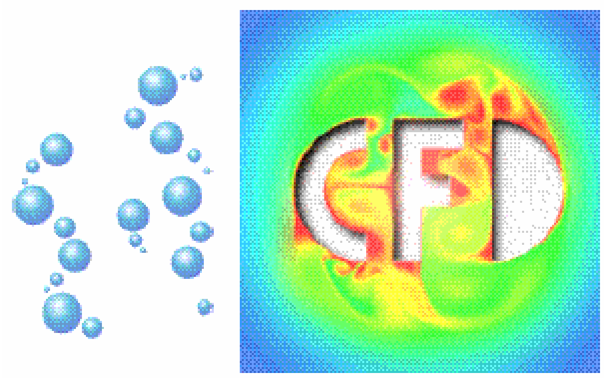
# Fictitious Boundary and Moving Mesh Methods for the Numerical Simulation of Particulate Flow


## Stefan Turek, Dan Anca, Decheng Wan

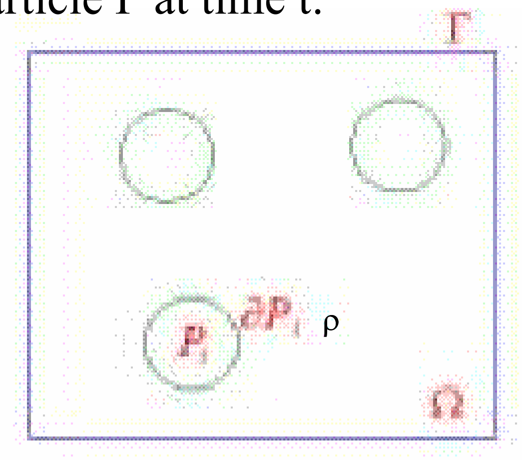
Institut für Angewandte Mathematik, Univ. Dortmund
http://www.mathematik.uni-dortmund.de/LS3
http://www.featflow.de


- Model for particulate flow
- Numerical techniques



# *Fluid – (Rigid) Solid Interfaces*

Consider flow of N solid particles in a fluid with density $\rho$ and viscosity $\mu$.
Denote by $\Omega_f(t)$ the domain occupied by the fluid at time t , and by $\Omega_p(t)$ domain occupied by the particle P at time t:




Fluid flow is modelled by the **Navier-Stokes equations** in $\Omega_f(t)$:

$$\rho\left(\frac{\partial u}{\partial t} + u \cdot \nabla u\right) - \nabla \cdot \sigma = f, \qquad \nabla \cdot u = 0$$

where $\sigma$ is the total stress tensor in the fluid phase, which is defined as :

$$\sigma(X,t) = -pI + \mu[\nabla u + (\nabla u)^T]$$

# Model for Particle Motion (I)

Motion of particles is described by the **Newton-Euler equations**, i.e., the **translational velocities** $U_p$ and **angular velocities** $\omega_p$ of the p-th particle satisfy:

$$M_p \frac{dU_p}{dt} = F_p + F_p^{'} + \left(\Delta M_p\right) g, \quad I_p \frac{d\omega_p}{dt} + \omega_p \times \left(I_p \omega_p\right) = T_p.$$

with $M_p$ the mass of the p-th particle ( p =1,…,N);

$I_p$ the moment of inertia tensor of the p-th particle;

$\Delta M_p$ the mass difference between the mass $M_p$ and the mass of the fluid

occupying the same volume.

# Model for Particle Motion (II)

$F_p$ and $T_p$ are the **hydrodynamical forces** and the **torque** at mass center acting on the p-th particle:

$$F_p = -\int_{\Gamma_p} \sigma \cdot n_p \, d\Gamma_p , \quad T_p = -\int_{\Gamma_p} \left( X - X_p \right) \times \left( \sigma \cdot n_p \right) d\Gamma_p$$

and $F_p^{'}$ are the **collision** or **agglomeration forces** (later).

$X_p$ is the position of the center of gravity of the p-th particle;

$\Gamma_p = \partial \Omega_p$ the boundary of the p-th particle;

$n_p$ is the unit normal vector on the boundary $\Gamma_p$.

# *Interaction between Particle and Fluid*

**No slip boundary conditions** at interface $\Gamma_p$ between particles and fluid i.e., for any $X \in \Gamma_p$, the velocity **u**(X) is defined by:

$$u(X) = U_p + \omega_p \times (X - X_p)$$

The **position** $X_p$ of the p-th particle and its **angle** $\theta_p$ are obtained by integration of the kinematic equations:

$$\frac{dX_p}{dt} = U_p, \qquad \frac{d\theta_p}{dt} = \omega_p$$

# Classification

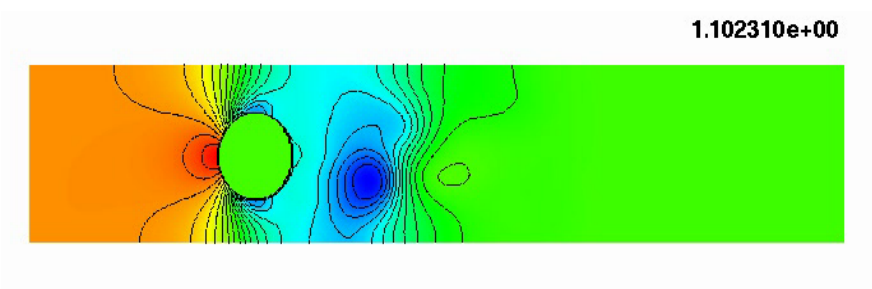
**Explicit coupling**

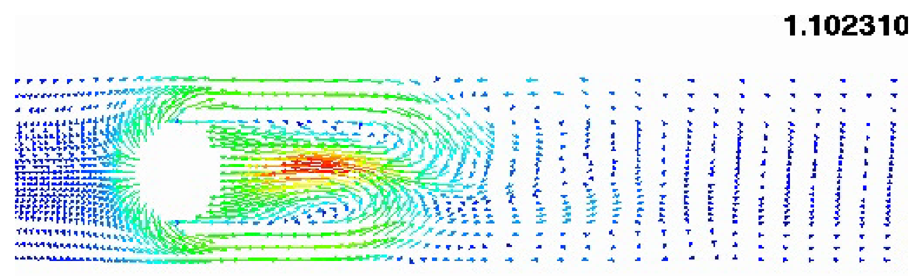
$\boxed{t^n \text{ fluid}} \rightarrow \boxed{t^n \text{ force on solid}} \rightarrow \boxed{t^{n+1} \text{solid}} \rightarrow \boxed{t^{n+1} \text{ fluid}}$

**Eulerian approach:** fixed meshes!

Use a **fixed** mesh that covers the whole domain where the fluid may be present.

## FEM-fictitious boundary methods






**Computational mesh (can be) independent of ‘internal objects’**

# Grid Deformation Method

**idea** : construct transformation $\phi$, $x = \phi(\xi, t)$ with $\det \nabla \phi = f$

⇨ local mesh area $\approx f$

1. Compute monitor function $f(x,t) > 0$, $f \in C^1$ and

$$\int_\Omega f^{-1}(x,t)\,dx = |\Omega|, \quad \forall t \in [0,1]$$

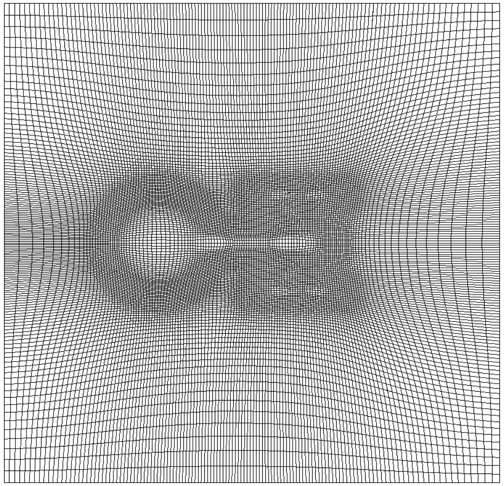
2. Solve $(t \in [0,1])$

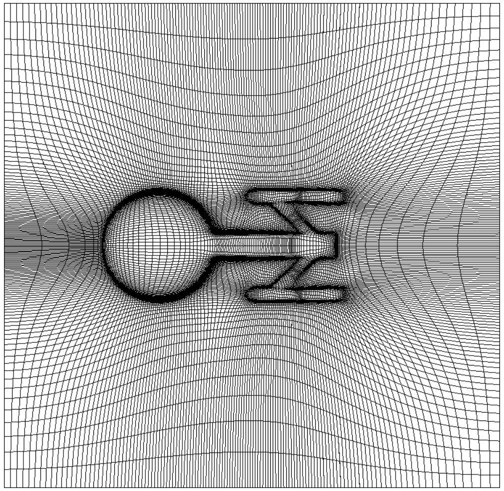
$$\Delta v(\xi, t) = -\frac{\partial}{\partial t}\left(\frac{1}{f(\xi, t)}\right), \quad \left.\frac{\partial v}{\partial n}\right|_{\partial\Omega} = 0$$

3. Solve the ODE system

$$\frac{\partial}{\partial t}\phi(\xi, t) = f(\phi(\xi, t), t)\nabla v(\phi(\xi, t), t)$$

new grid points: $x_i = \phi(\xi_i, 1)$





**Grid deformation preserves the (local) logical structure of the grid**

# How to Calculate Forces with FBM?

**Hydrodynamic forces and torque acting on the i-th particle**

$$F_i = -\int_{\partial P_i} \sigma \cdot n_i d\Gamma_i , \qquad T_i = -\int_{\partial P_i} (X - X_i) \times (\sigma \cdot n_i) d\Gamma_i$$

- FBM: 0/1 - reconstruction of the shape is only 1st order accurate
  ➔ local grid adaptivity or alignment near interface
  ➔ ”only” averaged/integral quantities are required

- But: The FBM can only decide ”INSIDE” or ”OUTSIDE”

**Idea: ‘Replace the surface integral by a volume integral’**

# Calculation of Hydrodynamic Forces

Define auxiliary function $\alpha$ as

$$\alpha_p(X) = \begin{cases} 1 & \text{for} \quad X \in \Omega_p \\ 0 & \text{for} \quad X \in \Omega_f \end{cases}$$

**Remark:** $\nabla \alpha_p = 0$ everywhere except at wall surface of the particles, and equal to the normal vector $n_p$ defined on the global grid.

$$n_p = \nabla \alpha_p$$

Force acting on the wall surface of the particles can be computed by

$$F_p = -\int_{\Gamma_p} \sigma \cdot n_p \, d\Gamma_p = -\int_{\Omega_T} \sigma \cdot \nabla \alpha_p \, d\Omega_T$$

with $\overline{\Omega}_T = \overline{\Omega}_f \cup \overline{\Omega}_p$ (analogously for the torque)

# (Explicit) Operator-Splitting Approach

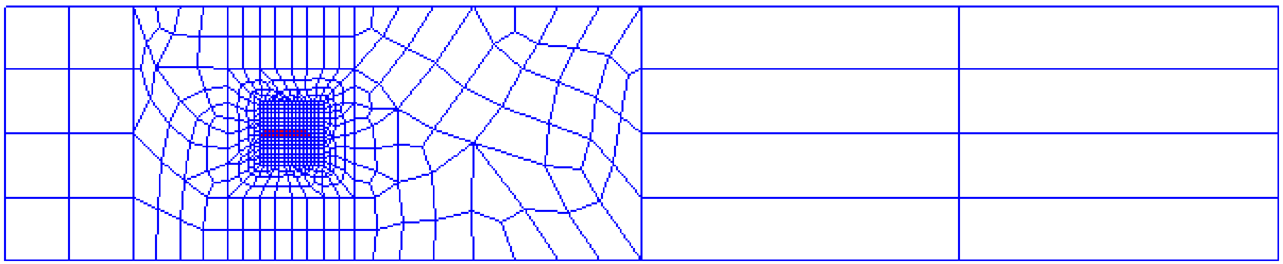
The algorithm for $t^n \rightarrow t^{n+1}$ consists of the following 4 substeps

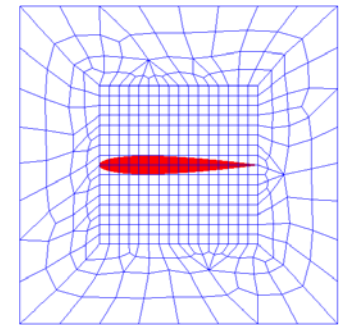
1. Fluid velocity and pressure : $NSE\left(u_f^{n+1}, p^{n+1}\right) = BC\left(\Omega_p^n, u_p^n\right)$
2. Calculate hydrodynamic forces: $F_p^{n+1}$
3. Calculate velocity of particles: $u_p^{n+1} = g\left(F_p^{n+1}\right)$
4. Update position of particles: $\Omega_p^{n+1} = f\left(u_p^{n+1}\right)$
5. Align new mesh

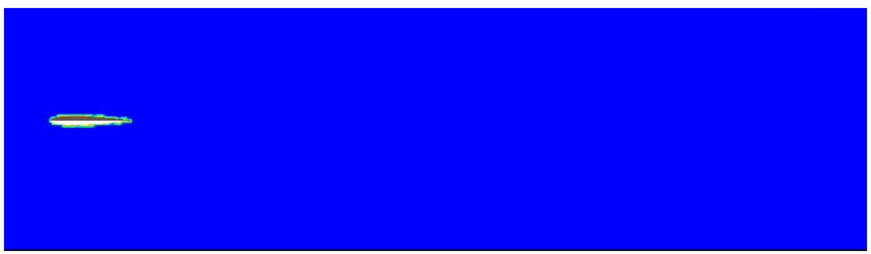
→ Required: efficient calculation of hydrodynamic forces
→ Required: efficient treatment of particle interaction (?)
→ Required: fast (nonstationary) Navier-Stokes solvers (!)

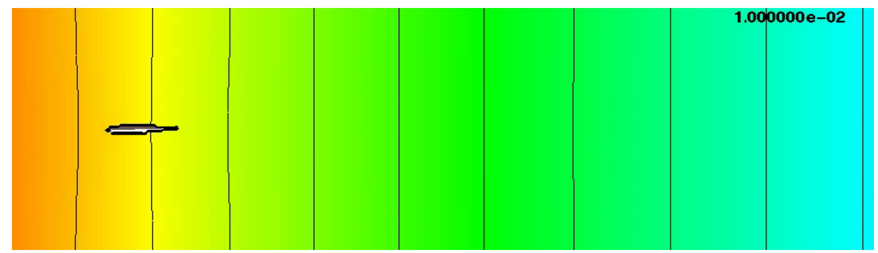
# Numerical Examples

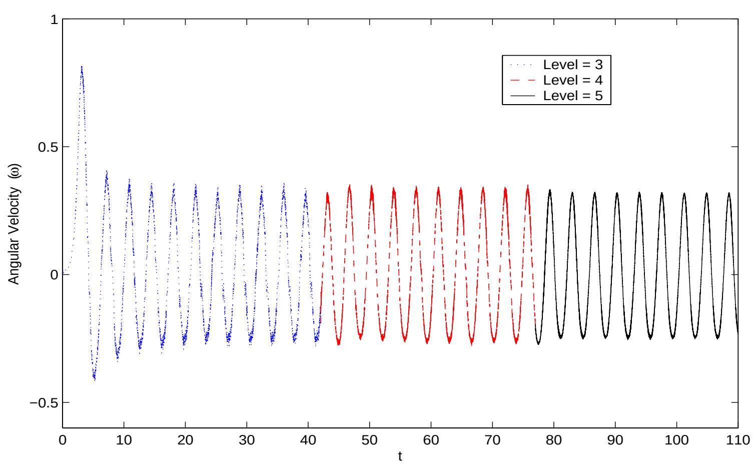
## ‘Viscous flow around a moving airfoil’ (Glowinski)

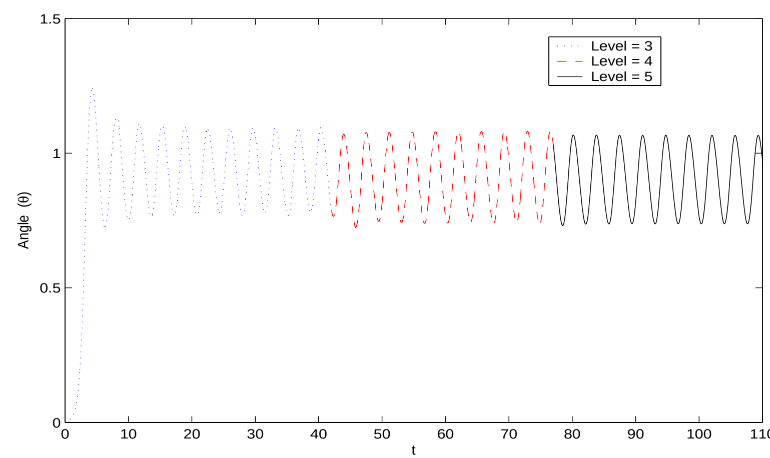




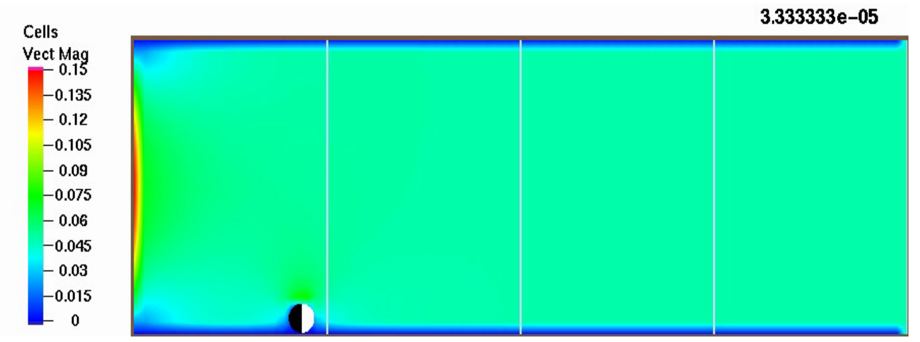


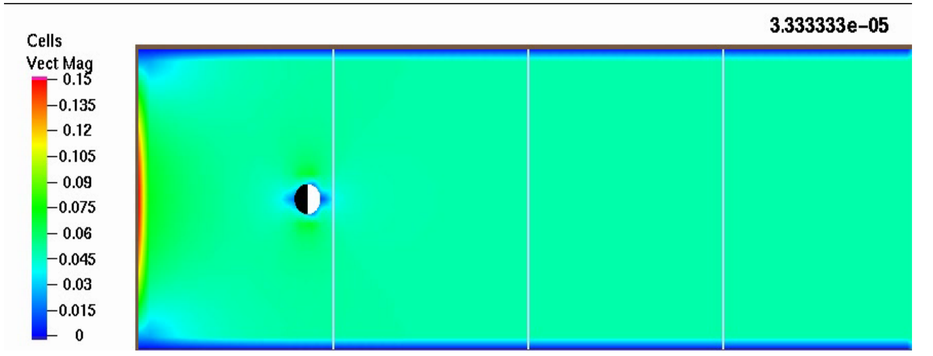


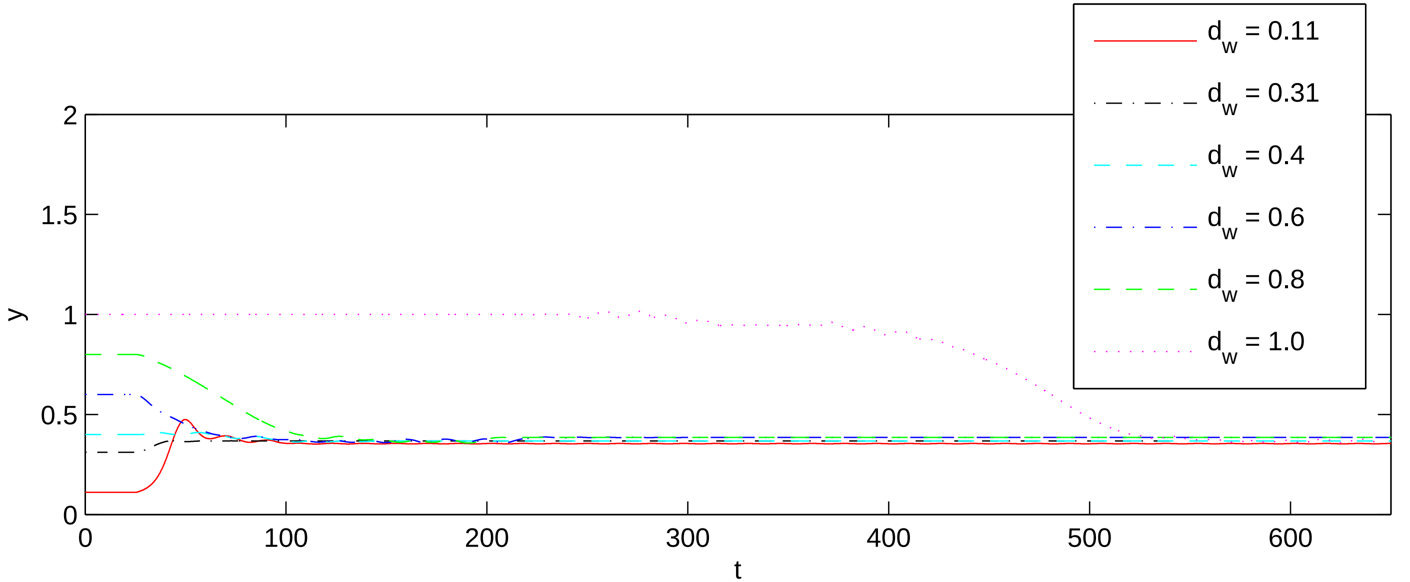
# Lift-Off for Circle



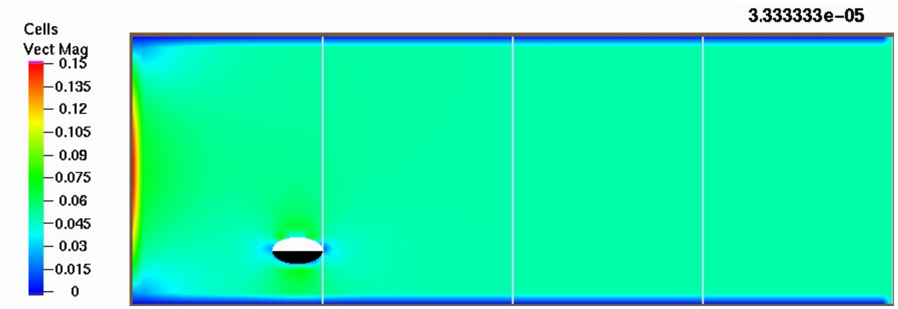

Velocity ($d_w = 0.1$)

Velocity ($d_w = 1.0$)

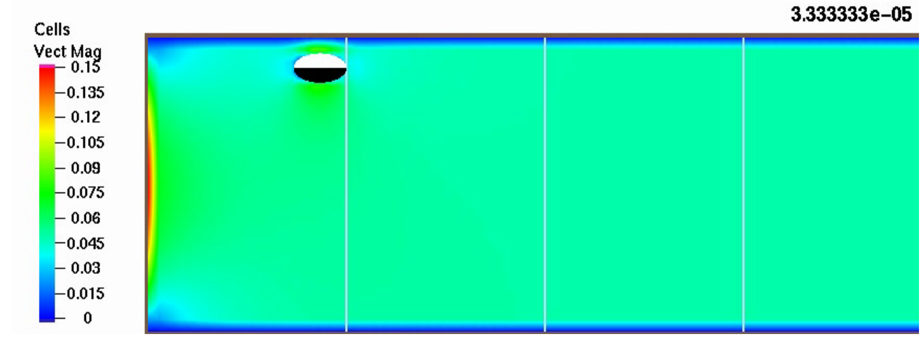



$y$ of center of ball

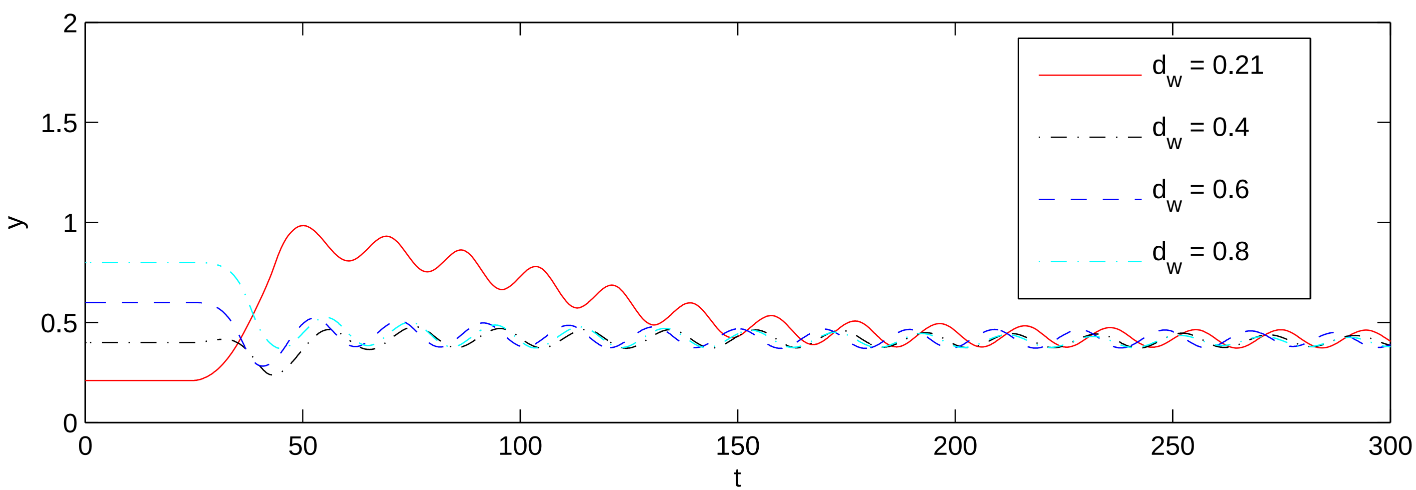
# Lift-Off for Ellipse




Velocity ($d_w = 0.4$)



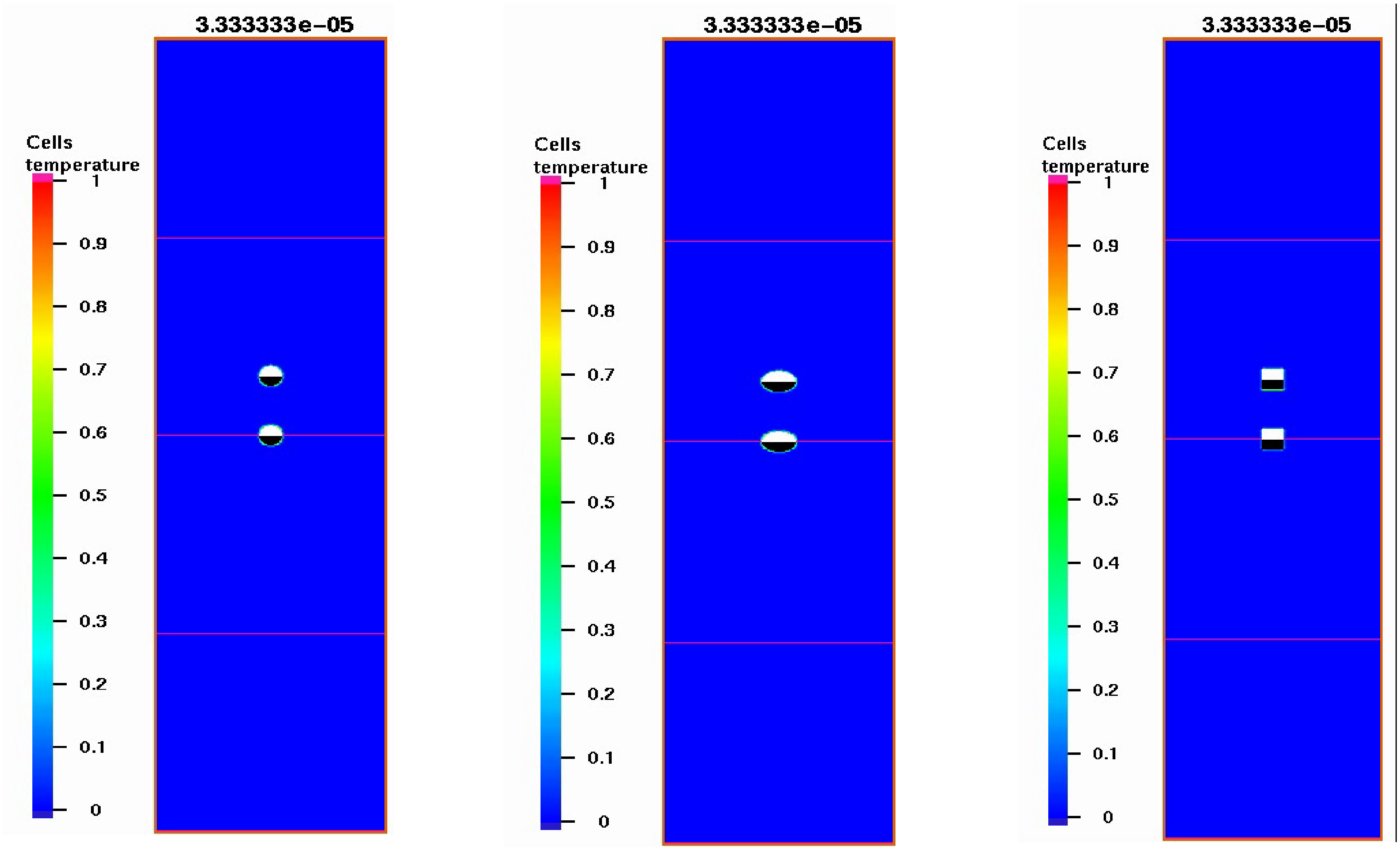

Velocity ($d_w = 1.8$)




$y$ of center of ellipse

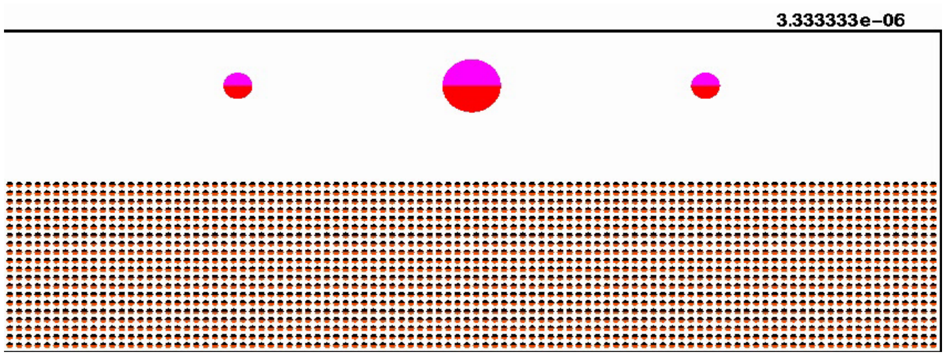
# Numerical Examples

## ‘Kissing, Drafting, Thumbling’

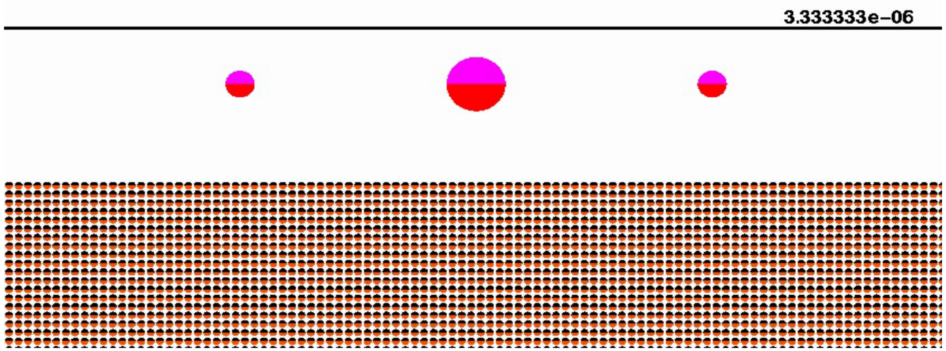


# Numerical Examples

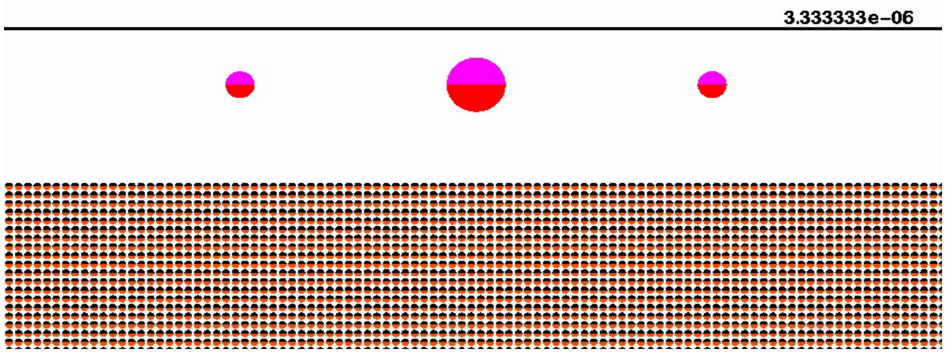
**'Impact of heavy balls on 2000 small particles'**




$$\rho_f = 1$$
$$\rho_{bd} = 2$$
$$\rho_{sp} = 1.1$$




$$\rho_f = 1$$
$$\rho_{bd} = 2$$
$$\rho_{sp} = 2$$

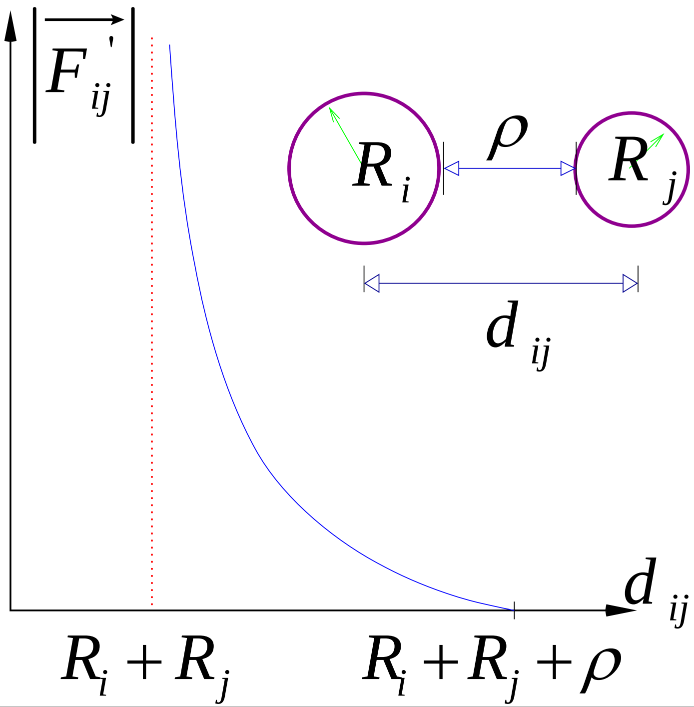



$$\rho_f = 1$$
$$\rho_{bd} = 2$$
$$\rho_{sp} = 20$$

# Collision Models

- **Theoretically**, it is impossible that smooth particle-particle collisions take place in finite time in the **continuous system** since there are repulsive forces to prevent these collisions in the case of viscous fluids.

- **In practice**, however, particles can contact or even overlap each other in **numerical simulations** since the gap can become arbitrarily small due to unavoidable numerical errors.




$$\begin{cases} \left|\overrightarrow{F_{ij}'}\right| = 0, & d_{ij} \geq R_i + R_j + \rho, \\ \left|\overrightarrow{F_{ij}'}\right| = \dfrac{d_{ij}}{\varepsilon}, & d_{ij} = R_i + R_j. \end{cases}$$

# Repulsive Force Collision Model

- Handling of small gaps and contact between particles
- Dealing with overlapping in numerical simulations

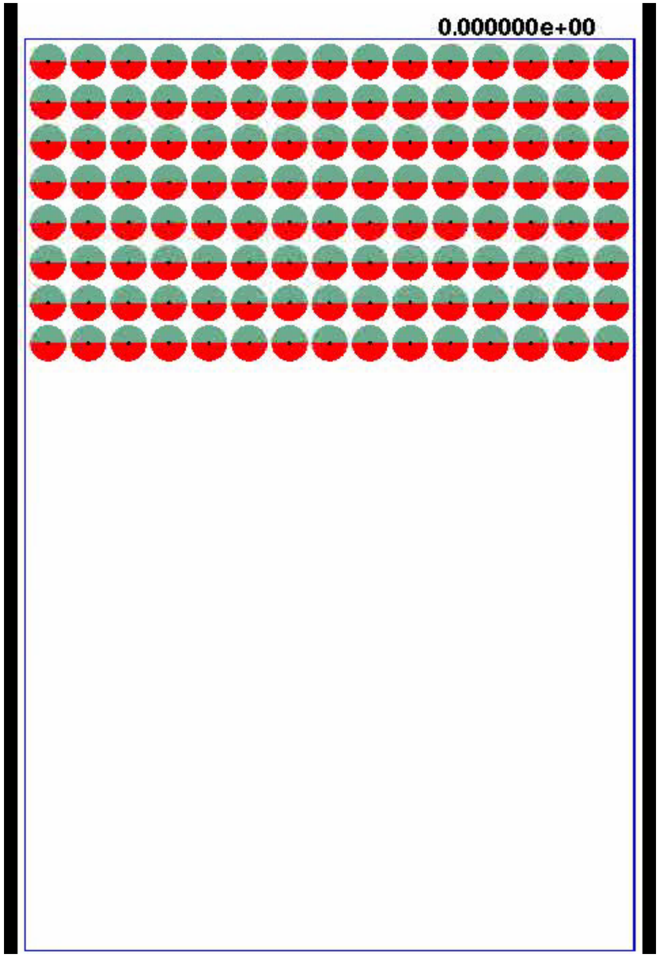
For the particle-particle collisions (analogous for the particle-wall collisions), the repulsive forces between particles read:

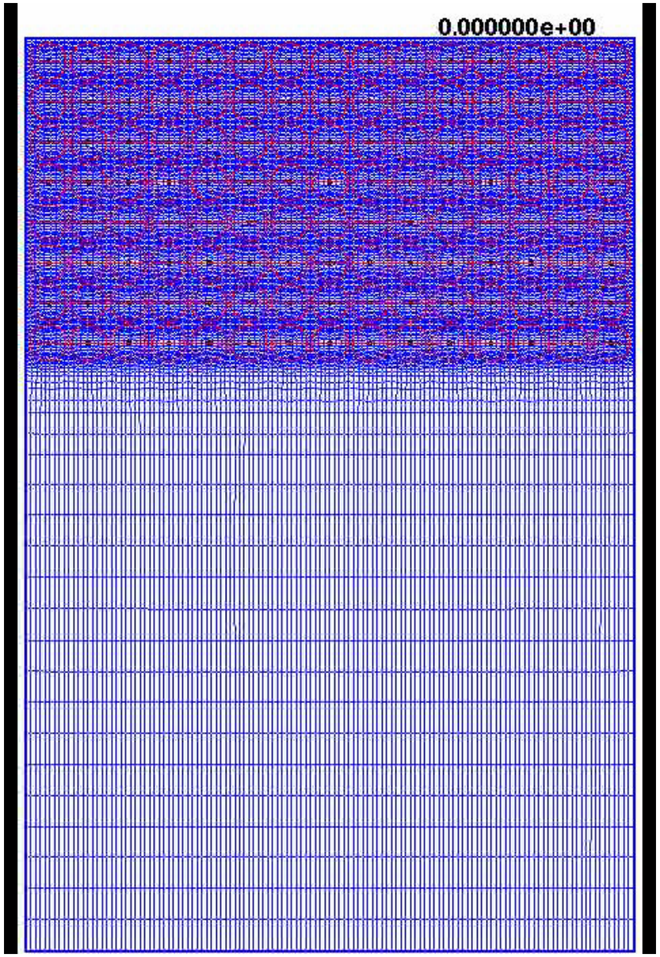
$$F_{ij}^{P} = \begin{cases} 0 & \text{for} \quad d_{i,j} > R_i + R_j + \rho \\ \dfrac{1}{\varepsilon_P}\left(X_i - X_j\right)\left(R_i + R_j + \rho - d_{i,j}\right)^2 & \text{for} \quad R_i + R_j \le d_{i,j} \le R_i + R_j + \rho \\ \dfrac{1}{\varepsilon_P^{'}}\left(X_i - X_j\right)\left(R_i + R_j - d_{i,j}\right) & \text{for} \quad d_{i,j} < R_i + R_j \end{cases}$$

The total repulsive forces exerted on the i-th particle by the other particles and the walls can be expressed as follows:

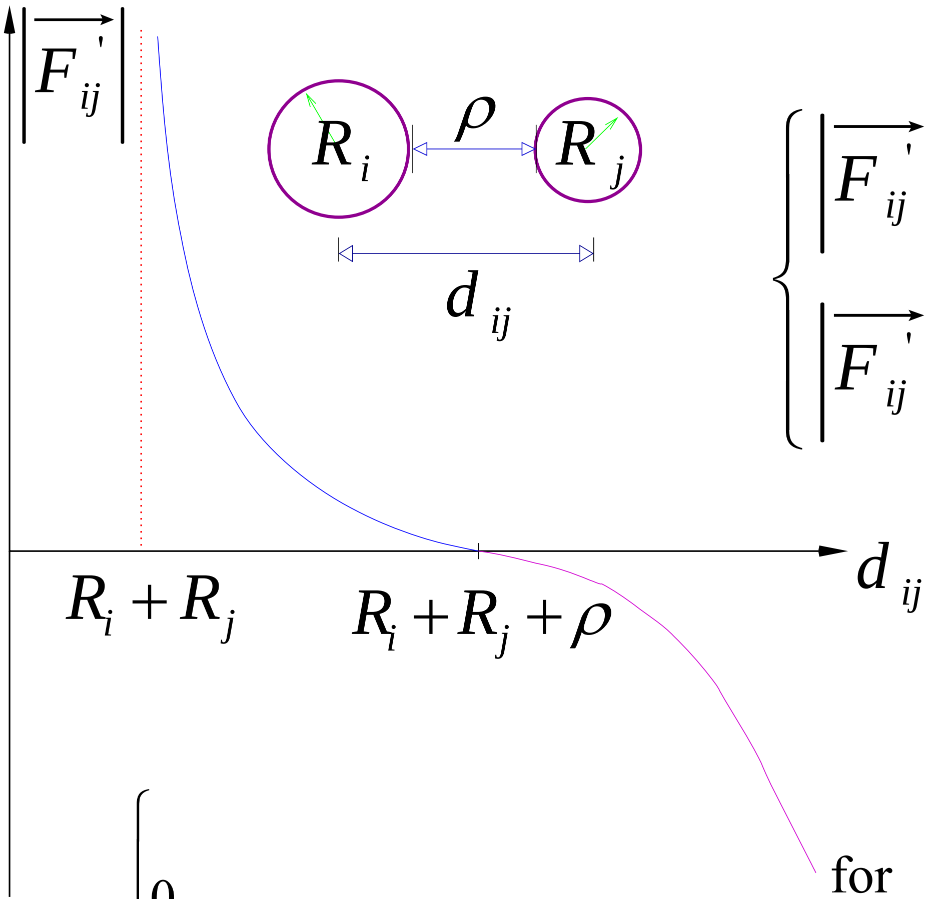
$$F_i^{'} = \sum_{j=1,\, j\neq i}^{N} F_{i,j}^{P} + F_i^{W}$$

# Examples



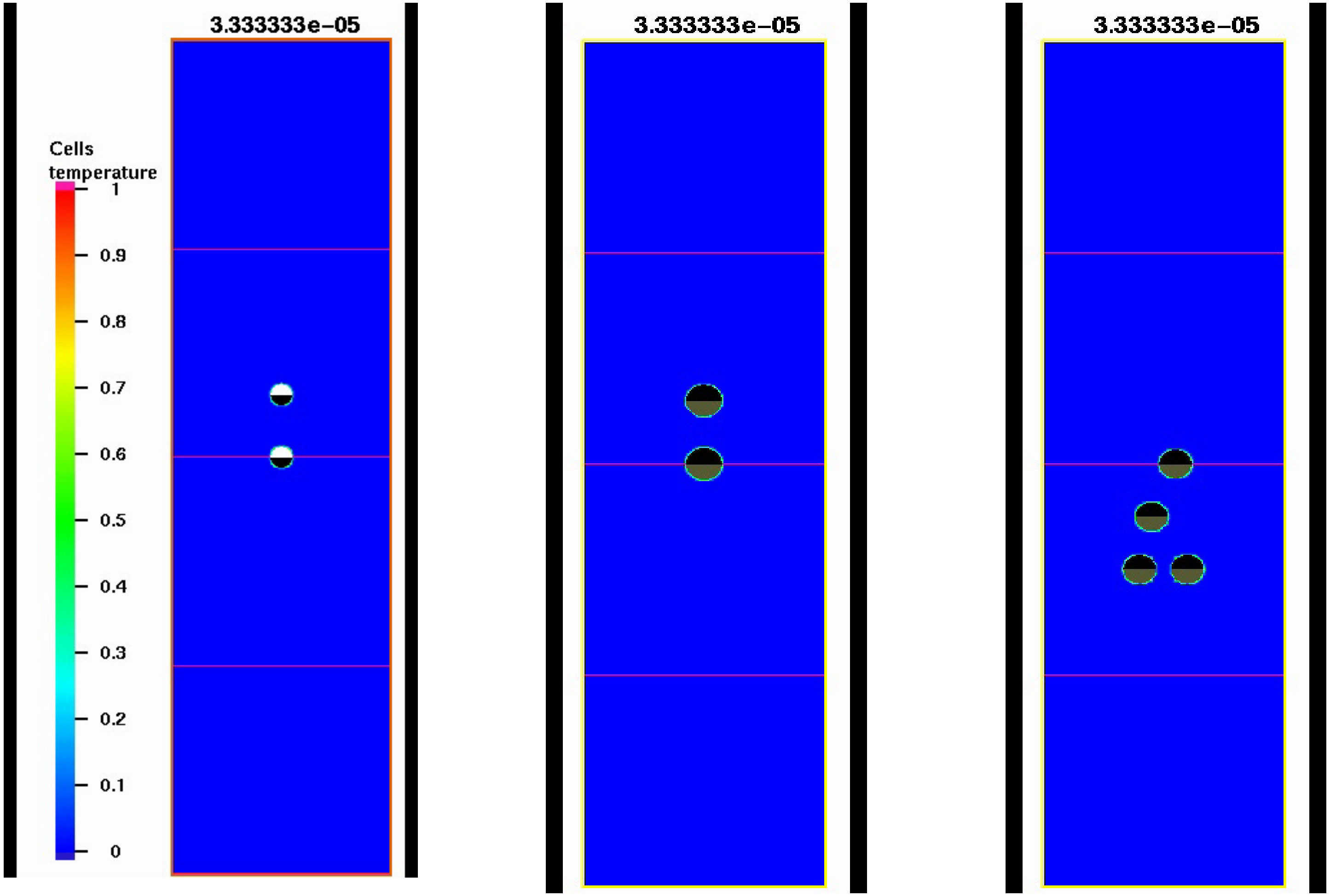
# Particle Agglomeration




$$\begin{cases} \left|\overrightarrow{F_{ij}^{'}}\right| \neq 0, & d_{ij} \geq R_i + R_j + \rho, \\ \left|\overrightarrow{F_{ij}^{'}}\right| = \dfrac{c_{ij}}{\varepsilon}, & d_{ij} = R_i + R_j. \end{cases}$$

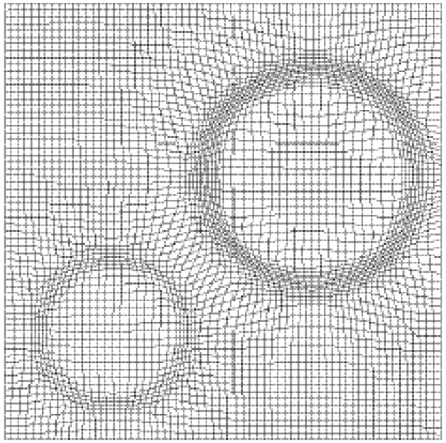
$$F_{ij}^{P} = \begin{cases} 0 & \text{for} \quad d_{i,j} > R_i + R_j + \rho \\ -\dfrac{1}{\varepsilon_P}\left(X_i - X_j\right)\left(R_i + R_j + \rho - d_{i,j}\right)^2 & \text{for} \quad R_i + R_j \leq d_{i,j} \leq R_i + R_j + \rho \\ \dfrac{1}{\varepsilon_P^{'}}\left(X_i - X_j\right)\left(R_i + R_j - d_{i,j}\right) & \text{for} \quad d_{i,j} < R_i + R_j \end{cases}$$

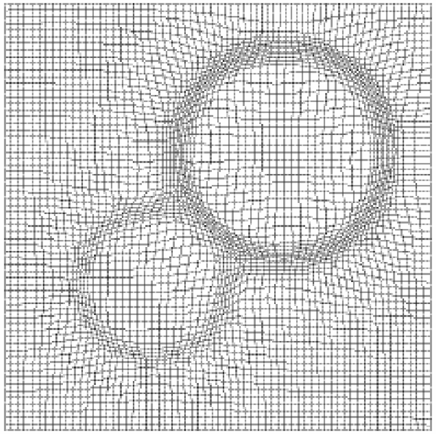
# Examples


3.333333e-05
Cells
temperature
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
3.333333e-05
3.333333e-05

# Challenges

- Adaptive time stepping + dynamical adaptive grid alignment/ALE





- (Better) collision models/Repulsive forces.
- Coupling with turbulence models.
- Modelling of Break-up/Coalescence phenomena.
- Deformable particles/fluid-structure interaction.
- Analysis of viscoelastic effects.
- Benchmarking and experimental validation for many particles.
- **Why tensorproduct-like meshes and r-adaptivity???.**

# Mathematics on Special Hardware

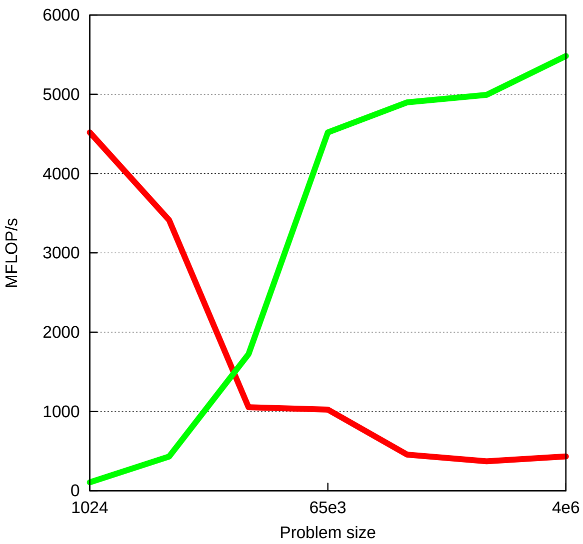
Sony PlayStation 3

Cell multicore processor, 7 synergetic processing units @ 3.2 GHz, **218 GFLOP/s**

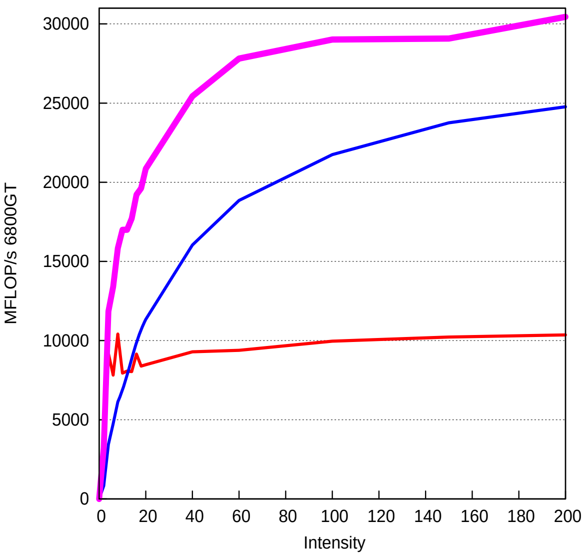
RSX graphics processor, 6+24 parallel pipelines, **1.8 TFLOP/s**

Memory interface @ 3.2 GHz

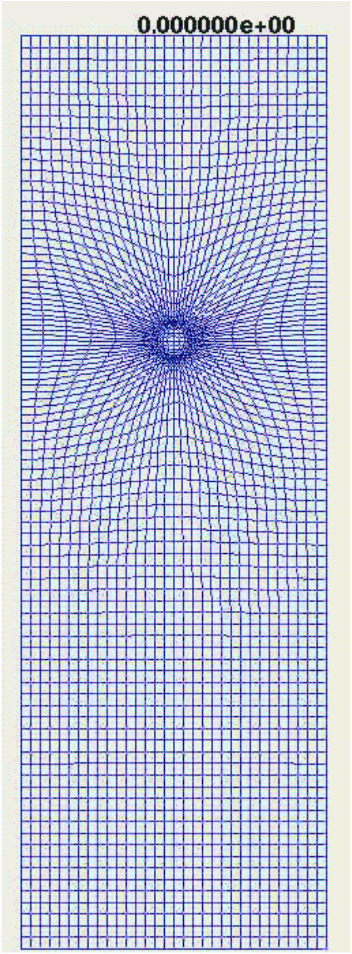
Typical performance for CPUs and GPUs:

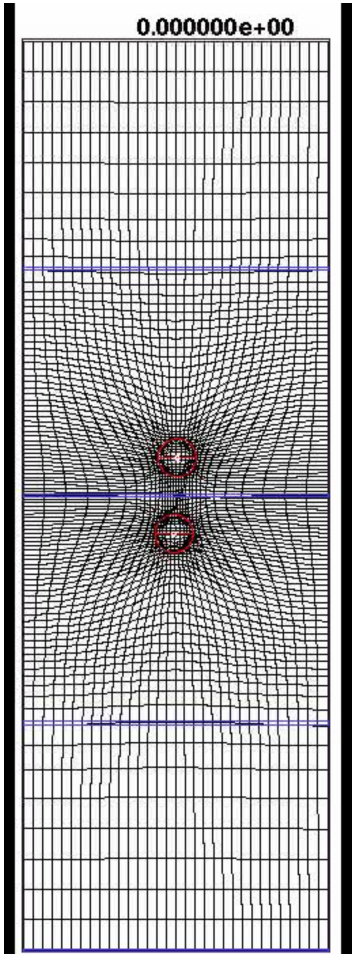



Goal: Improve performance by increasing numerical intensity:



# Numerical Examples



# Efficient Data Structures

$$L3 \approx 220.000 \; elements \quad \approx 1.100.000 \; d.o.f.s$$

$$L4 \approx 880.000 \; elements \quad \approx 4.400.000 \; d.o.f.s$$

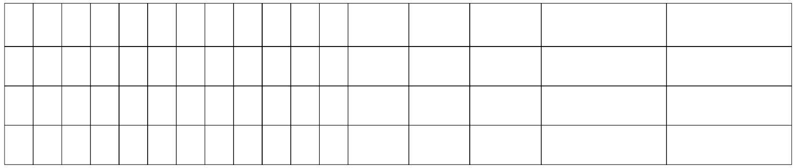
$$L5 \approx 3.530.000 \; elements \quad \approx 17.600.000 \; d.o.f.s$$

DEC/COMPAQ EV6, 833 MHz

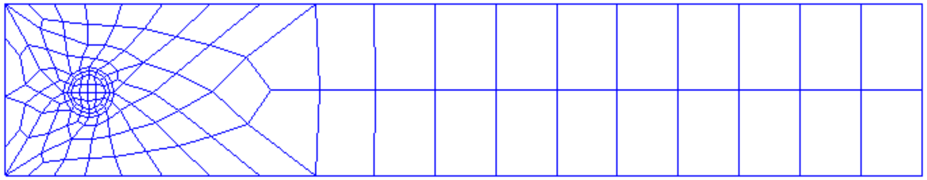
| CPU (s) | 'brute force' | | | | | | 'improved' | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| #PART | = 10 | | | = 1000 | | | = 10 | | | = 1000 | | |
| items | L=3 | L=4 | L=5 | L=3 | L=4 | L=5 | L=3 | L=4 | L=5 | L=3 | L=4 | L=5 |
| NSE | 17 | 88 | 440 | 16 | 80 | **403** | 17 | 95 | 423 | 17 | 83 | **435** |
| Force | 5 | 20 | 79 | 443 | 1771 | **7092** | 0 | 0 | 1 | 0 | 0 | **1** |
| Particle | 1 | 5 | 25 | 20 | 82 | **331** | 0 | 3 | 14 | 1 | 5 | **21** |
| Total | 24 | 114 | 546 | 480 | 1934 | **7827** | 18 | 98 | 439 | 18 | 89 | **468** |

Next: Efficient flow solver (for small $\Delta t$)???

# Evaluation of Force Calculations

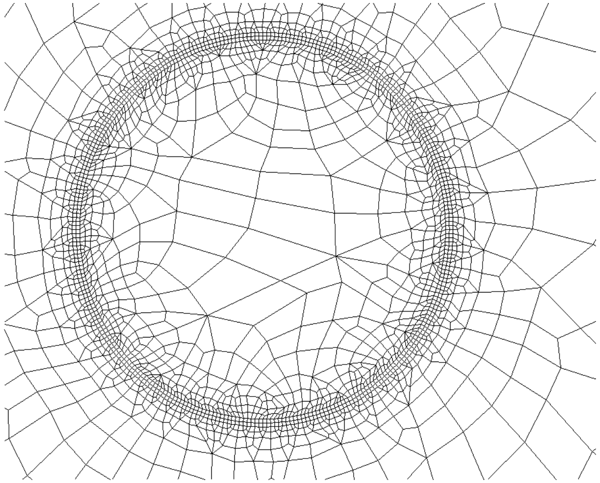


$LEVEL\ \ 6 \approx 280\ .000\ elements$



$LEVEL\ \ 6 \approx 150\ .000\ elements$

| LEVEL | ch. mesh I | ch. mesh II | ch. mesh I | ch. mesh II |
|---|---|---|---|---|
| 3 | 0.5529+01 | 0.5569+01 | 0.1216-01 | 0.2443-03 |
| 4 | 0.5353+01 | 0.5575+01 | 0.1074-01 | 0.0014-01 |
| 5 | 0.5427+01 | 0.5572+01 | 0.6145-02 | 0.0812-01 |
| 6 | 0.5501+01 | 0.5578+01 | 0.9902-02 | 0.1020-01 |
| | $C_d$ = 0.55795+01 | | $C_l$ = 0.10618-01 | |



| LEVEL | $C_d$ | $C_l$ |
|---|---|---|
| 2 | 0.55201+01 | 0.1057-01 |
| 3 | 0.55759+01 | 0.1036-01 |
| 4 | 0.55805+01 | 0.1041-01 |

$LEVEL\ \ 4 \approx 150\ .000\ elements$

# Complete Algorithm

**The complete algorithm $(t_n \to t_{n+1})$ for the coupled fluid-solid system can be summarized as follows:**

1. Given the position and velocity of the particles at time $t_n$
2. Set the fictitious boundary and its boundary condition for the fluid.
3. Solve the fluid equations to get the fluid velocity and the pressure.
4. Calculate the hydrodynamic forces acting on every particle.
5. Calculate the motion of the solid particles.
6. Check if the collision happens and calculate collision forces.
7. Update the particle position and velocity by the collision forces.
8. Return to the first step $(n \to n+1)$ and advance to the next time step.

# Induced Rotation of an Airfoil Wing