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

Stefan Turek, Raphael Münster

with support by Dan Anca, Otto Mierka, Kamran Usman and Decheng Wan

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

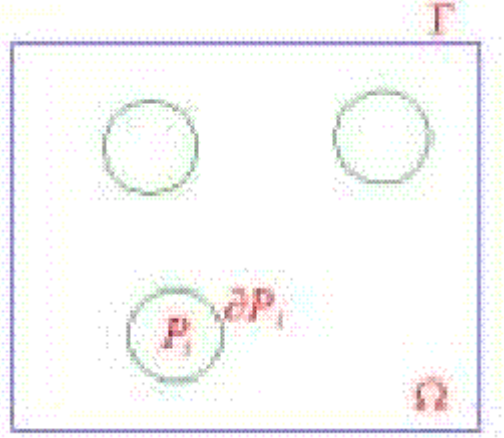
- Model for particulate flow
- Numerical techniques
- Examples
- Further improvements

Fluid – (Rigid) Solid Interfaces

technische universität dortmund

Consider the flow of N solid particles in a fluid with density ρ and viscosity μ.

Denote by $\Omega_f(t)$ the domain occupied by the fluid at time t, by $\Omega_P(t)$ the domain occupied by the particle at time t and let $\Omega = \Omega_f \cup \Omega_P$.



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

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

where σ 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** ω_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;

ΔM_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 **hydrodynamic 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**.

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 Γ_p.

No slip boundary conditions at interface Γ_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** θ_p are obtained by integration of the kinematic equations:

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

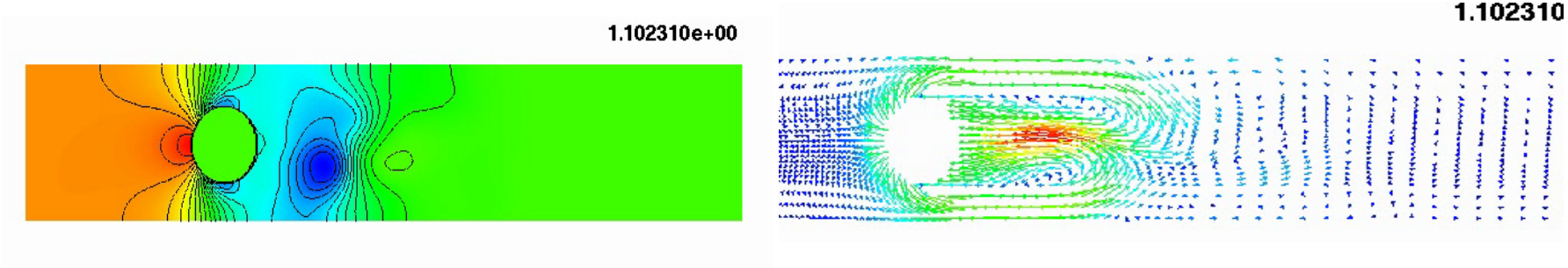
Explicit coupling

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

Eulerian approach: fixed meshes possible!

Use a (adapted) mesh that covers the whole domain where the fluid may be present.

FEM-Fictitious Boundary Methods (FBM)



Computational mesh (can be) independent of 'internal objects'

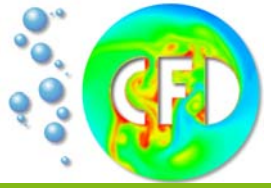
Hydrodynamic forces and torque acting on the i-th particle

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

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

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

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



Define auxiliary function α 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.

$$n_p = \nabla \alpha_p$$

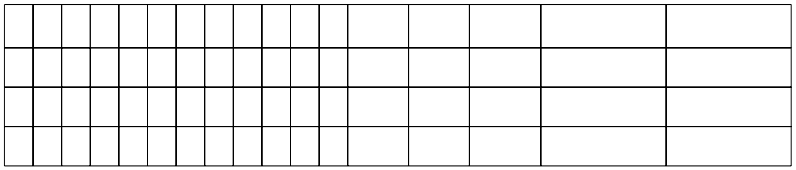
Force acting on the wall surface of the particles can be computed by

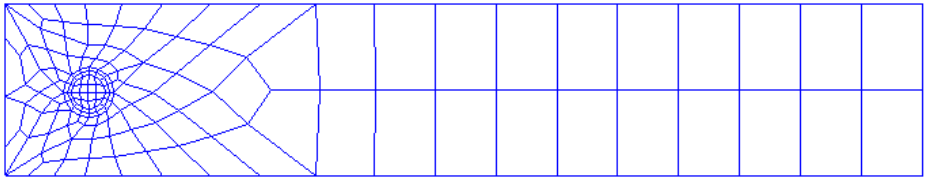
$$F_p = -\int_{\Gamma_p} \sigma \cdot n_p d\Gamma_p = -\int_{\Omega_T} \sigma \cdot \nabla \alpha_p d\Omega_T$$

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

Validation of Force Calculations

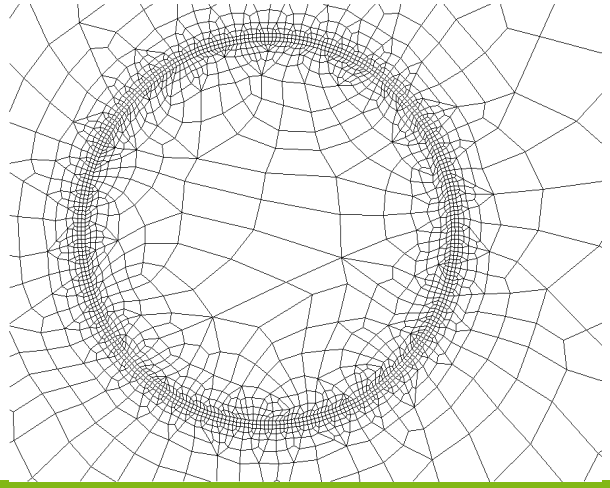
technische universität dortmund





LEVEL 6 ≈ 280 .000 *elements* *LEVEL* 6 ≈ 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

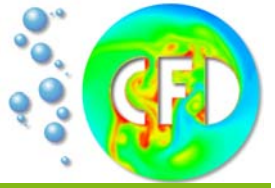
LEVEL 4 ≈ 150 .000 *elements*

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 (!)



Grid Deformation Method

technische universität dortmund

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

$\Rightarrow$ local mesh area $\approx f$

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

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

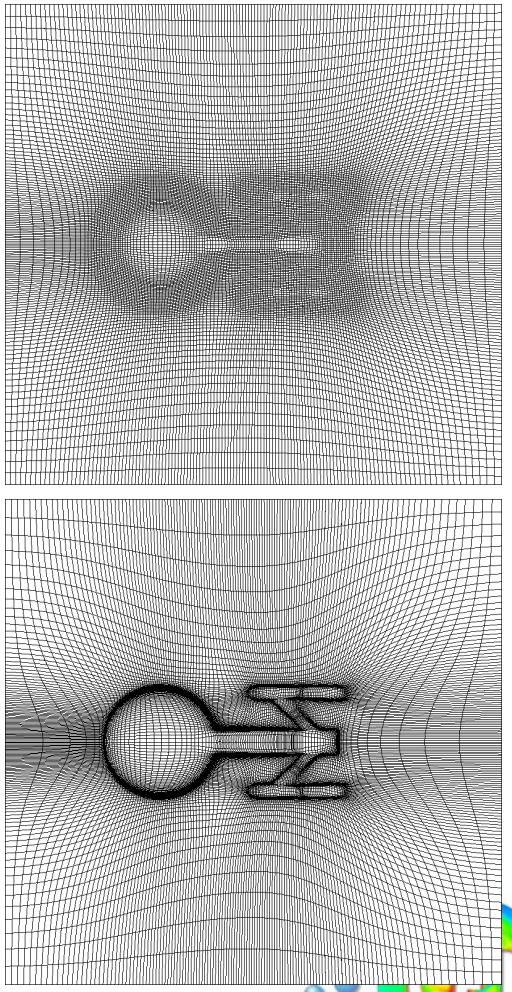
2. Solve ($t \in [0,1]$)

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

3. Solve the ODE system

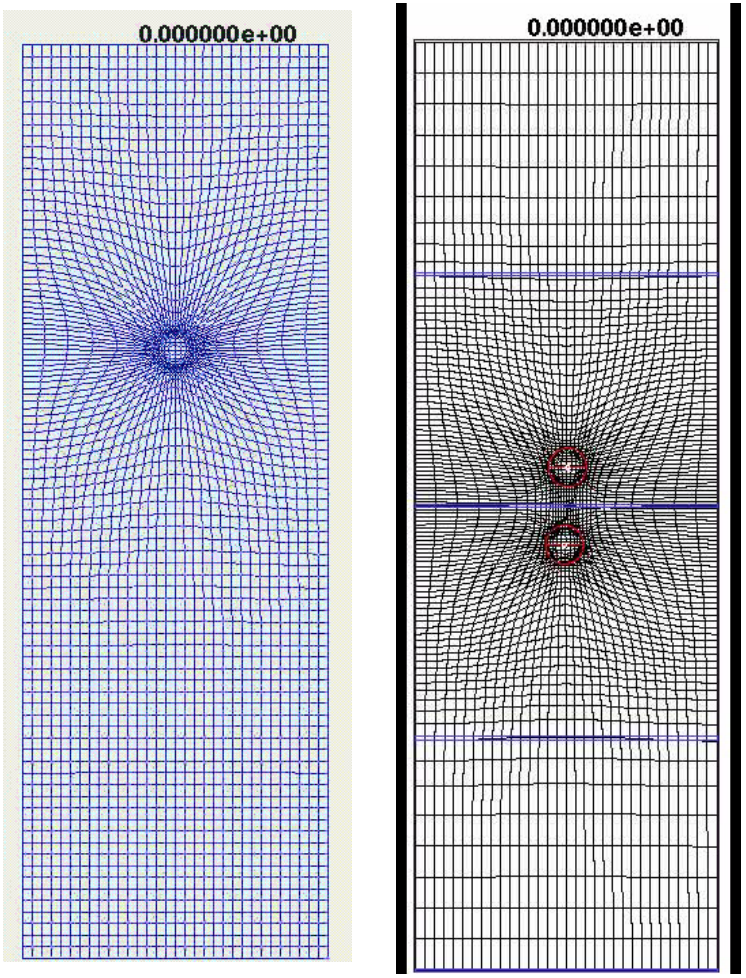
$$\frac{\partial}{\partial t}\phi(\xi,t) = f(\phi(\xi,t),t)\nabla v(\phi(\xi,t),t)$$

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



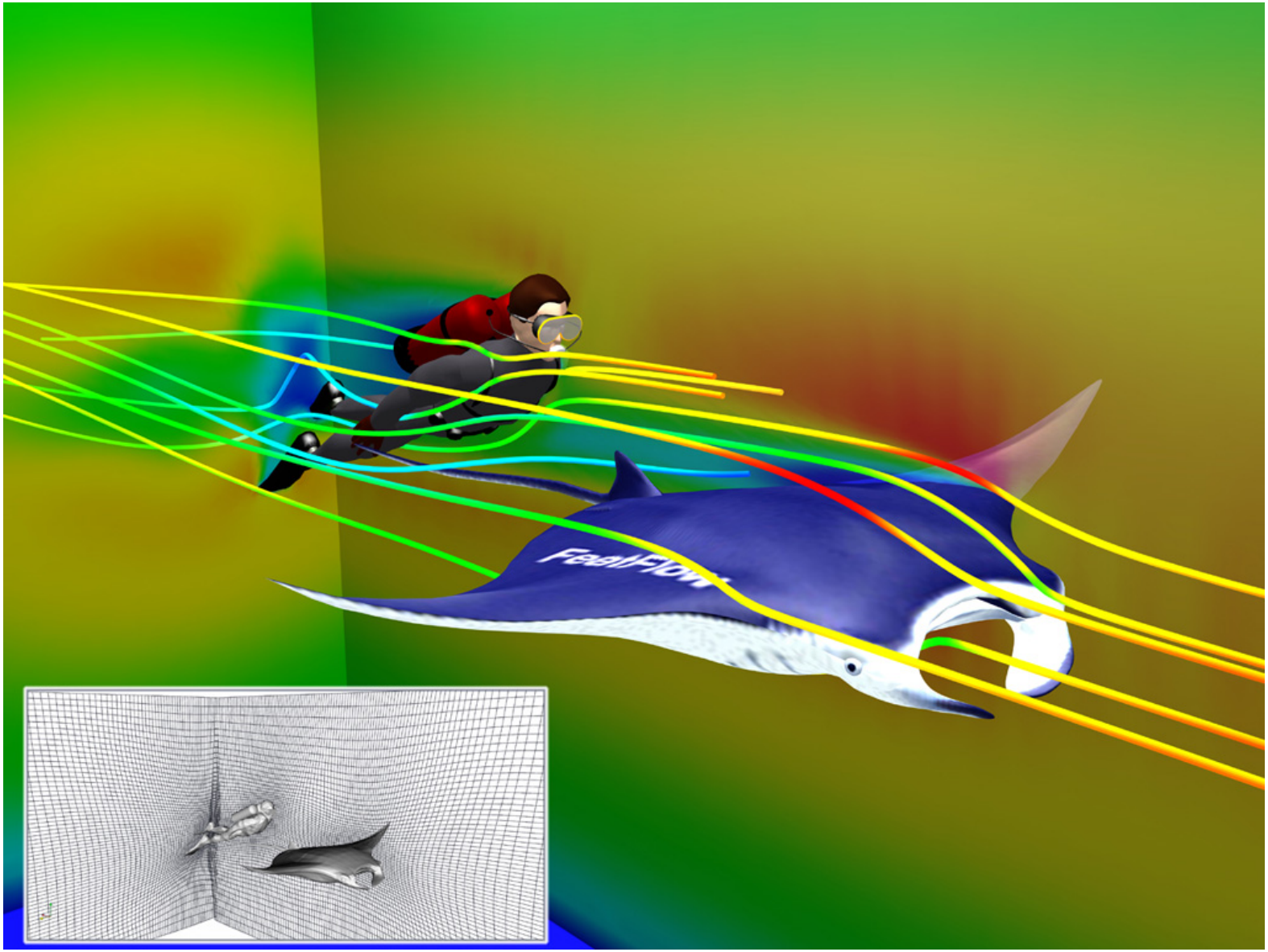
Grid deformation preserves the (local) logical structure of the grid

Numerical Examples



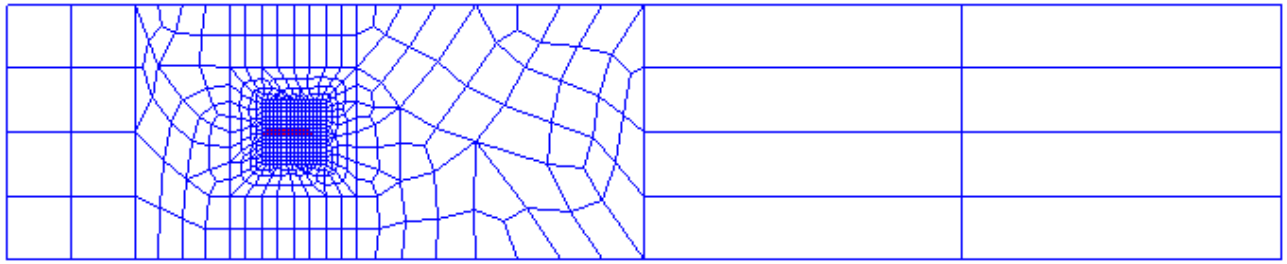
Generalized Tensorproduct Meshes

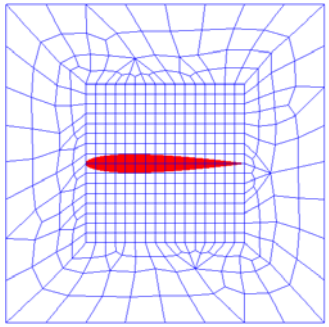
technische universität dortmund

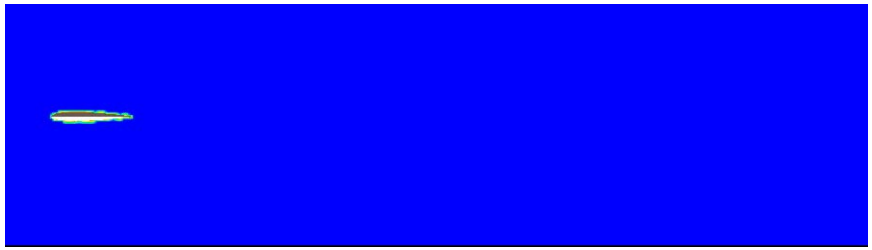


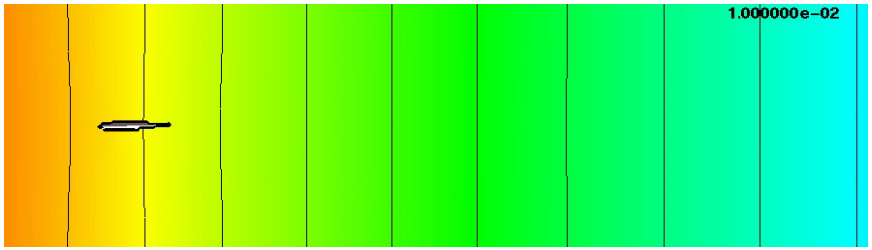
technische universität dortmund

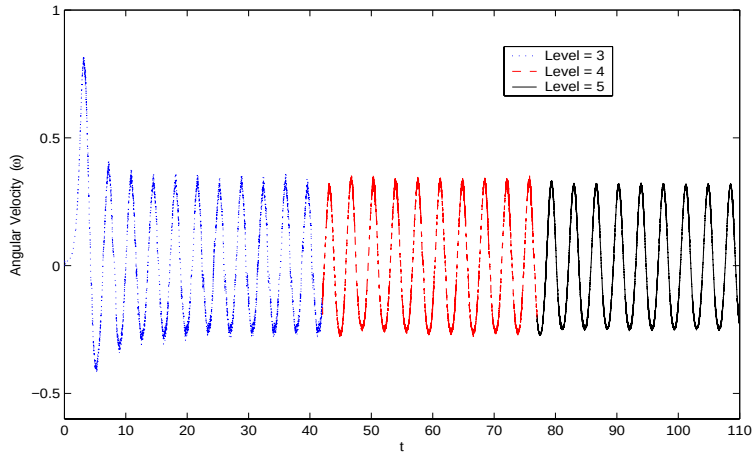
‘Viscous flow around a moving airfoil’ (Glowinski)

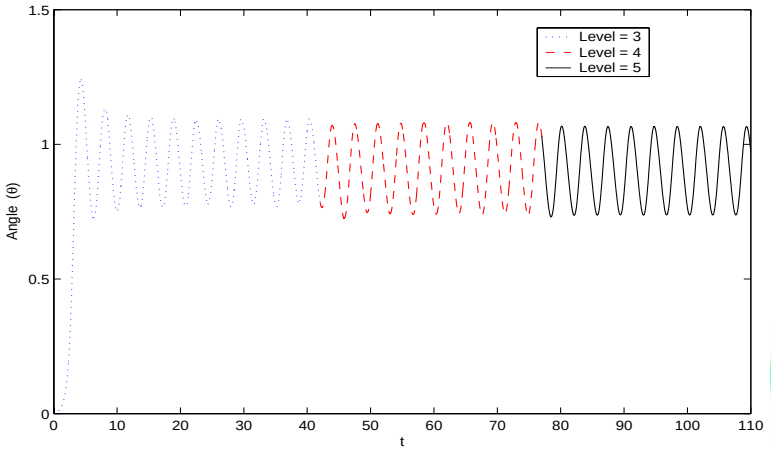




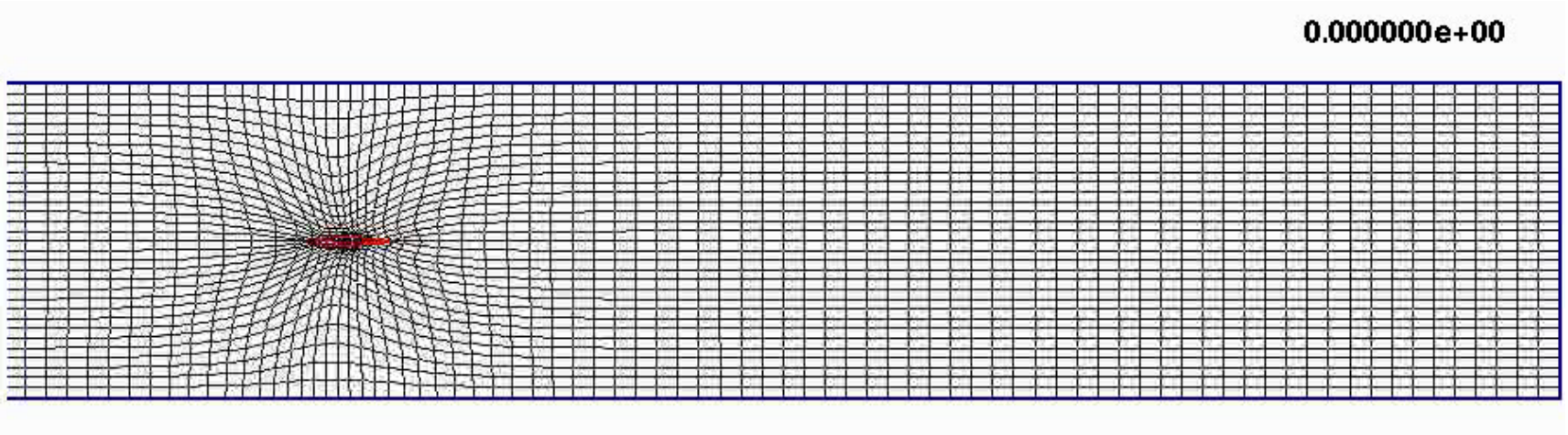


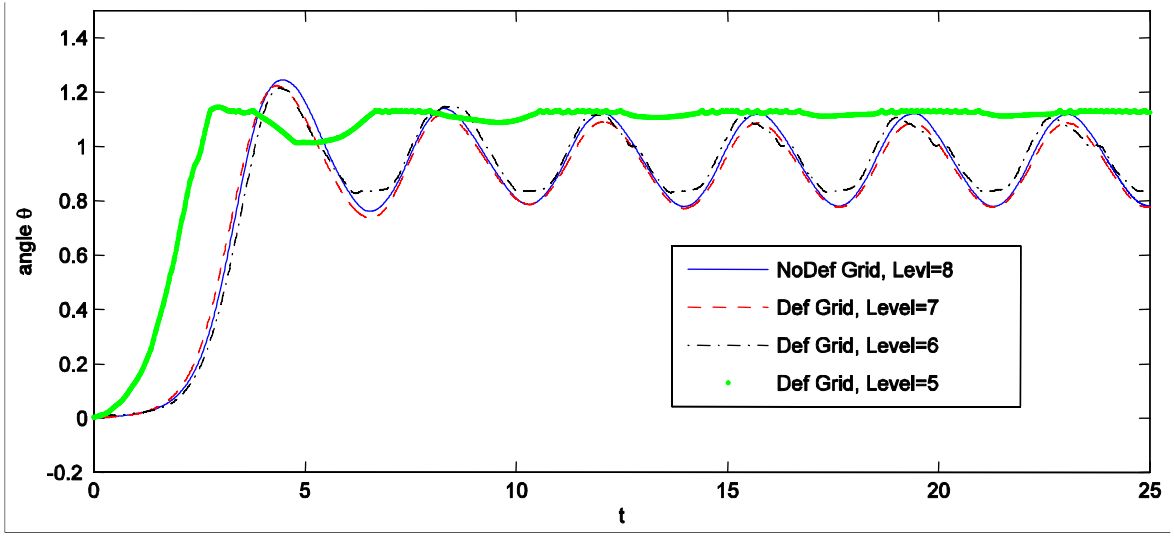




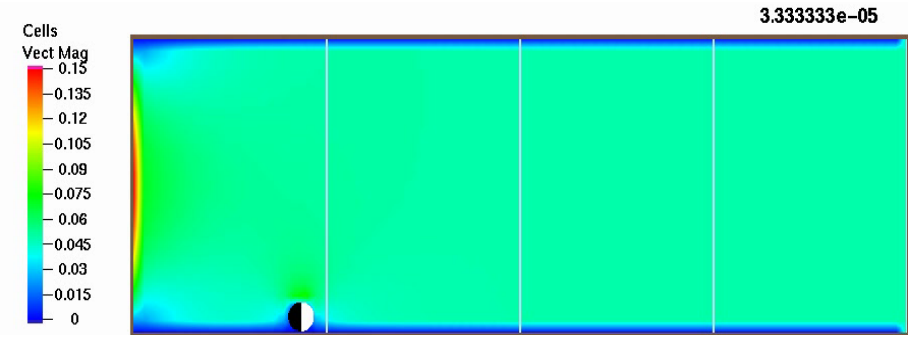


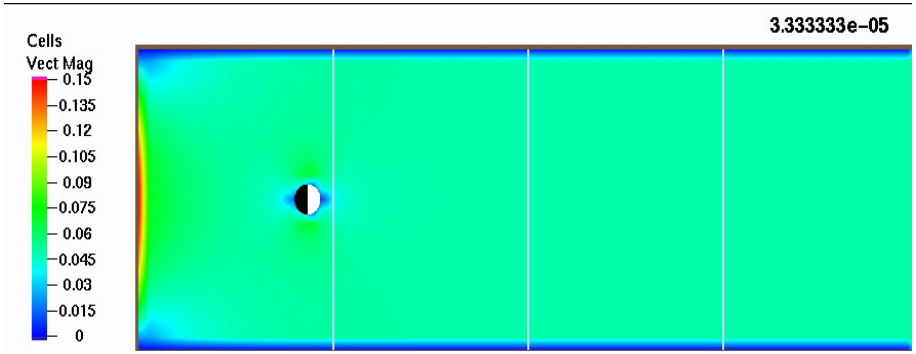
Rotation of an Airfoil Wing





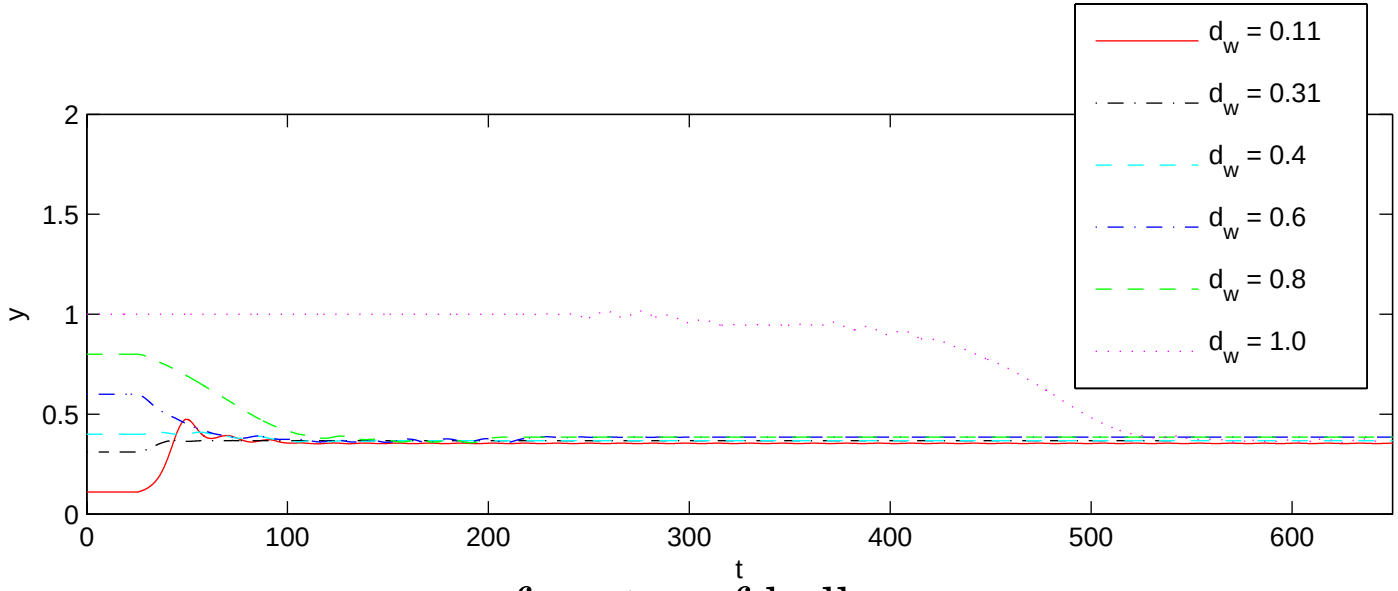
Lift-Off for Circle





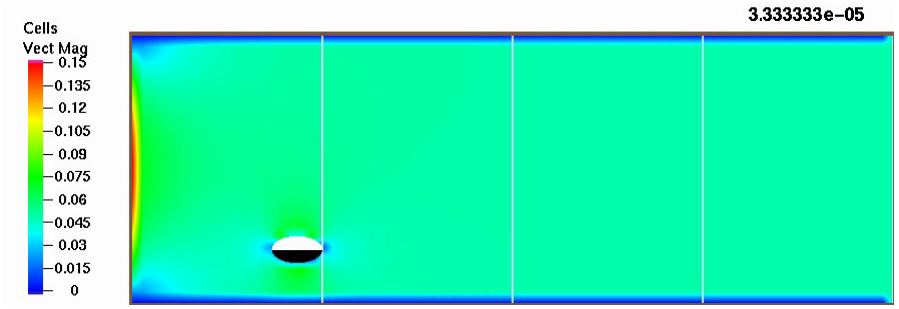
Velocity $(d_w = 0.1)$

Velocity $(d_w = 1.0)$

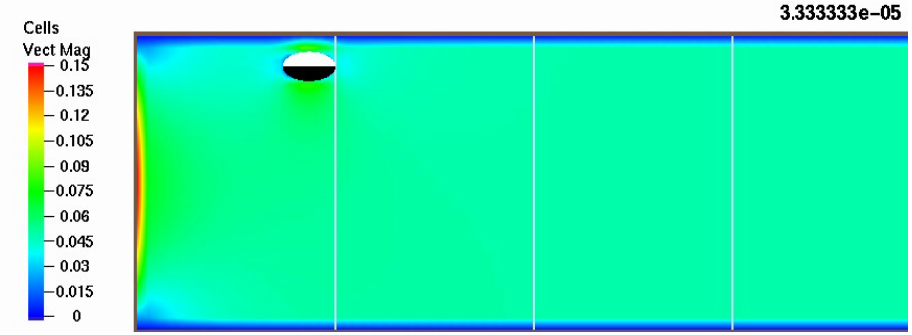


y of center of ball

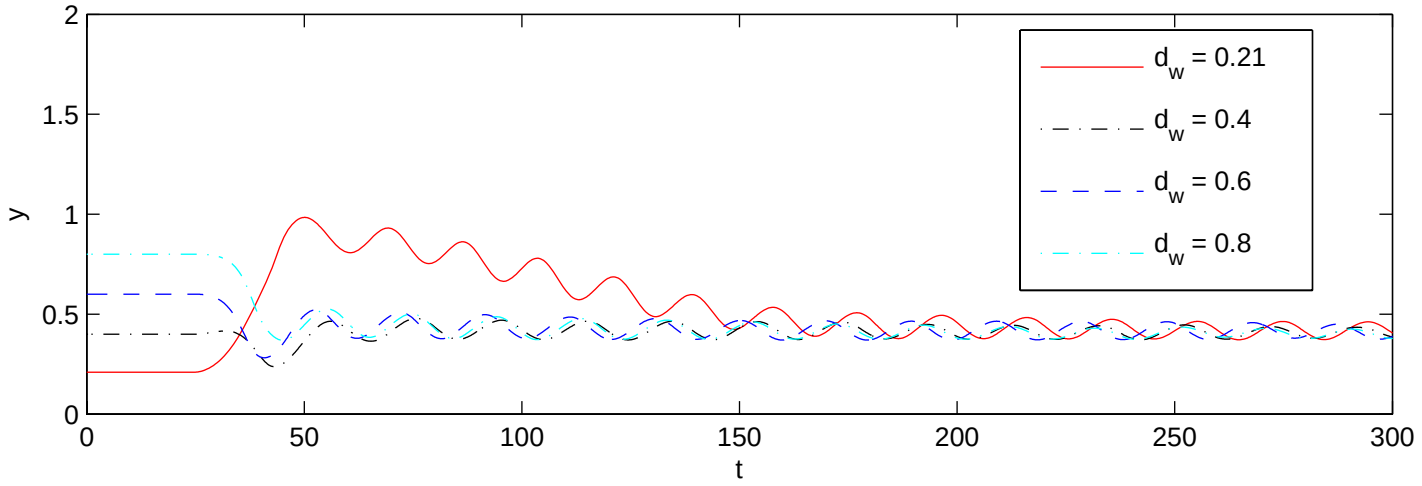
Lift-Off for Ellipse



Velocity $(d_w = 0.4)$



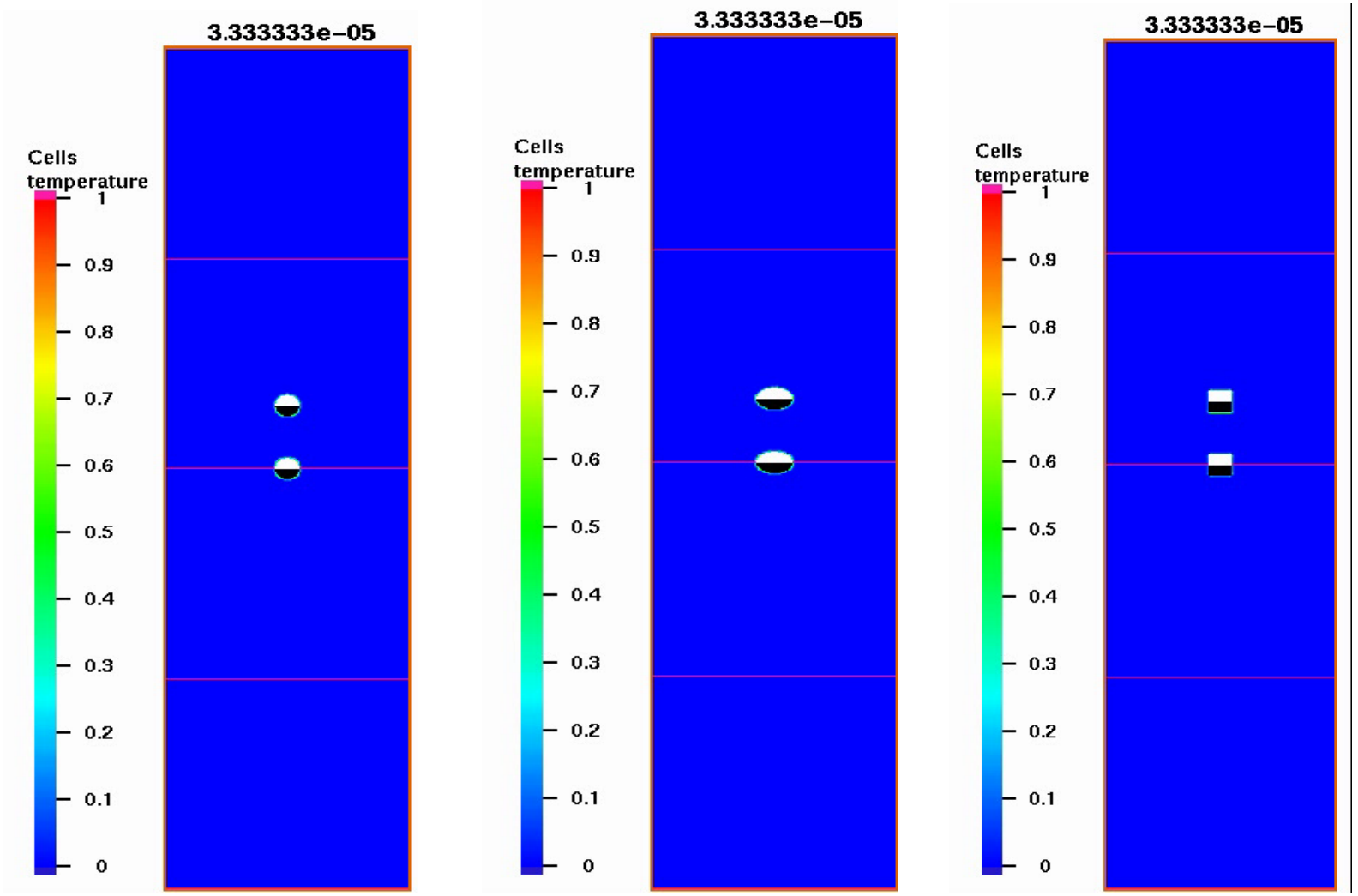
Velocity $(d_w = 1.8)$



y of center of ellipse

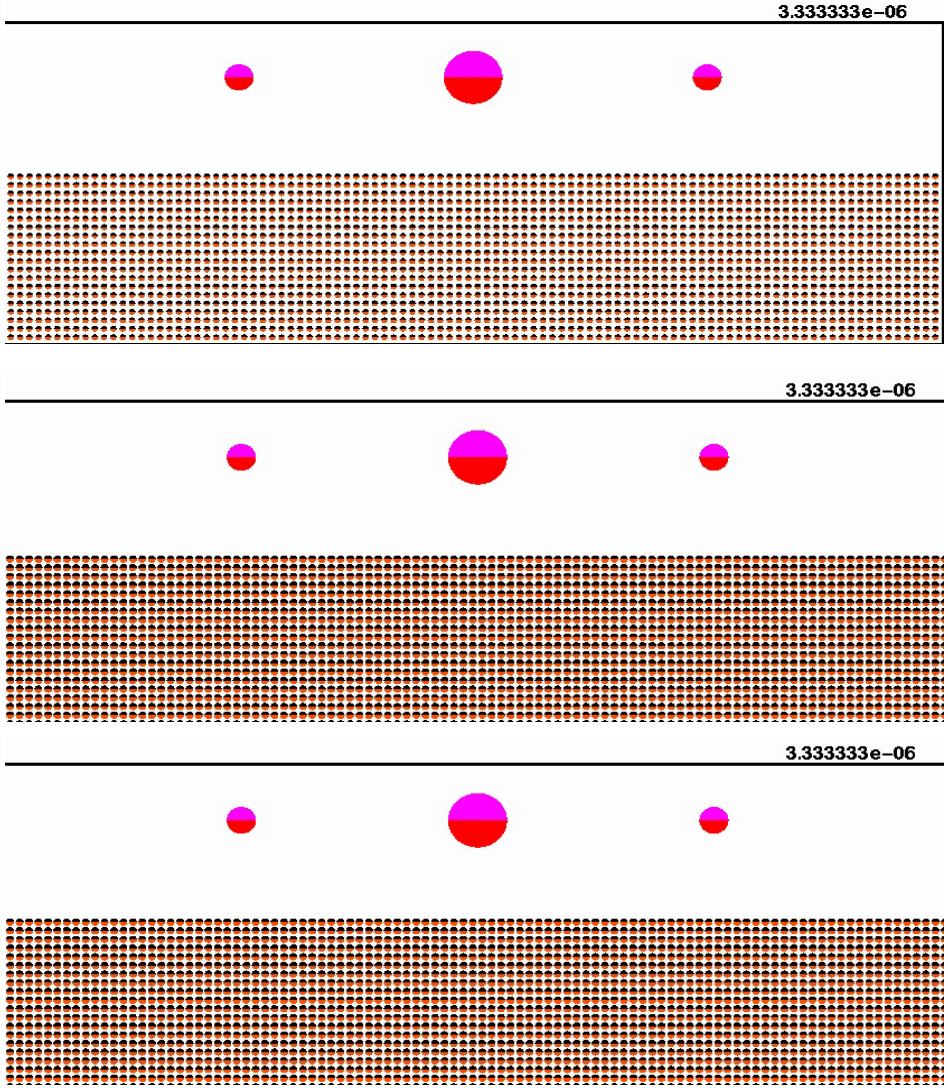
technische universität dortmund

‘Kissing, Drafting, Thumbling’



technische universität dortmund

‘Impact of heavy balls on 2000 small particles’



$\rho_f = 1$

$\rho_{bd} = 2$

$\rho_{sp} = 1.1$

$\rho_f = 1$

$\rho_{bd} = 2$

$\rho_{sp} = 2$

$\rho_f = 1$

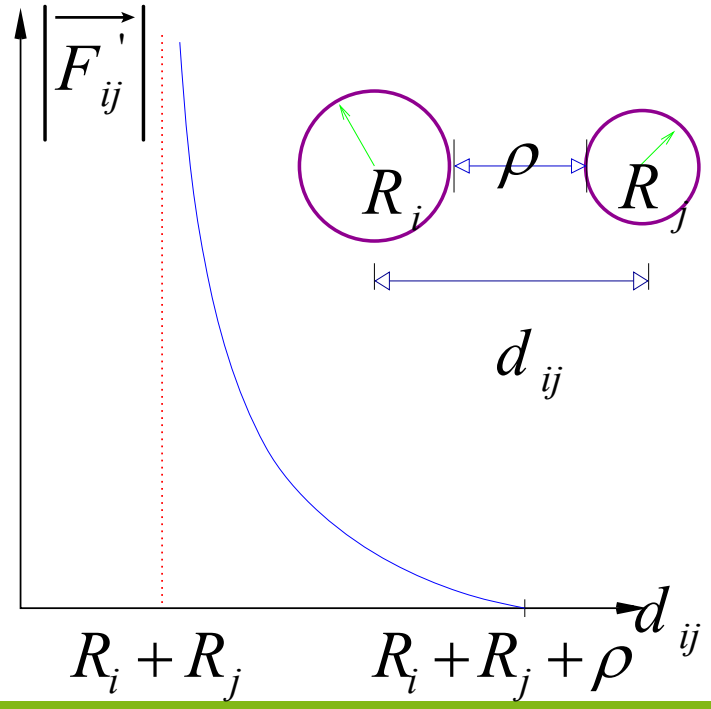
$\rho_{bd} = 2$

$\rho_{sp} = 20$

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:

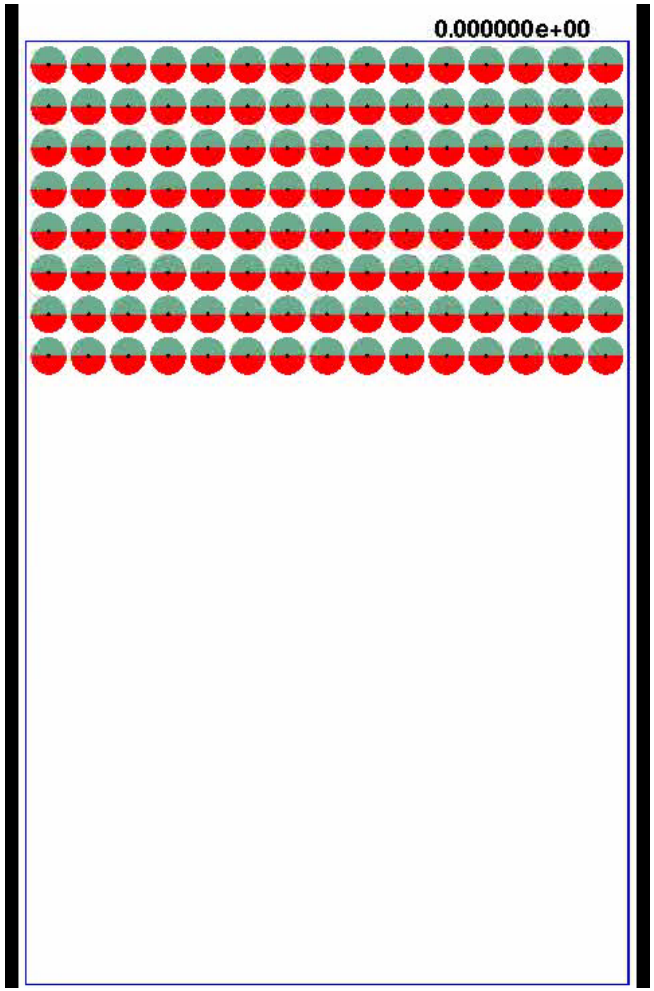
$$F_{ij}^{P} = \begin{cases} 0 & \text{for} \quad d_{i,j} > R_i + R_j + \rho \\ \frac{1}{\varepsilon_P}\left(X_i - X_j\right)\left(R_i + R_j + \rho - d_{i,j}\right)^2 & \text{for} \quad R_i + R_j \le d_{i,j} \le R_i + R_j + \rho \\ \frac{1}{\varepsilon_P'}\left(X_i - X_j\right)\left(R_i + R_j - d_{i,j}\right)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

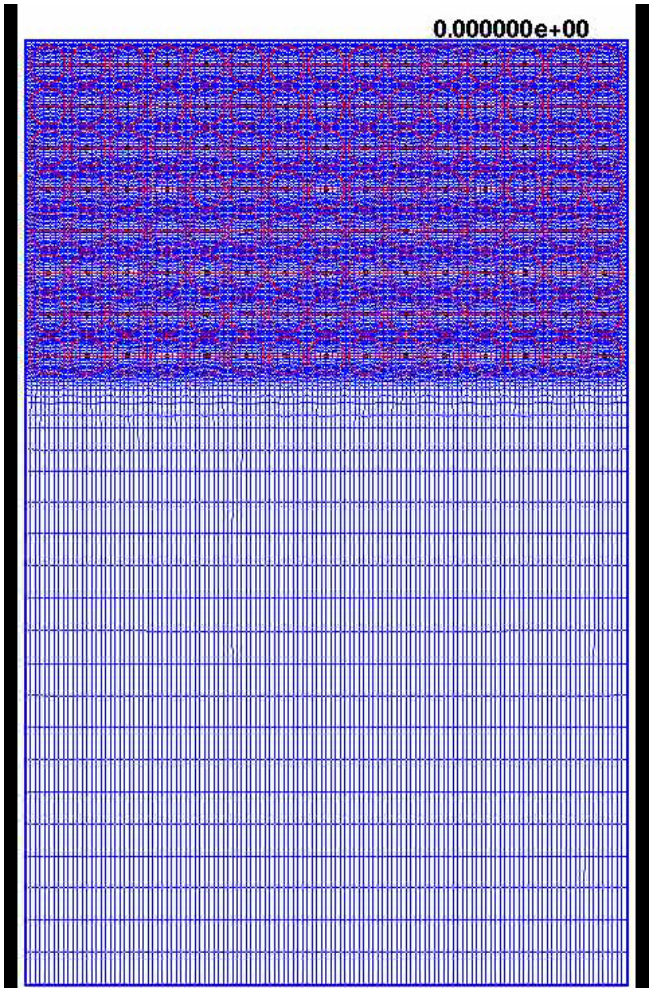
The total repulsive forces exerted on the i-th particle by the other particles and the walls can be expressed as follows:

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

Examples

technische universität dortmund





Efficient Data Structures

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

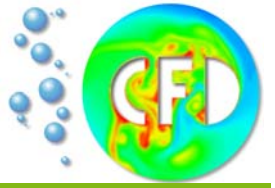
$$L4 \approx 880.000\ elements \approx 4.400.000\ d.o.f.s$$

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

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**

Required: Efficient flow solver (for small Δt)???



A Many Body Lubrication Model

In this model, proposed by *Maury*, the lubrication forces for particle – particle collisions (particle – wall collisions) are estimated by :

$$F_i^j = -F_j^i = -\kappa(D_{ij})[(\dot{x}_i - \dot{x}_j)\cdot e_{ij}]e_{ij} - \kappa^{\perp}(D_{ij})[(\dot{x}_i - \dot{x}_j)\cdot e_{ij}^{\perp}]e_{ij}^{\perp},$$

κ and $\kappa^{\perp}$ vanish when the distance D_{ij} between two bodies is greater than a given value $d_\circ$, which is taken equivalent to the size of the body.

$$\kappa(d) = \mu(1/d), \qquad \kappa^{\perp}(d) = \mu^{\perp}\ln(d_\circ/d);$$

x_i is the mass centre of the i^{th} body;

e_{ij} is the unit vector along $x_i x_j$ and $e_{ij}^{\perp}$ is perpendicular to e_{ij}.

Particle motion:

$$m_i x_i^{''} = \Phi_i + \sum_{j \neq i} F_{i,j}\left(x_{i,j}^{'}, x_{j,i}^{'}\right),$$

$$I_i \theta^{''} = \sum_{j \neq i} r_i \times F_{i,j}\left(x_{i,j}^{'}, x_{j,i}^{'}\right).$$

Φ_i is the body force.

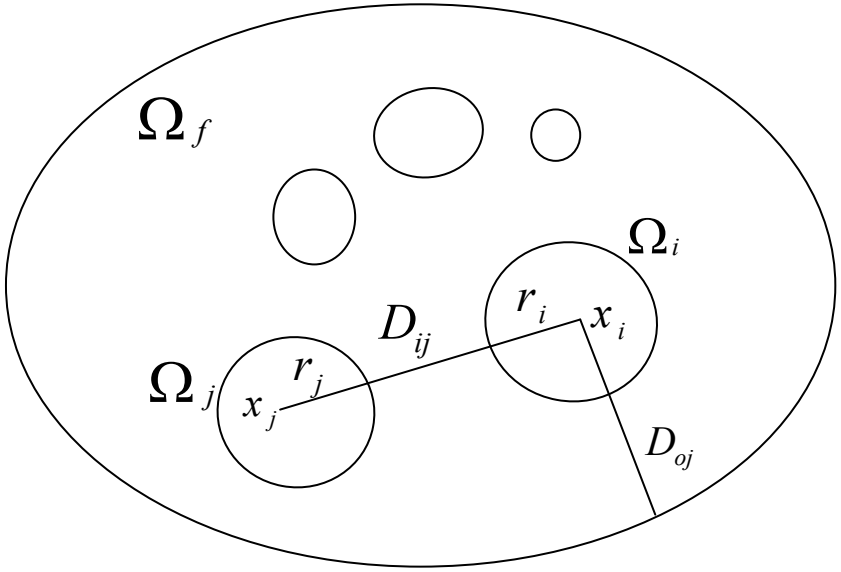
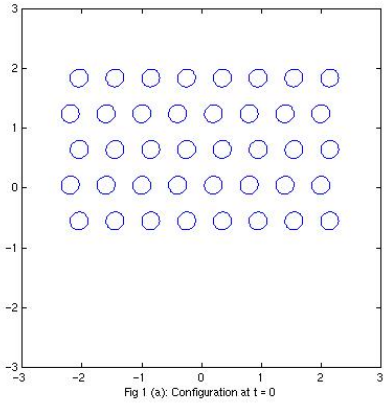
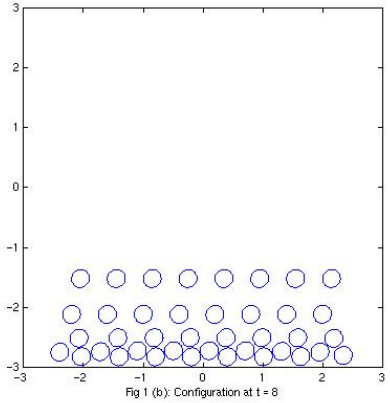
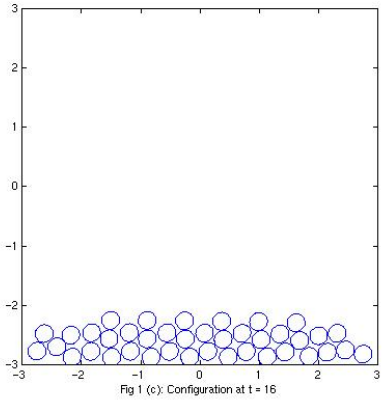


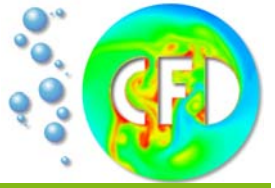
Fig. 1. Particle-Particle and Particle-Wall collision

Numerical example:

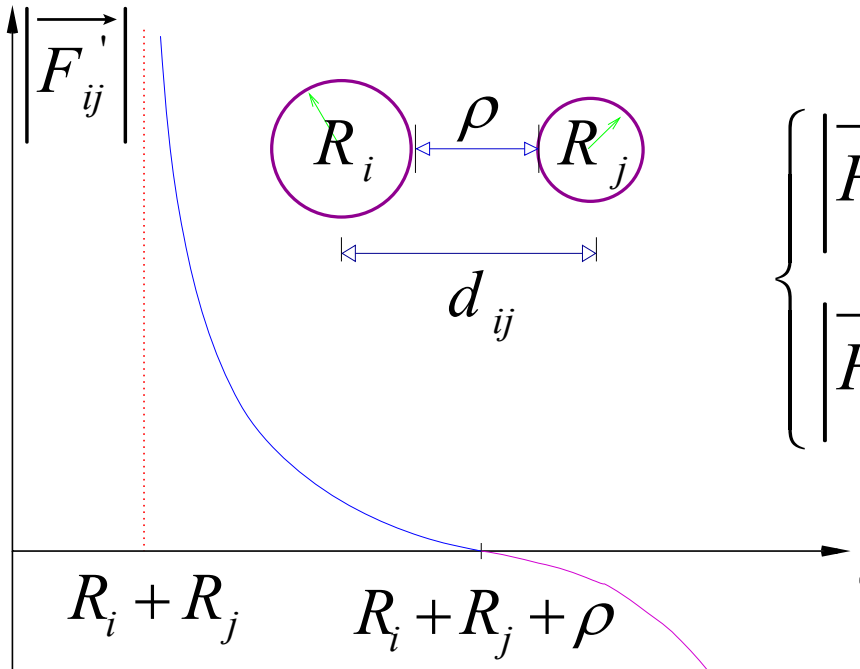








Particle Agglomeration

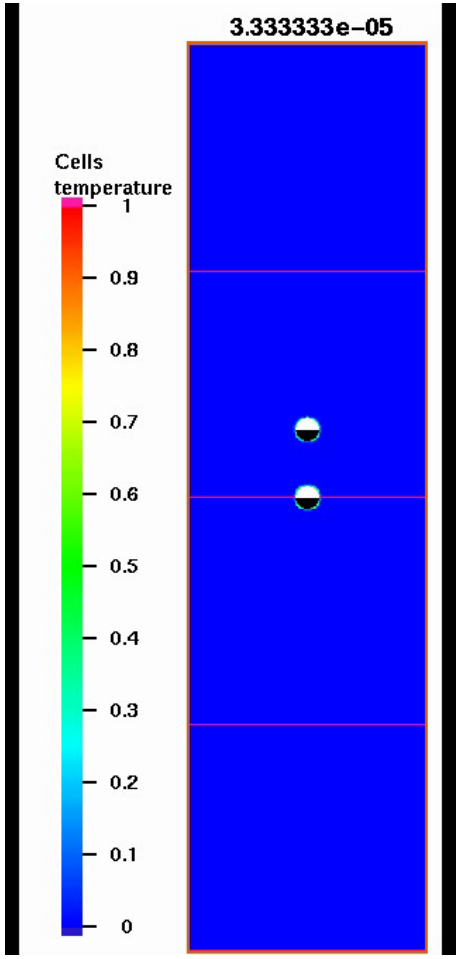


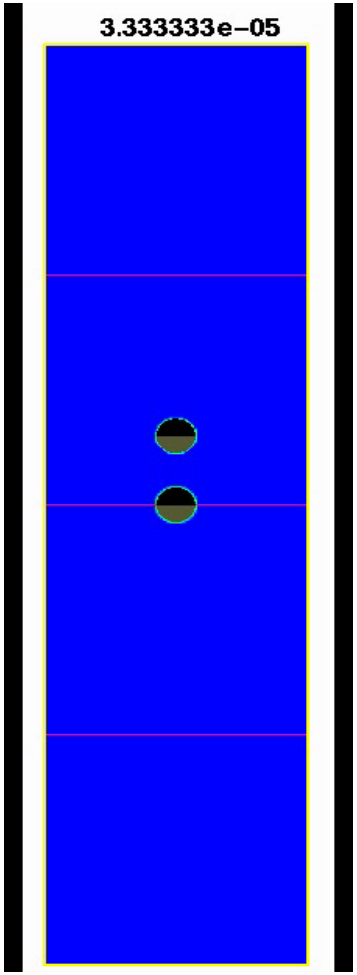
$$\begin{cases} \left|\overrightarrow{F_{ij}}'\right| \neq 0, & d_{ij} \geq R_i + R_j + \rho, \\ \left|\overrightarrow{F_{ij}}'\right| = \dfrac{c_{ij}}{\varepsilon}, & d_{ij} = R_i + R_j. \end{cases}$$

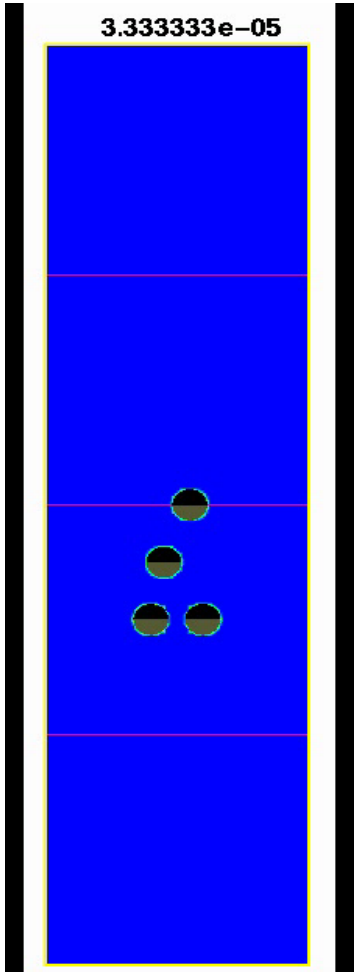
$$F_{ij}^P = \begin{cases} 0 & \text{for} \quad d_{i,j} > R_i + R_j + \rho \\ -\dfrac{1}{\varepsilon_P}\left(X_i - X_j\right)\left(R_i + R_j + \rho - d_{i,j}\right)^2 & \text{for} \quad R_i + R_j \leq d_{i,j} \leq R_i + R_j + \rho \\ \dfrac{1}{\varepsilon_P'}\left(X_i - X_j\right)\left(R_i + R_j - d_{i,j}\right)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

Examples

technische universität dortmund

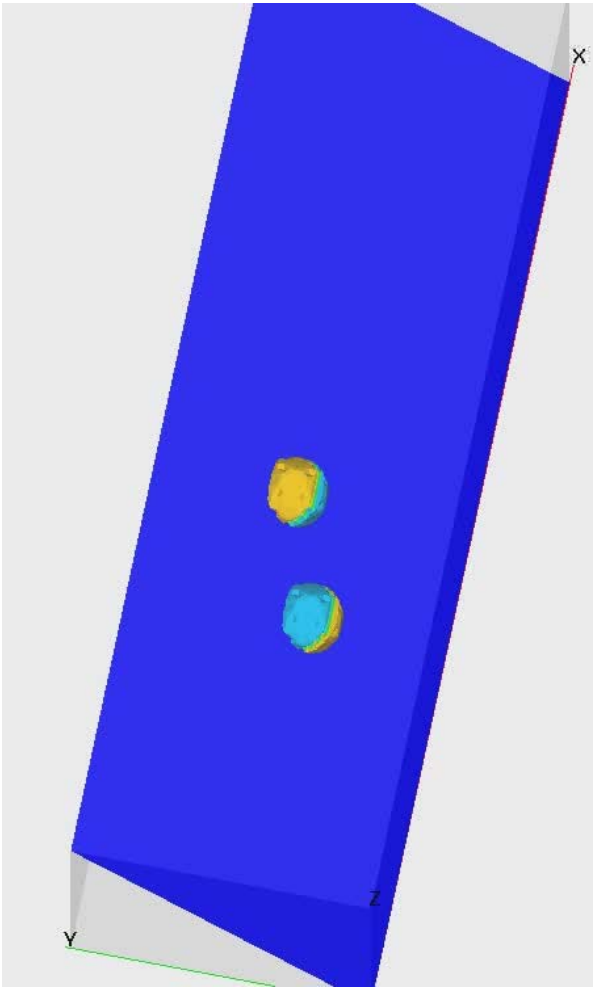


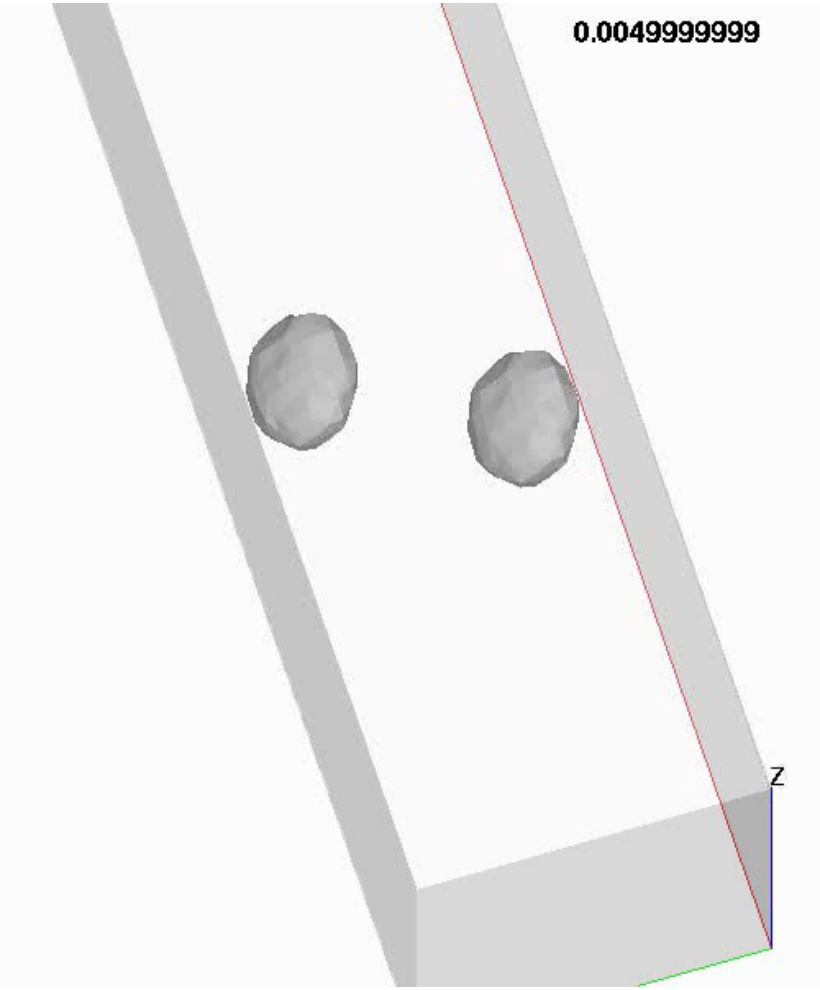




3D Examples

technische universität dortmund

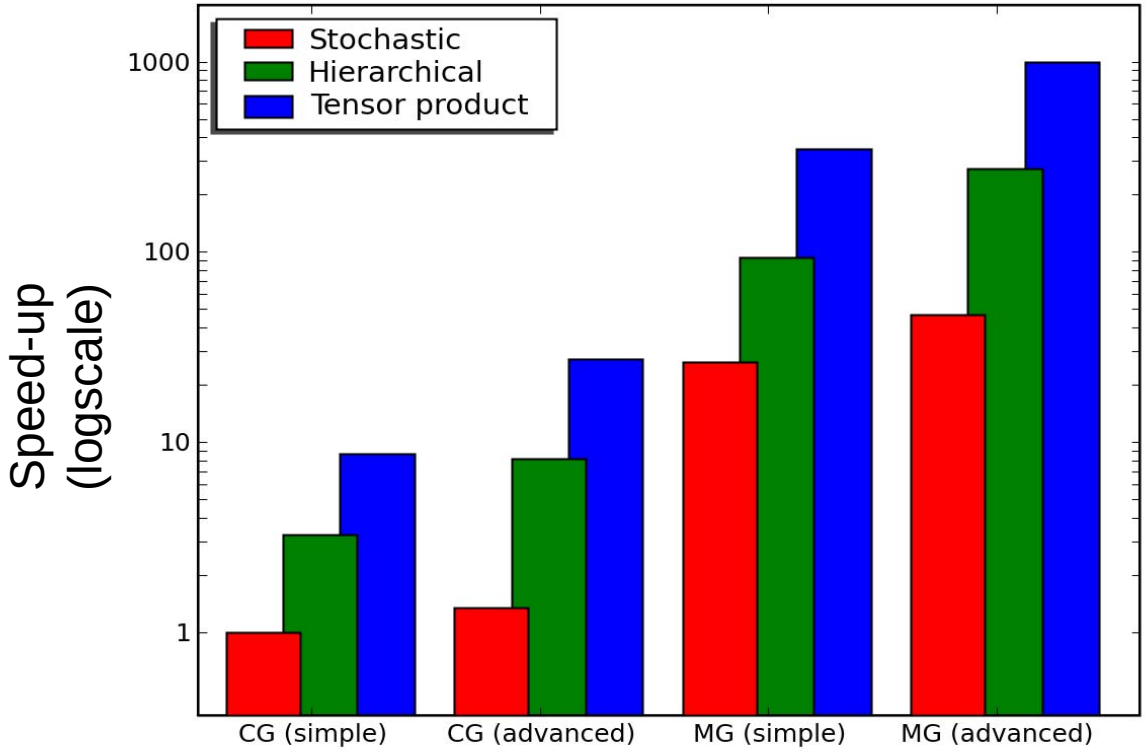




Challenges

- Adaptive time stepping + dynamical adaptive grid alignment/ALE
- Coupling with turbulence models.
- Deformable particles/fluid-structure interaction.
- Analysis of viscoelastic effects.
- Benchmarking and experimental validation for many particles.
- **Why tensorproduct-like meshes and r-adaptivity???.**

FEM for 8 Mill. unknowns on general domain, 1 CPU, Poisson Problem in 2D



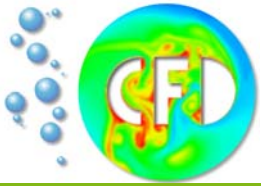
Dramatic improvement (factor 1000) due to **better Numerics** AND **better data structures/ algorithms** on 1 CPU

Numerics on special hardware

- CELL multicore processor (PS3),
  7 synergistic processing units
  @ 3.2 GHz, 218 GFLOP/s,
  Memory @ 3.2 GHz

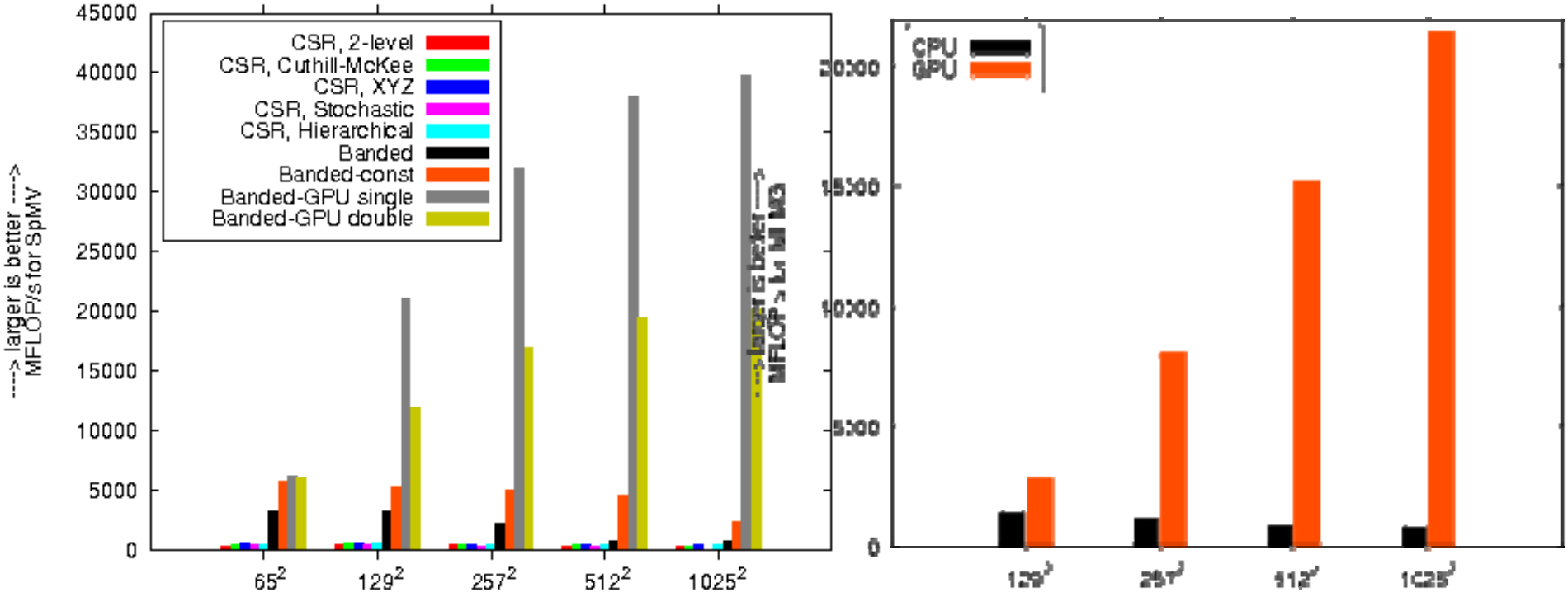
- GPU (NVIDIA GTX 285):
  240 cores @ 1.476 GHz,
  1.242 GHz memory bus (160 GB/s)
  ≈ 1.06 TFLOP/s

UnConventional High Performance Computing (UCHPC)



Example: Sparse MV on TP Grid

technische universität dortmund



40 GFLOP/s, 140 GB/s on GeForce GTX 280

0.7 (1.4) GFLOP/s on 1 core of Xeon E5450

Alternative Forceless Model

An ***alternative model***, based on the work of *Patankar*, allows to simulate the flow of rigid particles in a fluid
without the explicit calculation of the hydrodynamic forces!

The general idea of the model can be summarized as follows:

1. The Navier-Stokes equations are solved everywhere with ***different densities*** for the fluid and the rigid body.
2. In a ***postprocessing step*** the solution for the rigid body is ***projected*** from a fluid motion onto the motion of a rigid body.
3. The rigid body is moved according to the velocity calculated in the postprocessing step. Start the next time step n+1 at Step 1.

Fluid Motion Model

Consider the flow of a solid particle with ***density*** ρ_s in a fluid with ***density*** ρ_f and ***viscosity*** v. Denote by $\Omega_f(t)$ the ***domain*** occupied by the fluid at time t, by $\Omega_P(t)$ the ***area*** occupied by the particle at time t and let $\Omega = \Omega_f \cup \Omega_P$.

The fluid flow is modelled by the **Navier-Stokes equations** with the ***intermediate velocity*** $\tilde{\mathbf{u}}$:

$$\rho\left(\frac{\partial\tilde{\mathbf{u}}}{\partial t}+(\tilde{\mathbf{u}}\cdot\nabla)\tilde{\mathbf{u}}\right)-\nu\Delta\tilde{\mathbf{u}}+\nabla\tilde{p}=\rho g \quad ,\nabla\cdot\tilde{\mathbf{u}}=0$$

$$\rho=\rho_f\left(1-H^{n+1}\right)+\rho_s H^{n+1} \qquad H^{n+1}=\begin{cases}0 & in\ \Omega_f\\ 1 & in\ \Omega_P\end{cases}$$

Projection onto Rigid Motion

The quantities that govern the motion of a rigid body are its ***translational velocity*** **u** and its ***angular velocity*** ω. In the projection step we calculate these quantities from the solution of the fluid.

The ***projection*** onto rigid body motion is realized as follows:

$$\mathbf{U}^{n+1} = \frac{\rho_s}{M} \int_P \tilde{\mathbf{u}}^{n+1} \, d\Omega$$

$$\omega^{n+1} = \left(\int_P \mathbf{r} \times \tilde{\mathbf{u}}^{n+1} \, d\Omega \right) / M \cdot \left(\int_P |\mathbf{r}|^2 d\Omega \right)$$

The velocity inside the rigid body in the next time step n+1 is then set to:

$$\mathbf{u}^{n+1} = \mathbf{U}^{n+1} + \omega^{n+1} \times \mathbf{r} \quad in \quad P^{n+1}$$

Summary and further capabilites of the new method:

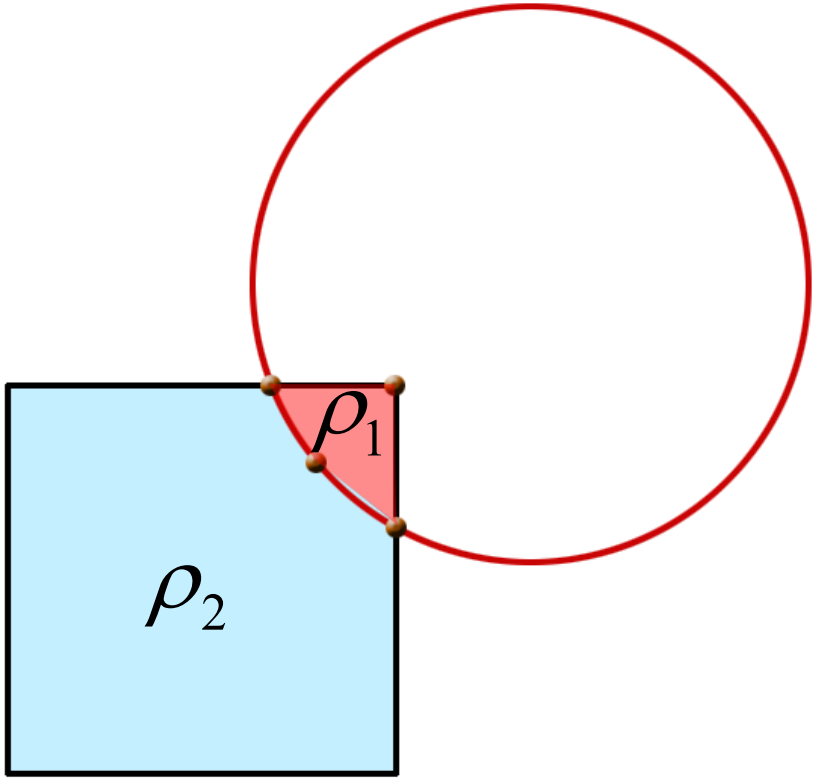
- No explicit calculation of the hydrodynamic forces.
- The calculation of the particle motion requires only the evaluation of two integrals.
- Better integration results by using the penalty method.
- The model is capable of handling non-rigid bodies or very complicated geometries by adding a level-set function

Penalty Method

Using the usual FBM-approach we get a function that is non-continuous on an element.

By using a ***penalty approach*** we can smooth this function to make it continuous, which will give more accurate results during integration.

This is done by calculating the area that fluid and respectively particle occupy and setting the value for the element to:

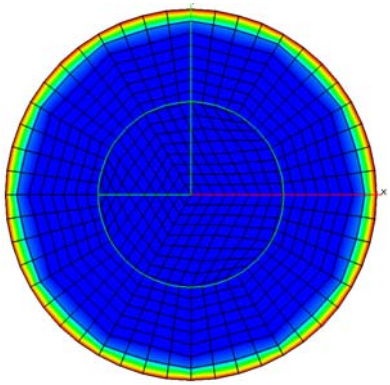


$$\rho_{Element} = \frac{a_{Fluid}}{a_{Element}} \cdot \rho_2 + \frac{a_{Solid}}{a_{Element}} \cdot \rho_1$$

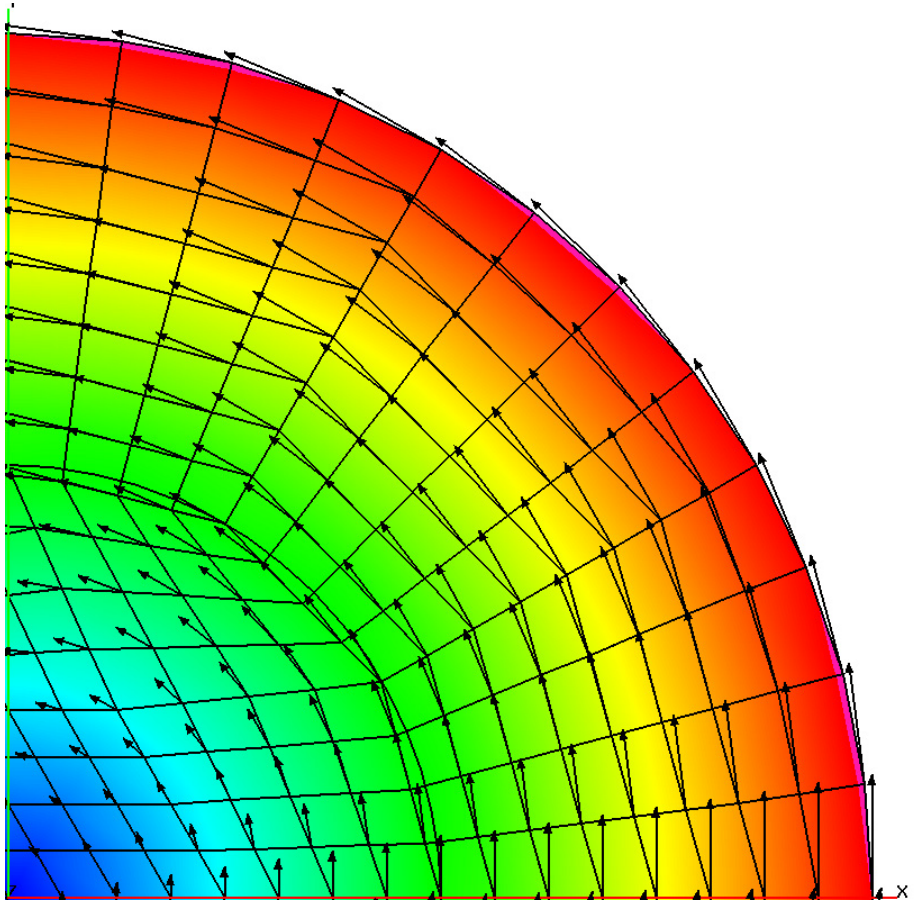
Rotating Disc Example

technische universität dortmund

viscosity	AngularVelReference	Angular Velocity Disc	Time	Grid Level 4
0.001	1.0000000000E+00	1.237011596120E+00	7.000000000000E+01	46272
0.01	1.0000000000E+00	1.055243213435E+00	4.100000000000E+01	46272
0.1	1.0000000000E+00	1.027059348253E+00	1.800000000000E+00	46272
1	1.0000000000E+00	1.024181253403E+00	1.000000000000E+00	46272



Examples



Particulate Flow Example

technische universität dortmund

LVL	Velocity Reference	Velocity	Grid Cells
6	3.0000000000E-01	2.780000000000E-01	1536
7	3.0000000000E-01	2.850000000000E-01	6144
8	3.0000000000E-01	2.909000000000E-01	24576
9	3.0000000000E-01	2.999700000000E-01	98304

Sedimentation Example

technische universität dortmund