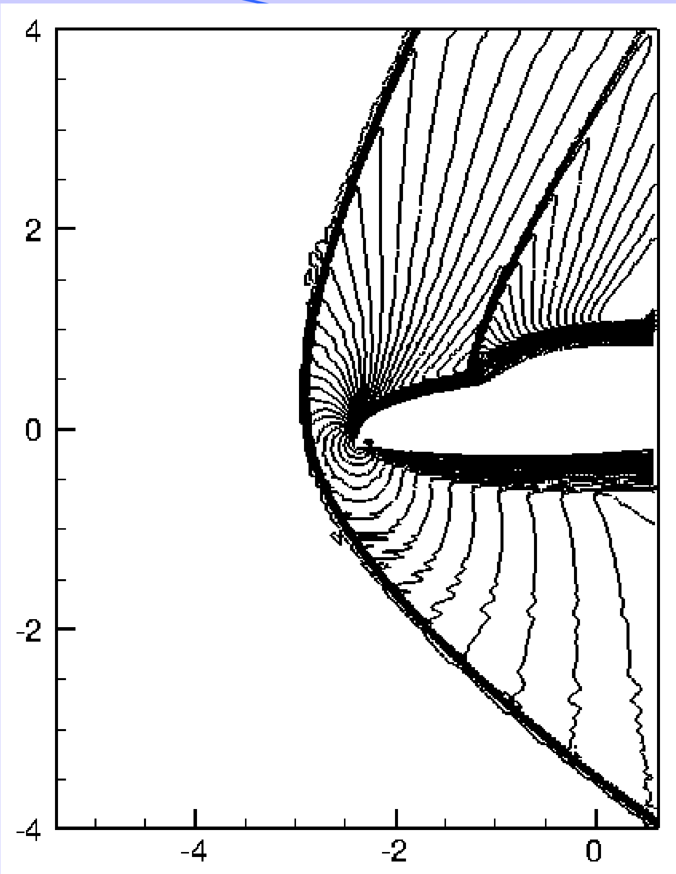


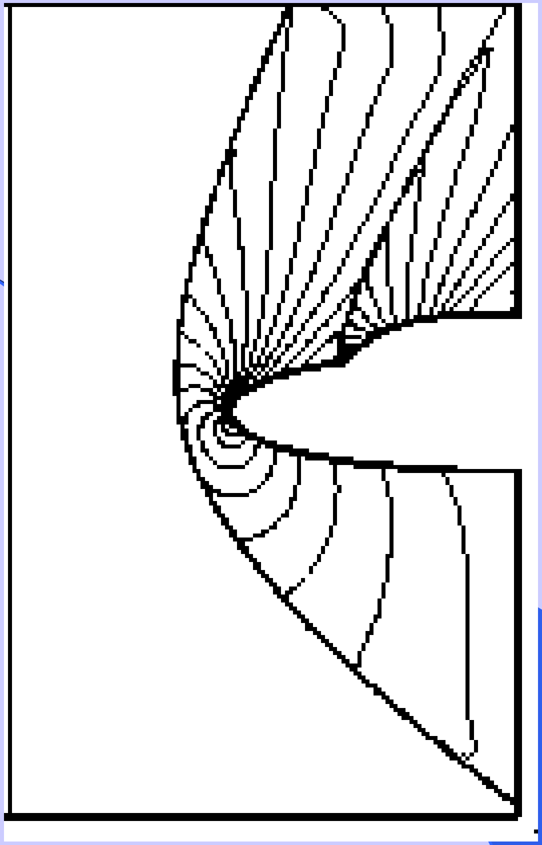



**Body-fitted results**



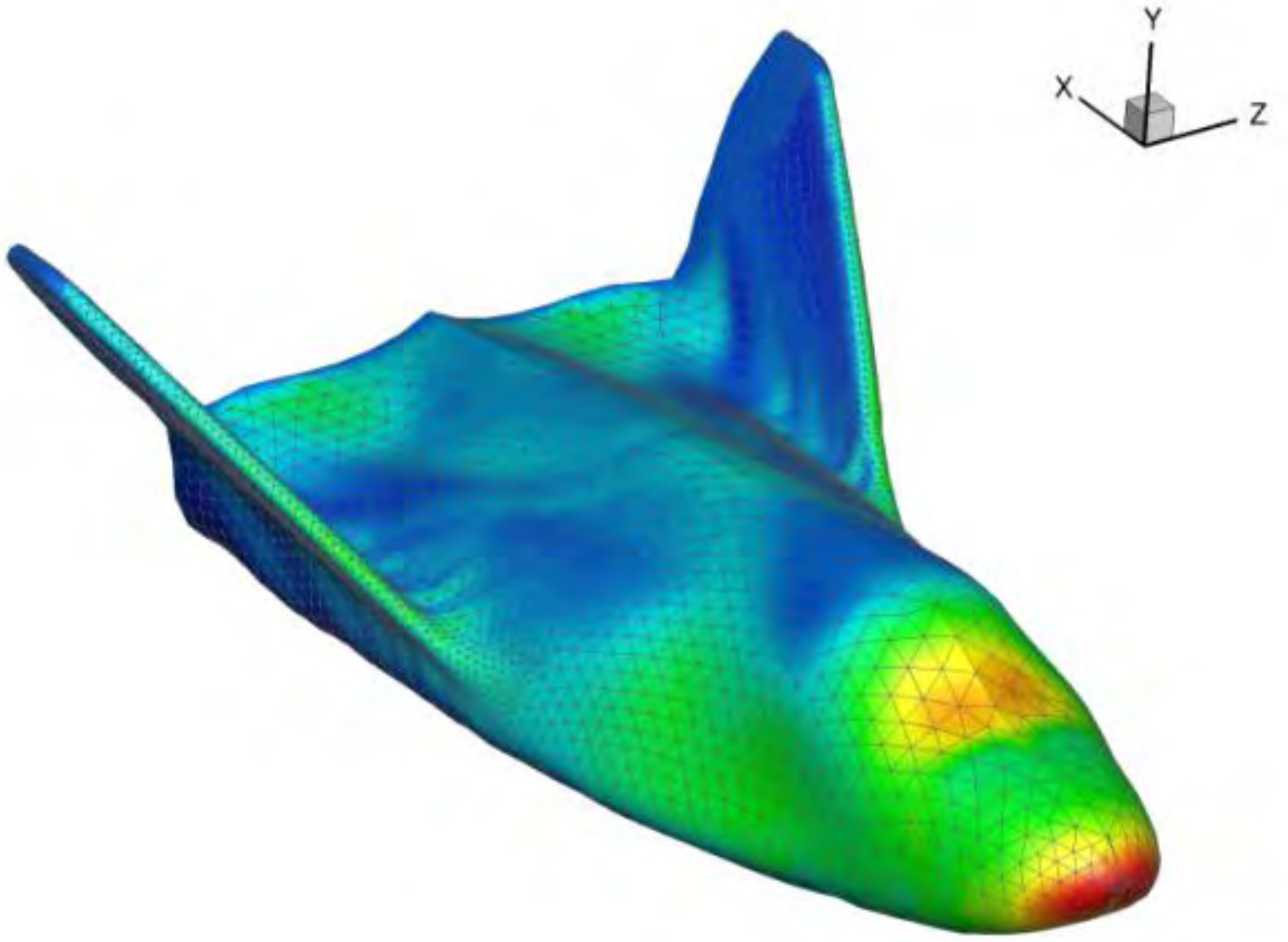

**IBM results**



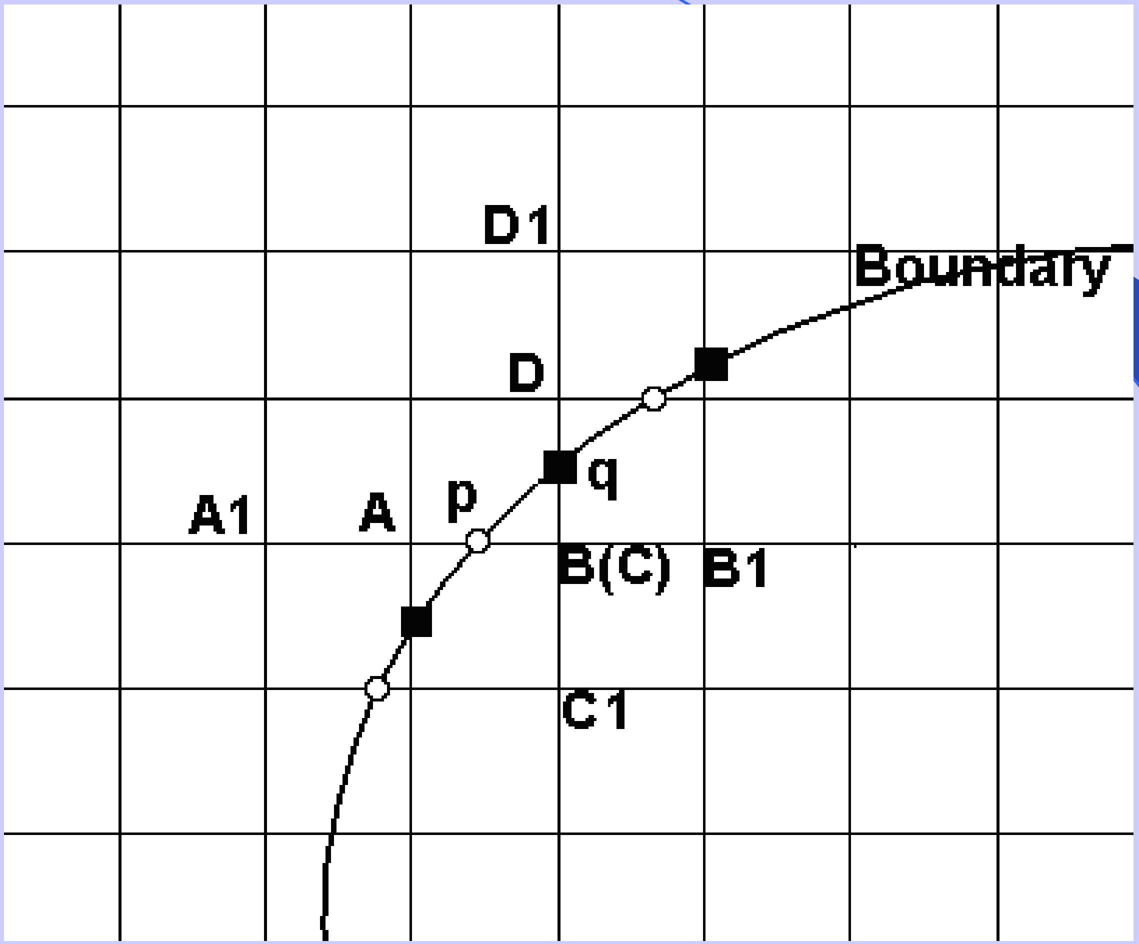
**IBM result** **Body-fitted result**

**M=2.0, angle of attack = $20^0$**




M=2.5

# 4.4 Direct implementation of no-slip condition with 2nd order of accuracy




**Interpolation from boundary point to both sides**

## LB Model of Guo, Zheng & Shi (PRE, 2002)

$$\rho \mathbf{u}^{*} = \sum_{\alpha} \mathbf{e}_{\alpha} f_{\alpha}$$

$$\rho \mathbf{u}^{c} = \rho \mathbf{u}^{*} + \frac{1}{2} \mathbf{f} \Delta t$$

$$\mathbf{f} = 2\rho \frac{\left(\mathbf{u}^{c} - \mathbf{u}^{*}\right)}{\Delta t}$$

**Velocity correction and force calculation are only applied at boundary-dependent points**

## Along horizontal mesh lines:

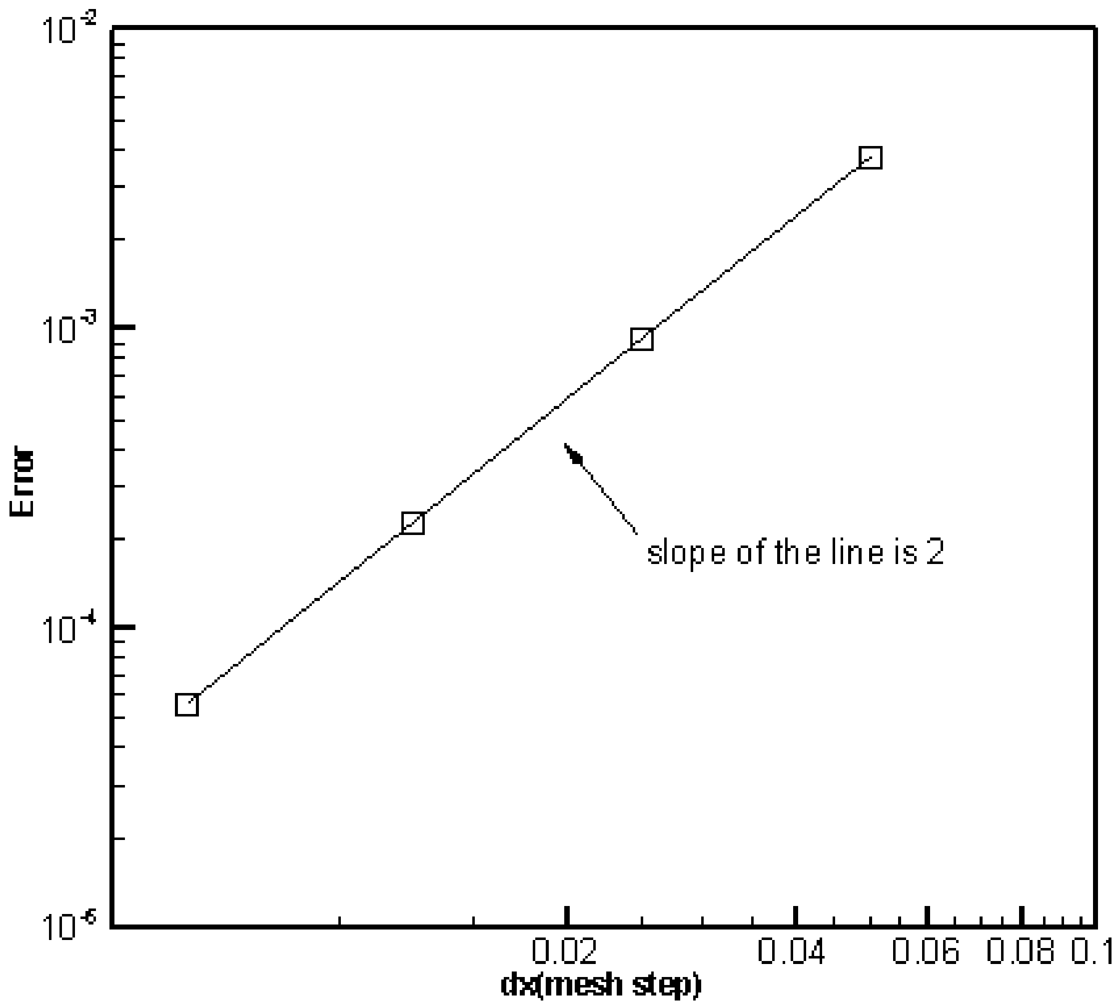
$$u_A^c = \frac{(x_A - x_{A1})(x_A - x_{B1})}{(x_p - x_{A1})(x_p - x_{B1})} u_p + \frac{(x_A - x_p)(x_A - x_{B1})}{(x_{A1} - x_p)(x_{A1} - x_{B1})} u_{A1} + \frac{(x_A - x_{A1})(x_A - x_p)}{(x_{B1} - x_{A1})(x_{B1} - x_p)} u_{B1}$$

$$v_A^c = \frac{(x_A - x_{A1})(x_A - x_{B1})}{(x_p - x_{A1})(x_p - x_{B1})} v_p + \frac{(x_A - x_p)(x_A - x_{B1})}{(x_{A1} - x_p)(x_{A1} - x_{B1})} v_{A1} + \frac{(x_A - x_{A1})(x_A - x_p)}{(x_{B1} - x_{A1})(x_{B1} - x_p)} v_{B1}$$

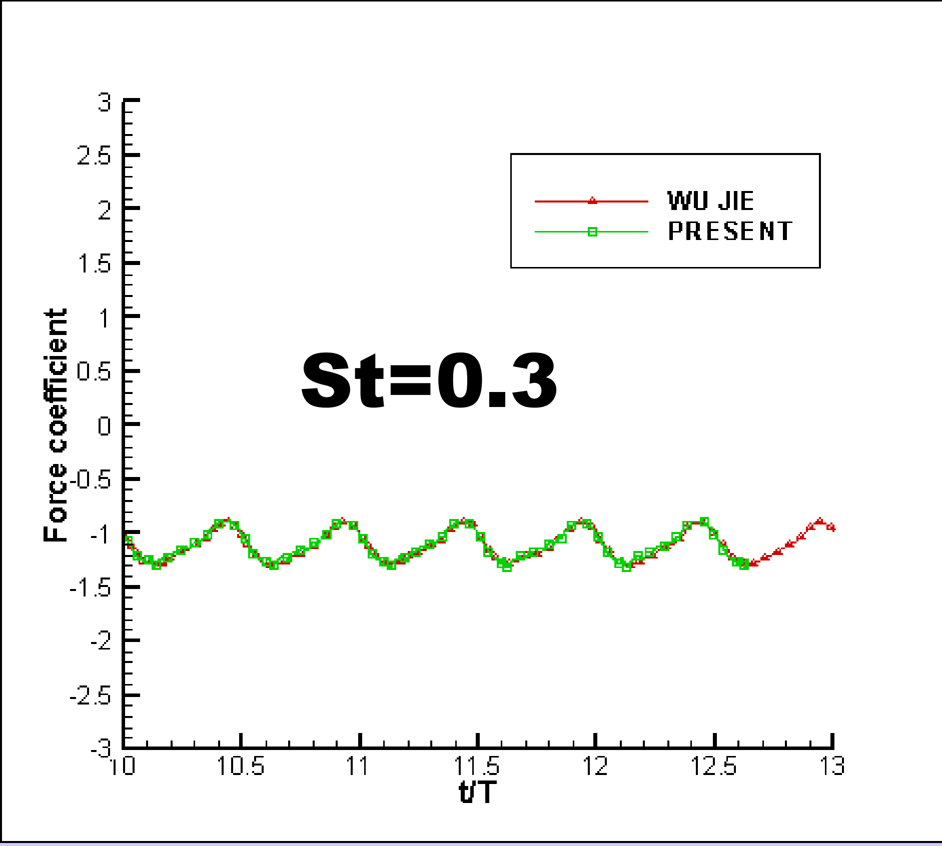
## Along vertical mesh lines:

$$u_D^c = \frac{(y_D - y_{C1})(y_D - y_{D1})}{(y_q - y_{C1})(y_q - y_{D1})} u_q + \frac{(y_D - y_q)(y_D - y_{D1})}{(y_{C1} - y_q)(y_{C1} - y_{D1})} u_{C1} + \frac{(y_D - y_{C1})(y_D - y_q)}{(y_{D1} - y_{C1})(y_{D1} - y_q)} u_{D1}$$

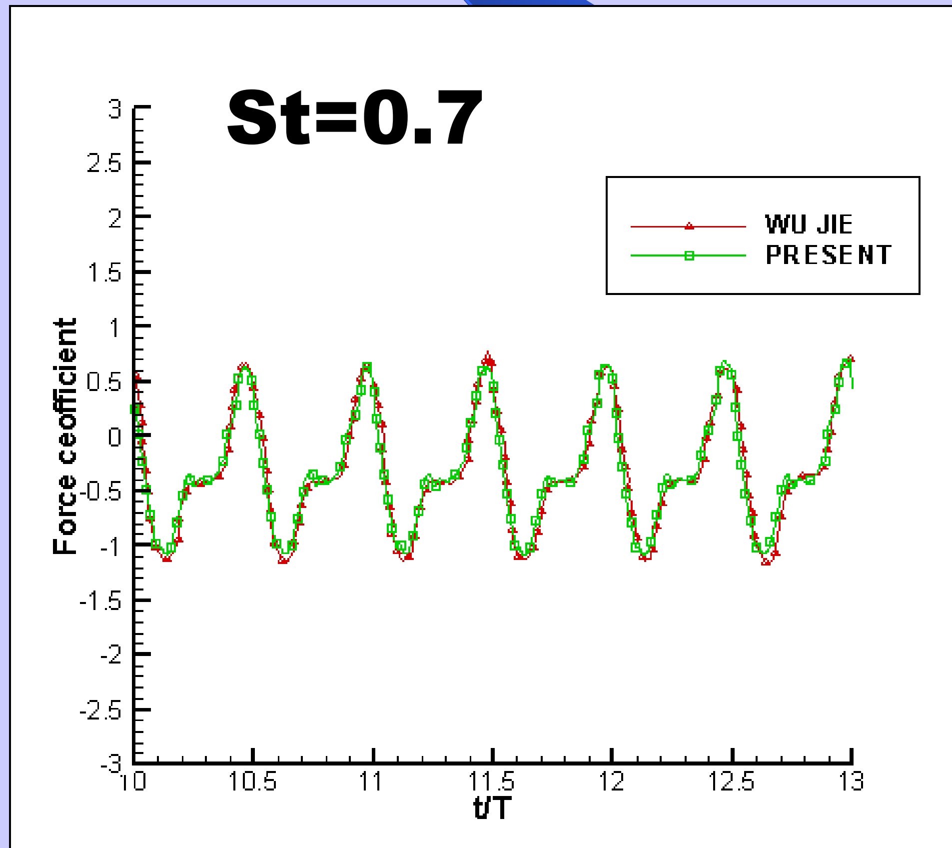
$$v_D^c = \frac{(y_D - y_{C1})(y_D - y_{D1})}{(y_q - y_{C1})(y_q - y_{D1})} v_q + \frac{(y_D - y_q)(y_D - y_{D1})}{(y_{C1} - y_q)(y_{C1} - y_{D1})} v_{C1} + \frac{(y_D - y_{C1})(y_D - y_q)}{(y_{D1} - y_{C1})(y_{D1} - y_q)} v_{D1}$$




# Accuracy study for decaying vortex problem




# 3D Fish Motion



# 5. Conclusions

- **Boundary condition-enforced IBM can accurately satisfy boundary conditions; No flow penetration at the solid boundary;**
- **Force and heat transfer rate can be easily calculated;**
- **It has a great potential for applications (keep advantages of IBM but provide more accurate results)**

**http://serve.me.nus.edu.sg/shuchang**

# Thank You