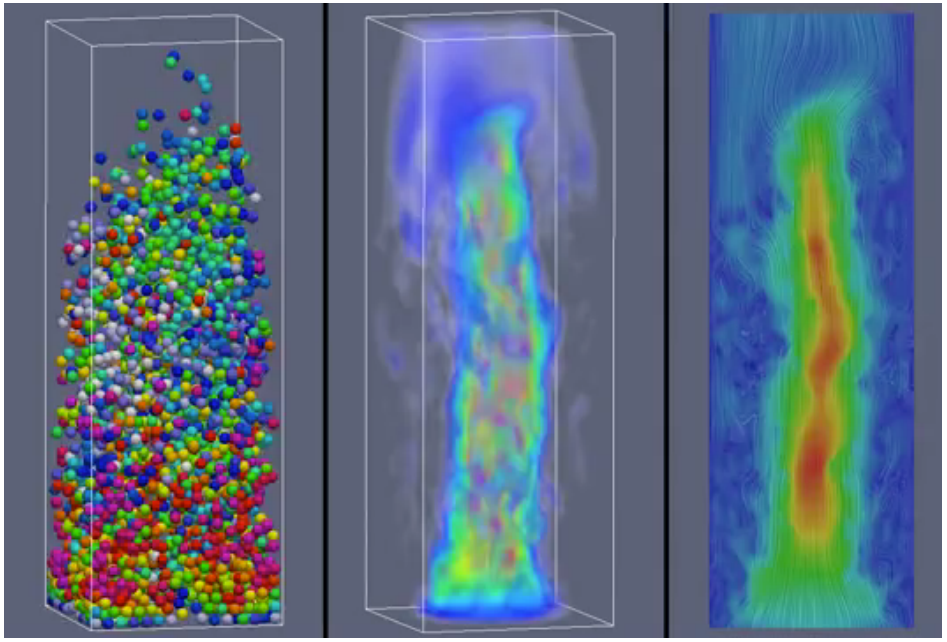
Particulate Flow Simulations with Complex Geometries using the Finite Element-Fictitious Boundary Method



Raphael Münster, Otto Mierka, Stefan Turek
Institut für Angewandte Mathematik, TU Dortmund

Related Multiphase Flow Solver

Basic CFD tool – **FEATFLOW**
(robust, parallel, efficient)

Numerical features:
- Higher order Q2P1 FEM schemes
- FCT & EO FEM stabilization techniques
- Use of unstructured meshes
- **Fictitious Boundary (FBM) methods**
- Dynamic adaptive grid deformation
- Newton-Multigrid solvers

HPC features:
- Massively parallel
- GPU computing
- Open source

Non-Newtonian flow module:
- generalized Newtonian model (Power-law, Carreau, ... etc.)
- viscoelastic model (Giesekus, Oldroyd B, …etc.)

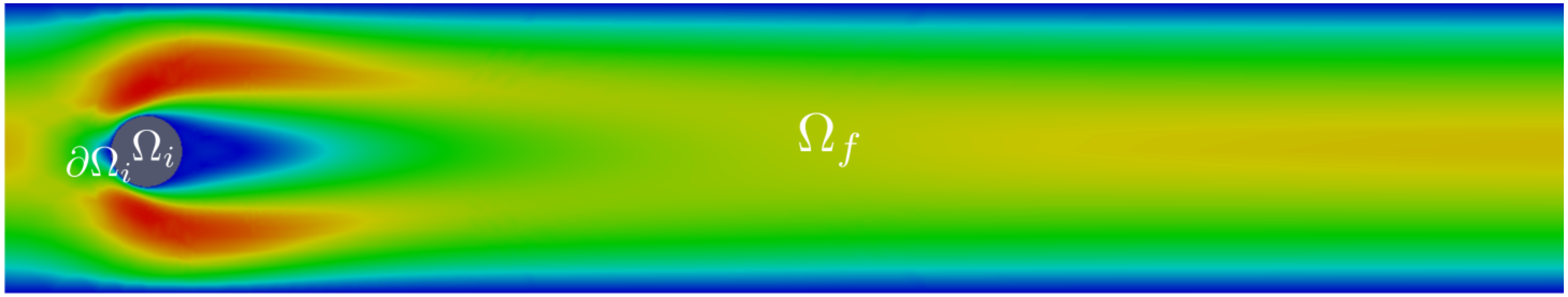
Multiphase flow module (resolved interfaces):
- l/l – interface tracking (Level Set)
- s/l – interface capturing (FBM)
- s/l/l – combination of l/l and s/l

Engineering aspects:
- Geometrical design
- Modulation strategy
- Optimization

FEM-based simulation tools for the accurate prediction of multiphase flow problems, particularly with liquid-(rigid) solid interfaces

Liquid–(Rigid) Solid Interfaces

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 $\Omega_i(t)$ the domain occupied by the ith-particle at time t and let $\overline{\Omega} = \overline{\Omega}_f \cup \overline{\Omega}_i$.



The fluid flow is modelled by the **Navier-Stokes equations**:

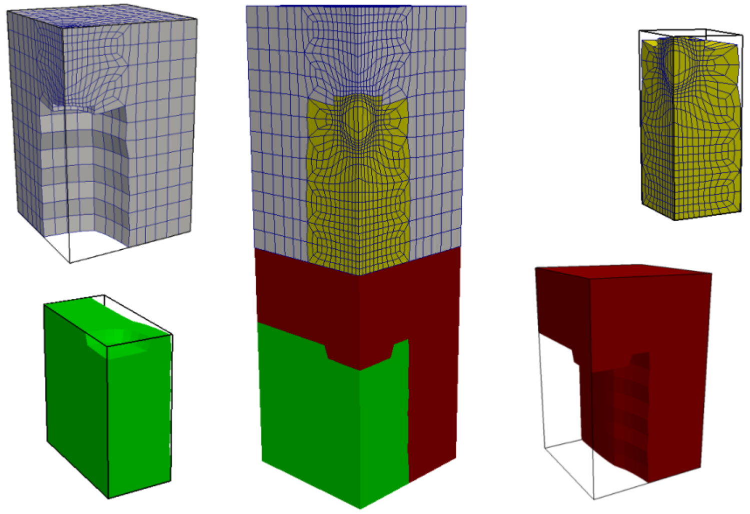
$$\rho\left(\frac{\partial u}{\partial t} + u \cdot \nabla u\right) - \nabla \cdot \sigma = f, \quad \nabla \cdot u = 0$$

where σ is the total stress tensor of the fluid phase:

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

Mesh Setup

- Hierachical unstructured meshes
- Domain decomposition:
  - → Grid hierarchy on each subdomain
- Mapping from spatial coordinates to mesh cells (indices) generally not possible for unstructured meshes



$$f: p(x,y,z) \rightarrow cellIndex$$

- Overlay an additional structured grid layer (hashed uniform grids) to obtain position to mesh cell mapping
- Direct mapping from positions crucial for fast computations involving the mesh or the geometry represented by the mesh

Numerical Solution Scheme

Solve for velocity and pressure applying FBM-conditions

$$NSE\left(u_f^{n+1}, p^{n+1}\right) = BC\left(\Omega_i^n, u_i^n\right)$$

Calculate hydrodynamic force, torque and apply

$$F_i^{n+1} \qquad T_i^{n+1}$$

Contact force calculation

$$F_{c,i}^{n+1}$$

Compute new velocity and angular velocity

$$u_i^{n+1} = u_i^n + \Delta t(F_{c,i}^{n+1}/M_i) \quad \omega_i^{n+1} = \omega_i^n + \Delta t I_i^{-1}\left(r \times F_{c,i}\right)$$

Position update

$$X_i^{n+1} = X_i^n + \Delta t u_i^n \qquad \theta_i^{n+1} = \theta_i^n + \Delta t \omega_i^n$$

Equations of Motion (I)

The motion of particles can be described by the **Newton-Euler equations**.
A particle moves with **a translational velocity** U_i and **angular velocity** ω_i
which satisfiy:

$$M_i \frac{dU_i}{dt} = F_i + F_i' + (\Delta M_i)g, \qquad I_i \frac{d\omega_i}{dt} + \omega_i \times (I_i \omega_i) = T_i,$$

M_i	: mass of the i-th particle (i=1,...,N)
I_i	: moment of inertia tensor of the i-th particle
ΔM_i	: mass difference between M_i and the mass of the fluid
F_i	: hydrodynamic force acting on the i-th particle
T_i	: hydrodynamic torque acting on the i-th particle

Equations of Motion (II)

The position and orientation of the i-th particle are obtained by integrating the **kinematic equations**:

$$\frac{dX_i}{dt} = U_i\,, \quad \frac{d\theta_i}{dt} = \omega_i\,, \quad \frac{d\omega_i}{dt} = I_i^{-1} T_i$$

which can be done numerically by an explicit Euler scheme:

$$X_i^{n+1} = X_i^n + \Delta t U_i^n \qquad \omega_i^{n+1} = \omega_i^n + \Delta t\left(I_i^{-1} T_i^n\right) \qquad \theta_i^{n+1} = \theta_i^n + \Delta t \omega_i^n$$

Boundary Conditions

We apply the velocity u(X) as no-slip boundary condition at the interface $\partial\Omega_i$ between the i-th particle and the fluid, which for $X \in \Omega_i$ is defined by:

$$u(X) = U_i + \omega_i \times (X - X_i)$$

Hydrodynamic Forces

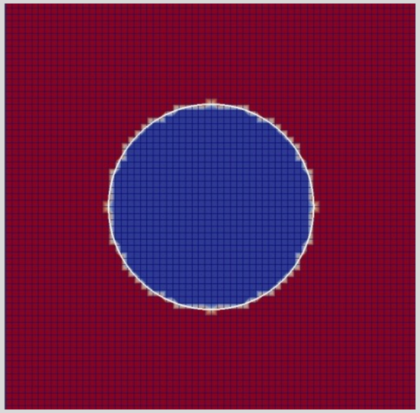
Hydrodynamic force and torque acting on the i-th particle

$$F_i = -\int_{\partial\Omega_i} \sigma \cdot n_i d\Gamma_i, \quad T_i = -\int_{\partial\Omega_i} (x - x_i) \times (\sigma \cdot n_i) d\Gamma_i$$

Force Calculation with Fictitious Boundary Method

<u>The FBM can only decide:</u>

- \`INSIDE\`(1) and \`OUTSIDE\`(0)
- Only first order accuracy



Alternative:

Replace the surface integral by a volume integral

Numerical Force Evaluation

Define an ***indicator function*** α_i:

$$\alpha_i(x) = \begin{cases} 1 \text{ for } x \in \Omega_i \\ 0 \text{ for } x \in \Omega_f \end{cases}$$

Remark: The gradient of α_i is zero everywhere except at the surface of the i-th Particle and approximates the normal vector (in a weak sense), allowing us to write:

$$F_i = -\int_{\Omega_T} \sigma \cdot \nabla\alpha_i d\Omega, \;\; T_i = -\int_{\Omega_T} (x - x_i) \times (\sigma \cdot \nabla\alpha_i) d\Omega$$

On the finite element level we can compute this by:

$$F_i = -\sum_{T \in T_{h,i}} \int_{\Omega_T} \sigma_h \cdot \nabla\alpha_{h,i} d\Omega$$

$$T_i = -\sum_{T \in T_{h,i}} \int_{\Omega_T} (x - x_i) \times (\sigma_h \cdot \nabla\alpha_{h,i}) d\Omega$$

$\alpha_{h,i}$ (x) : finite element interpolant of α(x)

$T_{h,i}$: elements intersected by i-th particle

Grid Deformation and ALE

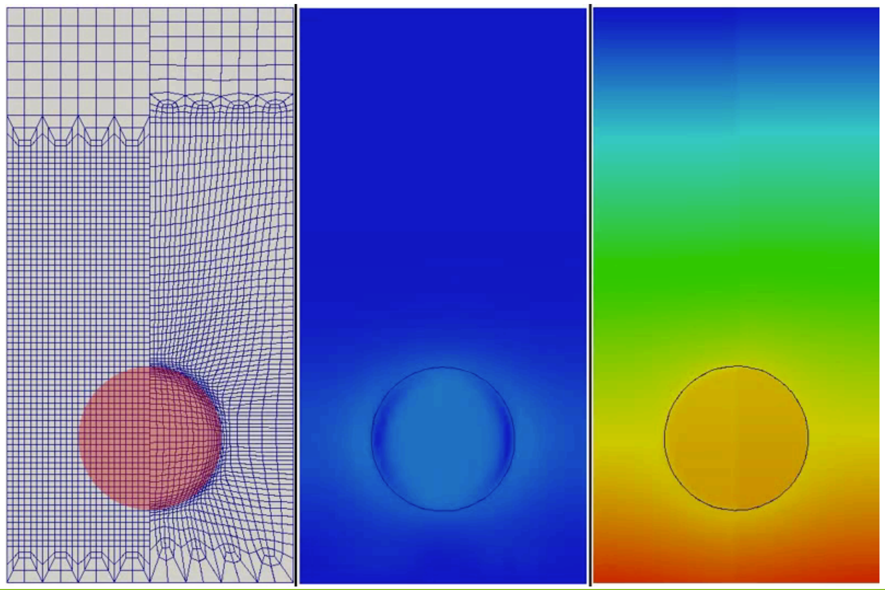
Advantages:

- Constant mesh/data structure → **GPU**
- Increased resolution in regions of interest
- PDE approach is **not** necessary → anisotropic 'umbrella' smoother
- Straightforward usage in 3D unstructured meshes

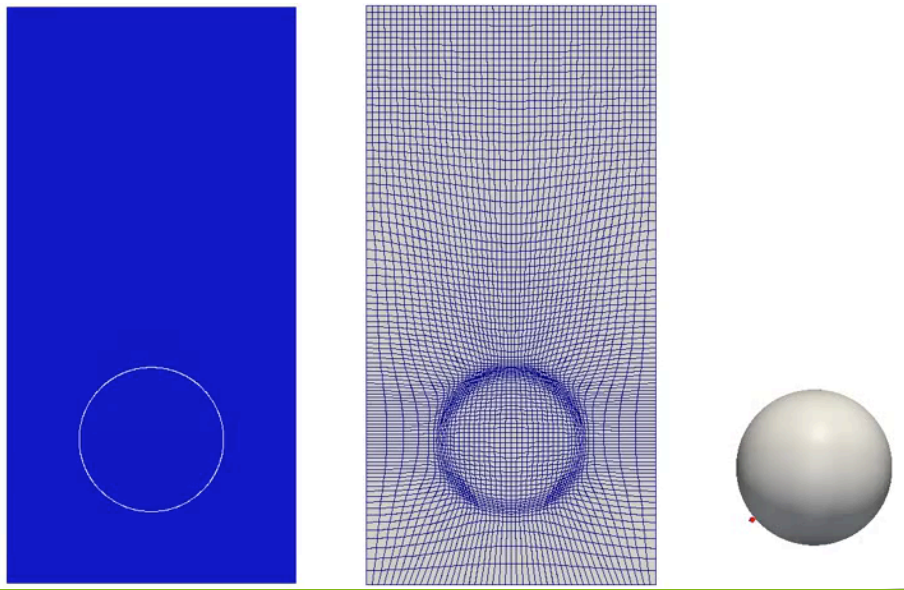
Quality of the method depends on the construction of the monitor function

- Geometrical description (solid body, interface triangulation)
- Monitor function based on distance information
- Field oriented description (steep gradients, fronts) → numerical stabilization

Validation: 2.5D Rising bubble – light setup



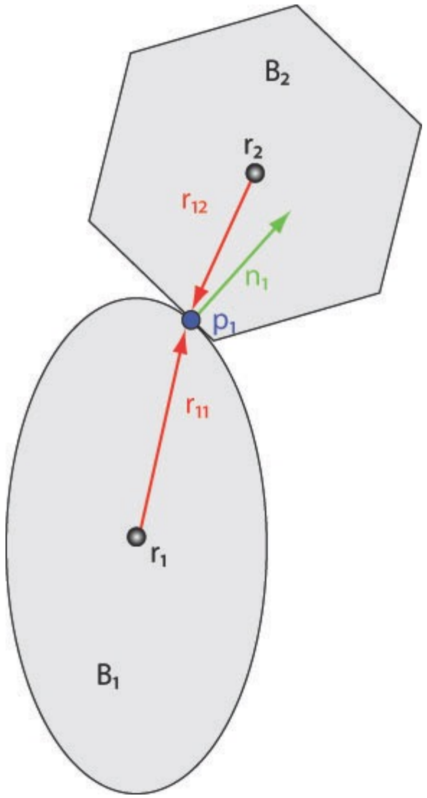
Testing: 3D Rising bubble - hard setup



Contact Force Calculation

- Contact force calculation realized as a three step process
  - → Broadphase
  - → Narrowphase
  - → Contact/Collision force calculation
- Worst case complexity for collision detection is $O(n^2)$
  - → Computing contact information is expensive
  - → Reduce number of expensive tests → Broad Phase
- *Broad phase*
  - → Simple rejection tests exclude pairs that cannot intersect
  - → Use hierarchical spatial partitioning
- *Narrow phase*
  - → Uses Broadphase output
  - → Calculates data neccessary for collision force calculation

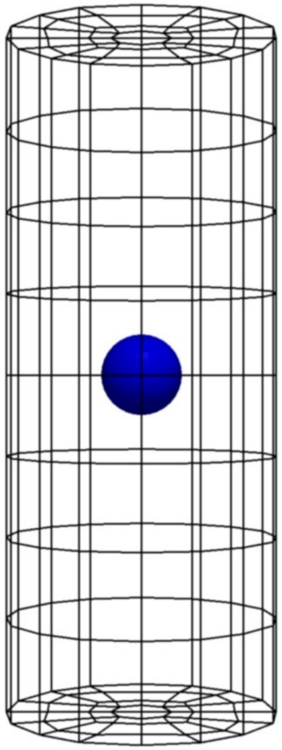
► Special single, resp., multibody collision models (as **linear complementarity problems) on GPUs**

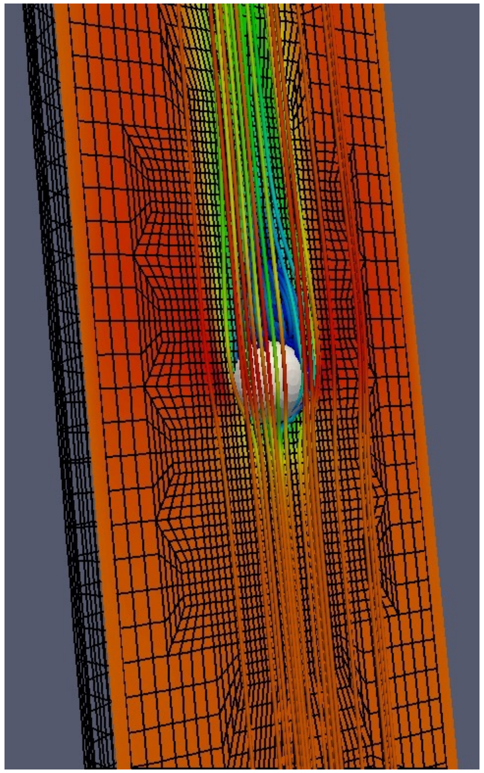


Benchmarking and Validation (I)

Free fall of particles:

- Terminal velocity
- Different physical parameters
- Different geometrical parameters





Münster, R.; Mierka, O.; Turek, S.: Finite Element fictitious boundary methods (FEM-FBM) for 3D particulate flow, IJNMF, 2011

$d_s = 0.3, \quad \rho_s = 1.14$

ν	$U_{featflow}$	U_{exp}	Relative error (%)
0.02	5.885	6.283	6.33
0.05	4.133	3.972	4.05
0.1	2.588	2.426	6.66
0.2	1.492	1.401	6.50

$d_s = 0.2, \quad \rho_s = 1.14$

ν	$U_{featflow}$	U_{exp}	Relative error (%)
0.02	4.370	4.334	0.83
0.05	2.699	2.489	8.44
0.1	1.649	1.552	6.25
0.2	0.946	0.870	8.74

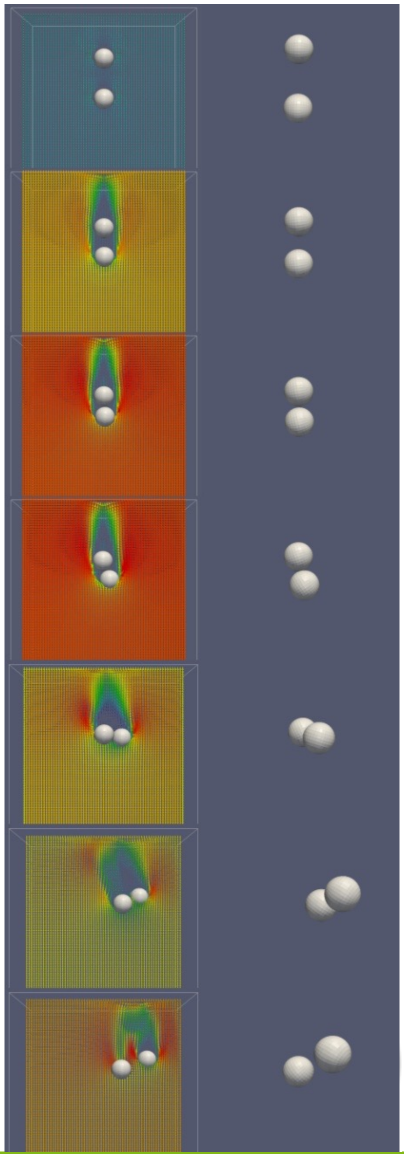
$d_s = 0.3, \quad \rho_s = 1.02$

ν	$U_{featflow}$	U_{exp}	Relative error (%)
0.01	2.167	2.107	2.84
0.02	1.495	1.436	4.11
0.05	0.809	0.749	8.01
0.1	0.402	0.404	0.44
0.2	0.218	0.216	1.02

$d_s = 0.2, \quad \rho_s = 1.02$

ν	$U_{featflow}$	U_{exp}	Relative error (%)
0.01	1.4660	1.4110	3.90
0.02	0.9998	0.9129	9.52
0.05	0.4917	0.4603	6.82
0.1	0.2637	0.2571	2.57
0.2	0.1335	0.1317	1.37

Source: Glowinski et al. 2001

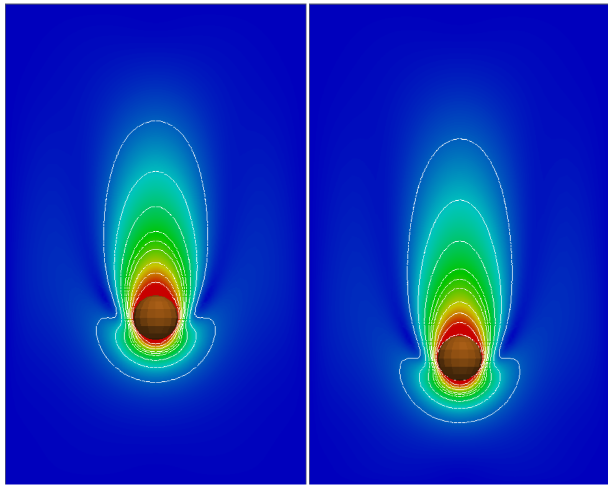


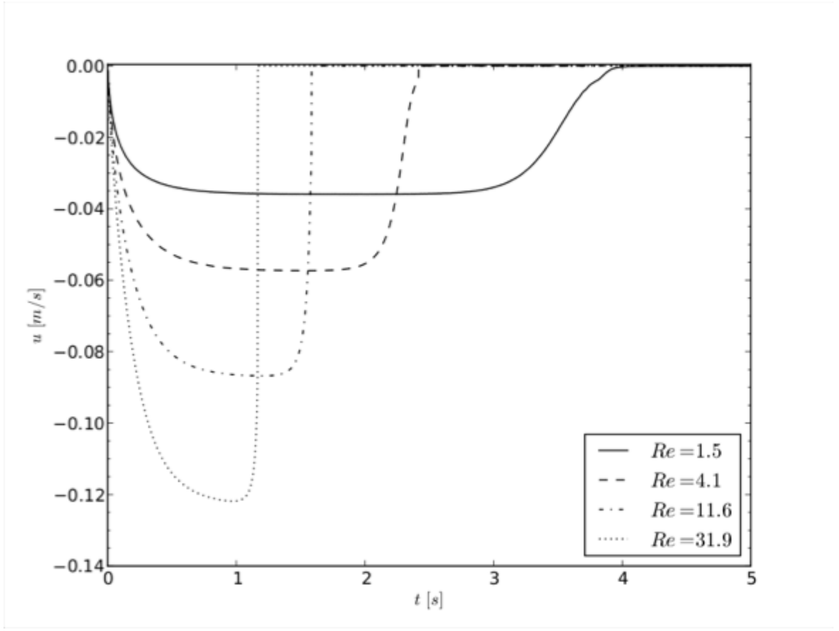
Sedimentation Benchmark

Setup

Computational mesh:
- 1.075.200 vertices
- 622.592 hexahedral cells
- Q2/P1:
  - $\rightarrow$ 50.429.952 DoFs

Hardware Resources:
- 32 Processors

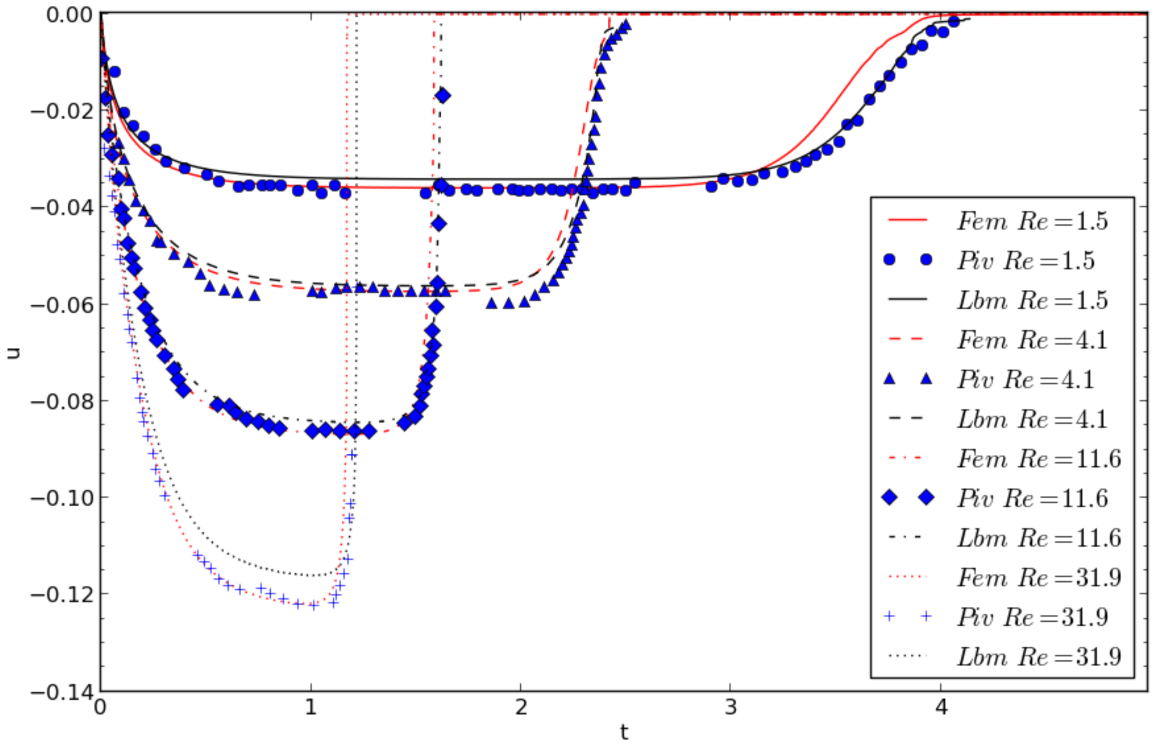




Re	u_{max}/u_∞	u_{max}/u_∞ *ten Cate*	u_{max}/u_∞ *exp*
1.5	0.945	0.894	0.947
4.1	0.955	0.950	0.953
11.6	0.953	0.955	0.959
31.9	0.951	0.947	0.955

Tab. 1 Comparison of the u_{max}/u_∞ ratios between the FEM-FBM, ten Cate's simulation and ten Cate's experiment

Comparison



Comparison of FEM-FBM and the experimental values and the LBM results of the group of Sommerfeld

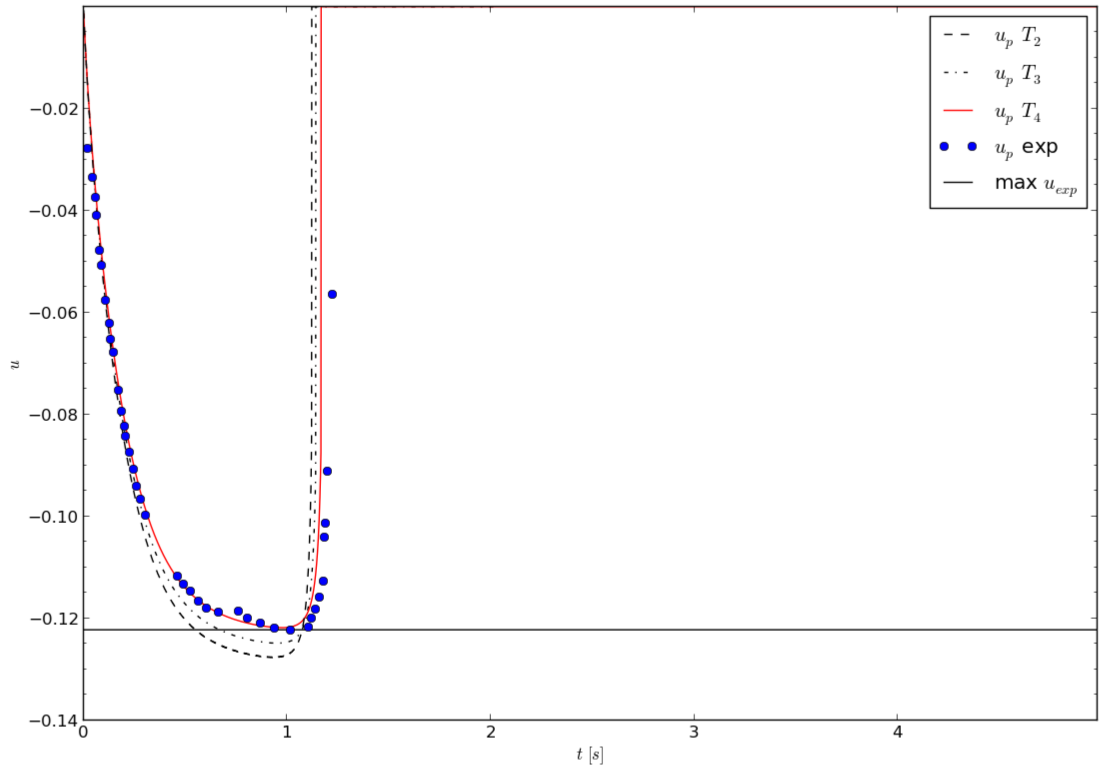
Source: 13th Workshop on Two-Phase Flow Predictions 2012
Acknowledgements: Ernst,M., Dietzel,M., Sommerfeld,M.

Multi-level Analysis

FEM-Multigrid Framework

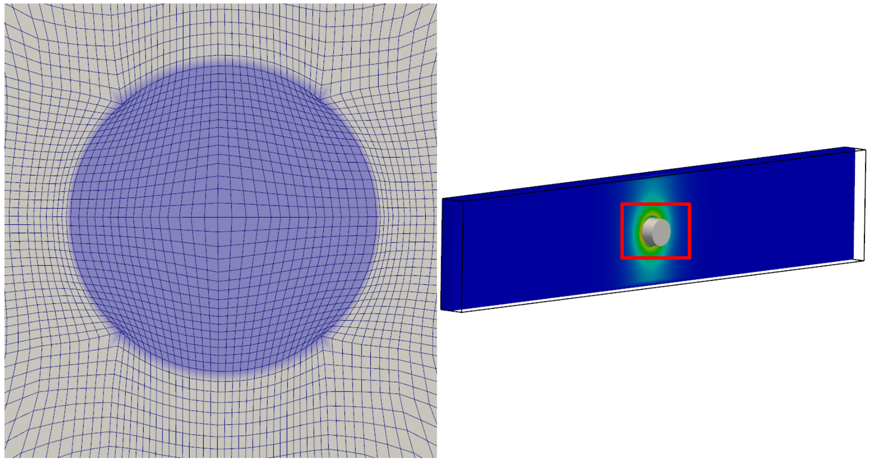
- *Increasing the mesh resolution produces more accurate results*

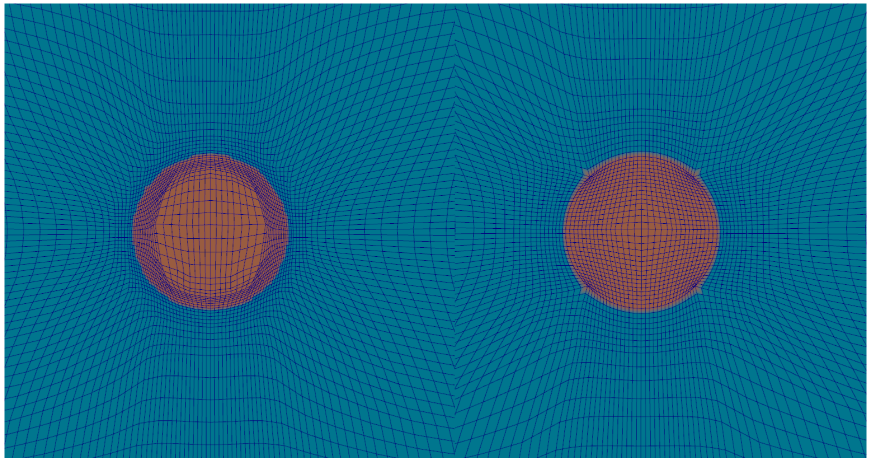
  Test performed at different mesh levels
  - Maximum velocity is approximated better ✓
  - Shape of the velocity curve matches better ✓



Oscillating Cylinder

- Measure Drag/Lift Coefficients for a sinusoidally oscillating cylinder
- Compare results for FBM, adapted FBM and adapted FBM + boundary projection/parametrization

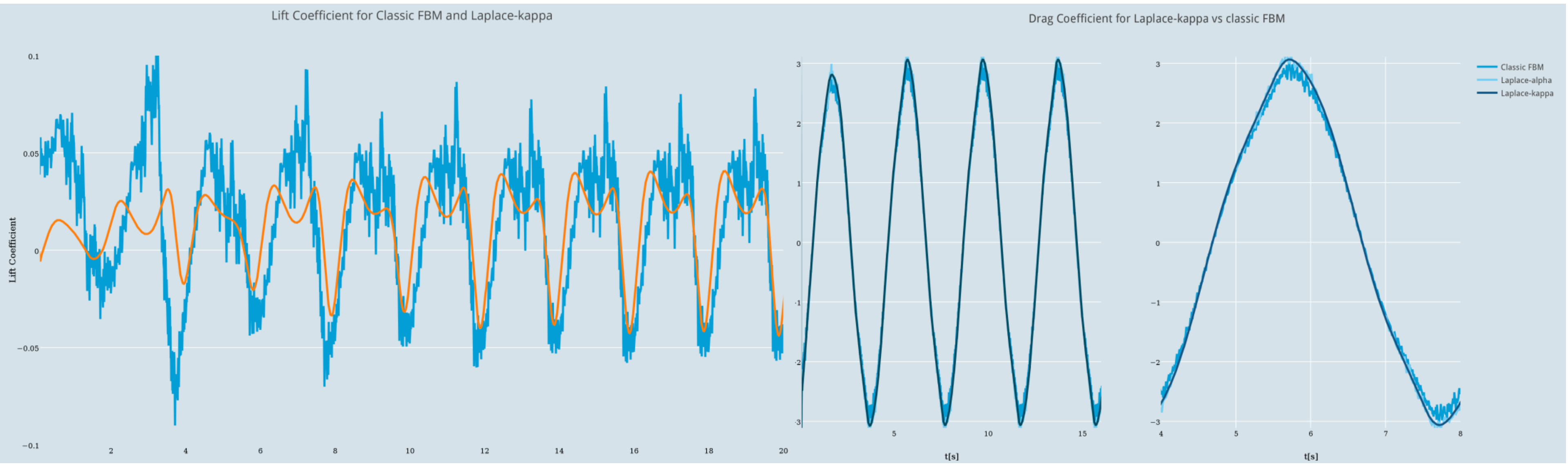


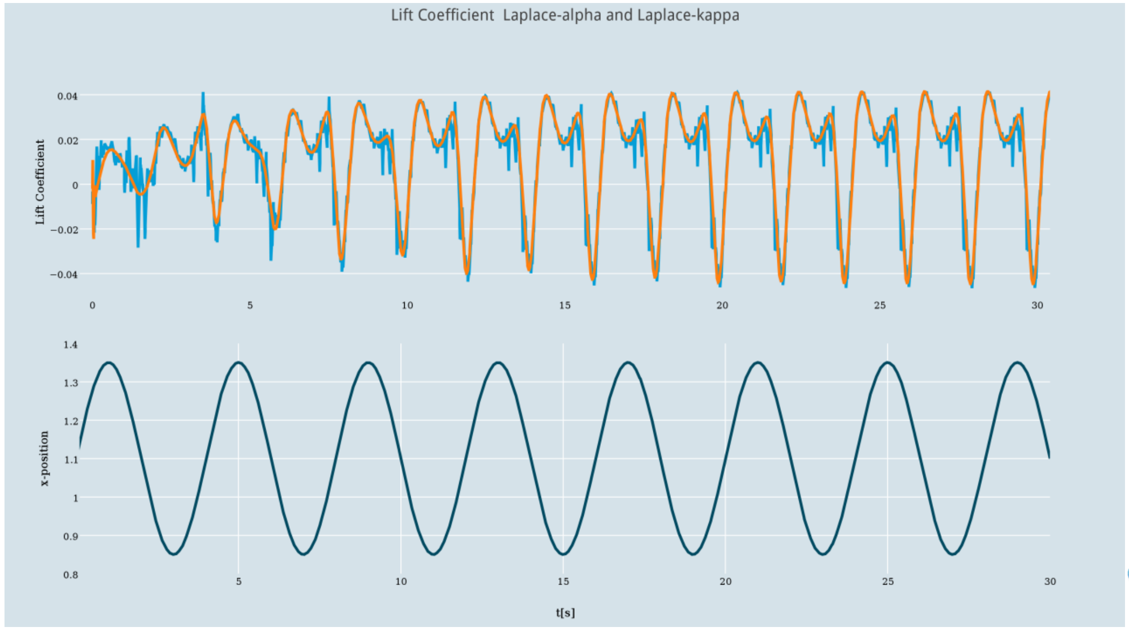


Nodes concentrated near liquid-solid interface

Nodes projected and parametrized on boundary plus concentration of nodes near boundary

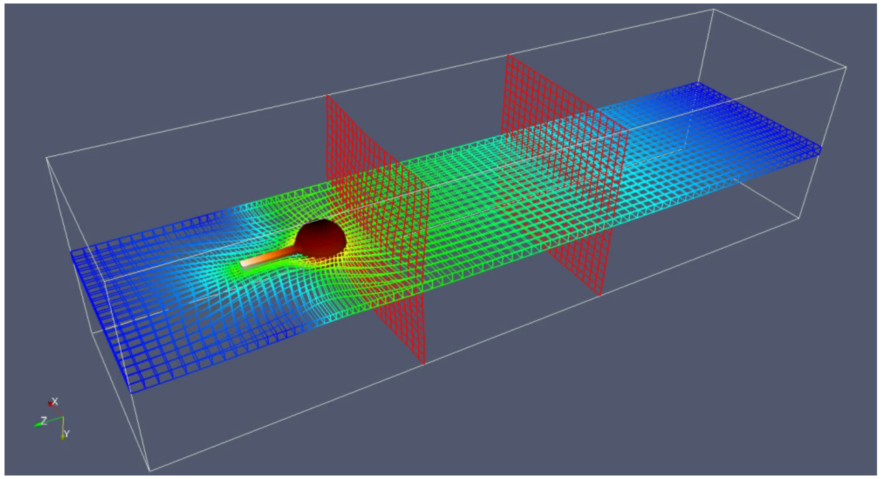
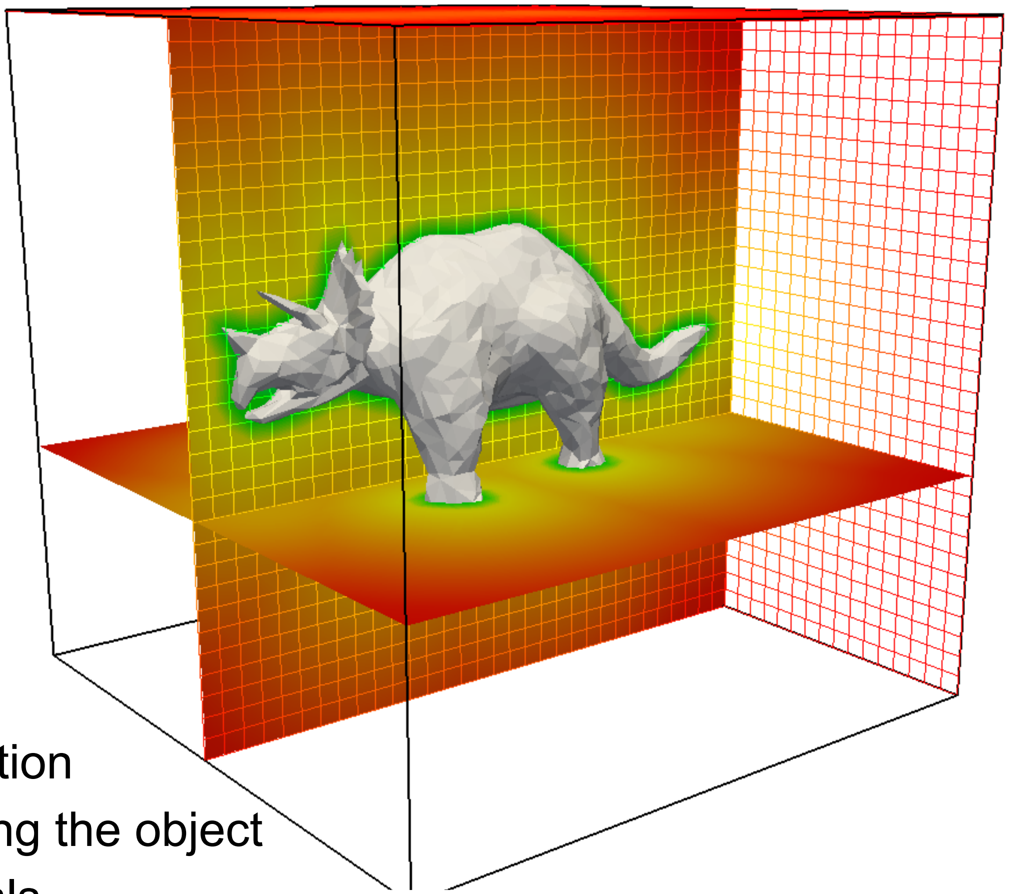
Oscillating Cylinder Results





- Highly smooth results when the vertices are projected directly onto the geometry ✓

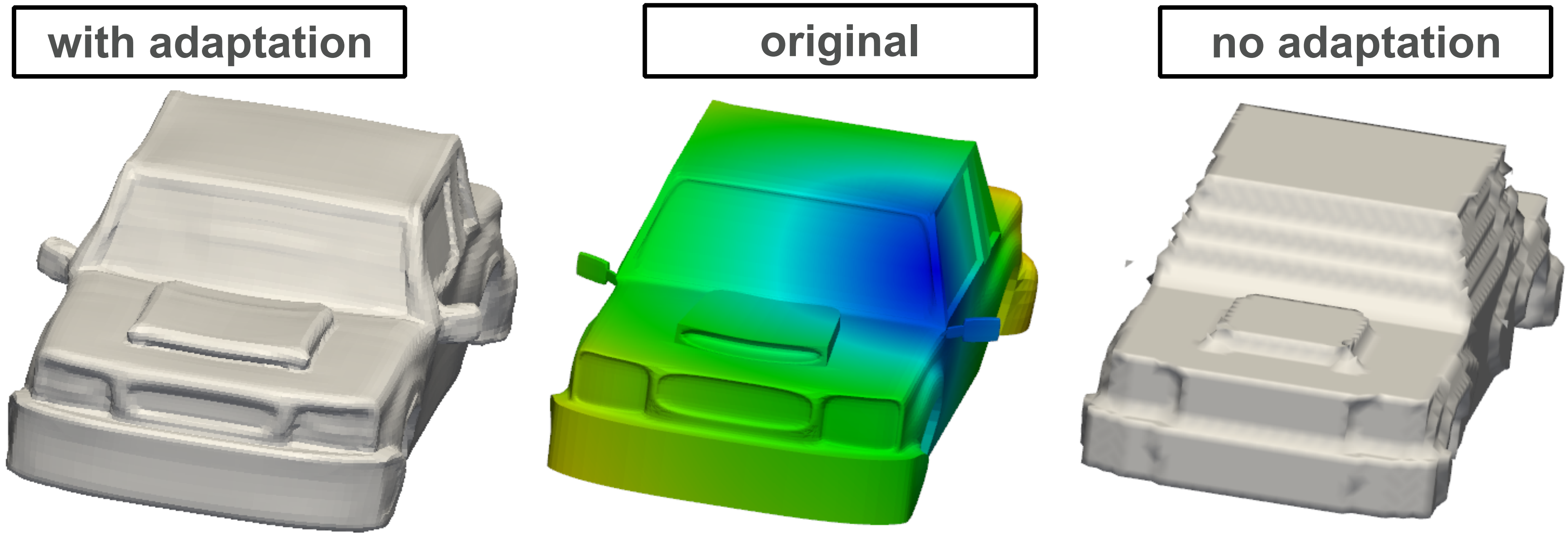
Distance Maps for Fast MinDist

- Data structure for fast distance calculation
- Equidistant structured mesh surrounding the object
- Precompute and store distance, normals
- Transform quantities into distance map, use precomputed values
- Algorithm maps excellently to the GPU
- Provides fast distance computation and collision queries for complex geometries

Influence of Mesh Adaptation

Car representation by the computation mesh



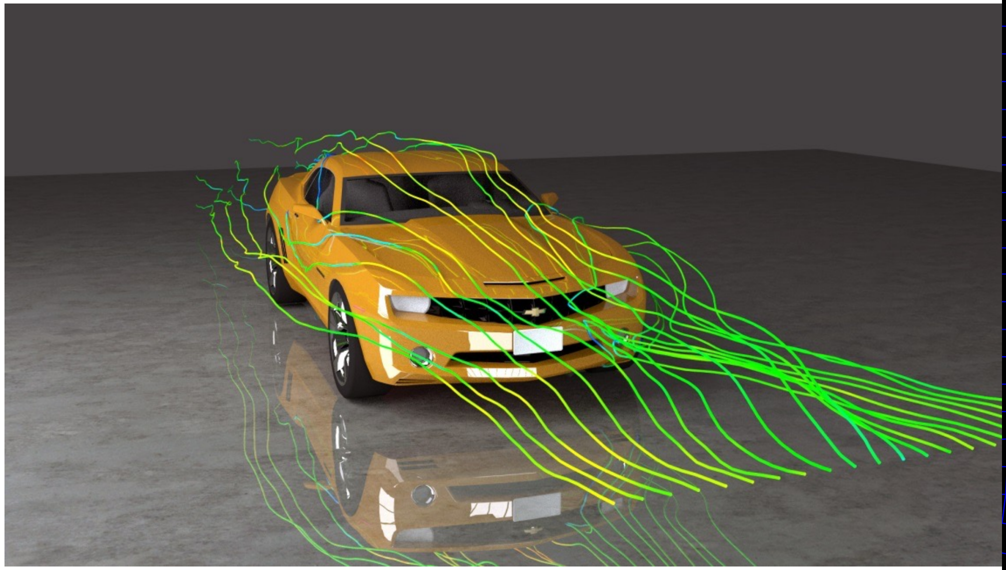
- Details may be lost without adaptation
- Better resolution with the same number of DOFs
- Mesh adaptation equivalent to at least one refinement level

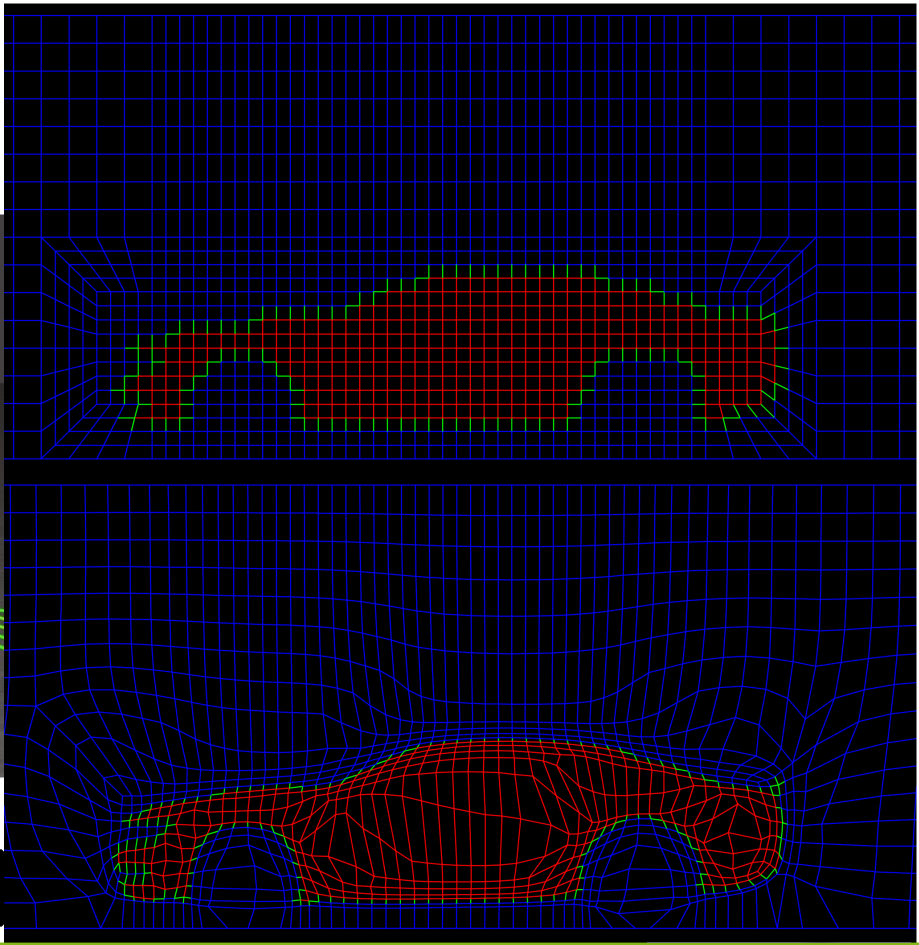
Example: Virtual Wind Tunnel

technische universität dortmund

- Numerical simulation of complex geometries
- Use of a regular base mesh
- Resolution of small scale details by mesh adaptation

Streamline visualization of the flow field around a car





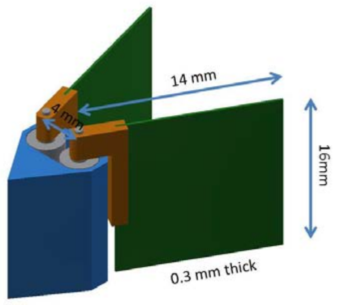
Mesh Slices with and without adaptation

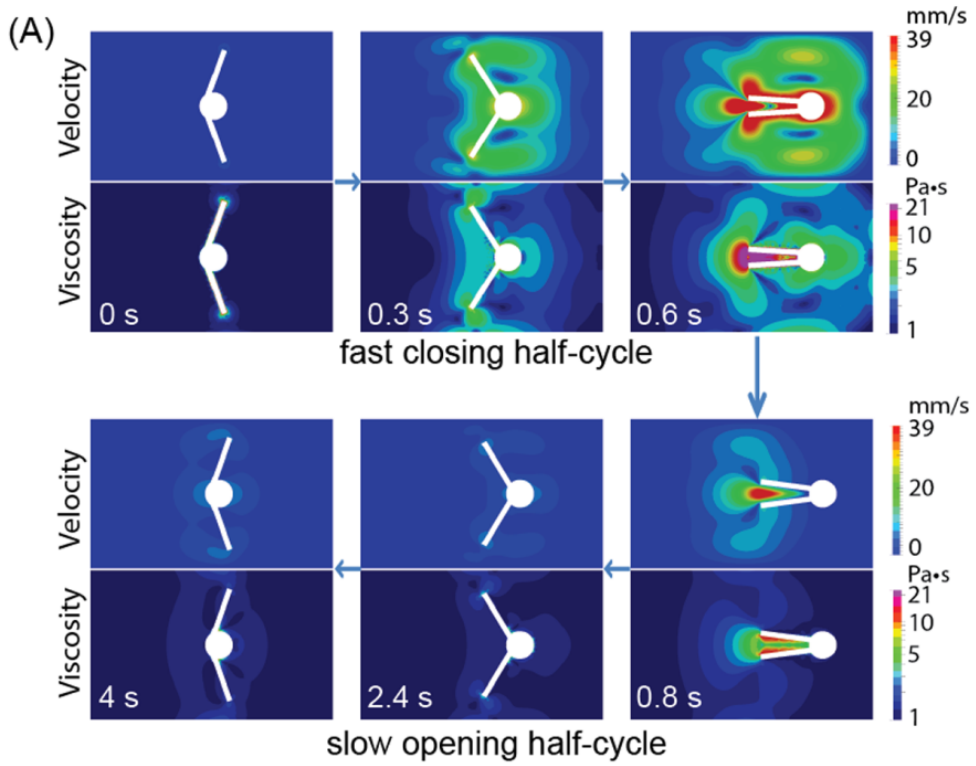
Microswimmer Example

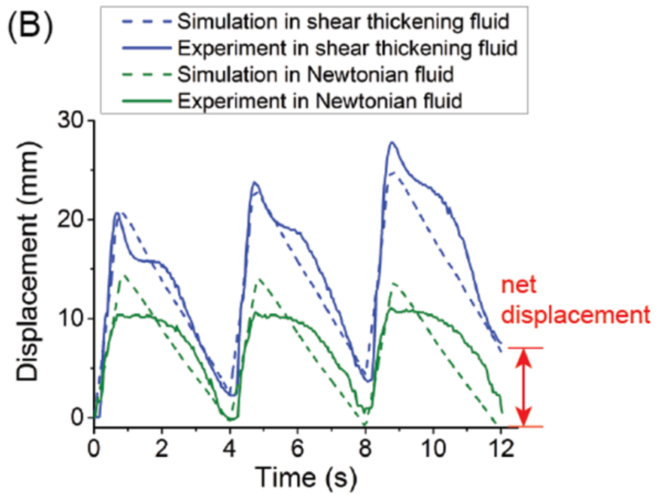
Swimming by Reciprocal Motion at Low Reynolds Number

Tian Qiu, Tung-Chun Lee, Andrew G. Mark, Konstantin I. Morozov , Raphael Münster , Otto Mierka , Stefan Turek, Alexander M. Leshansky and Peer Fischer

Nature Communications, November 2014



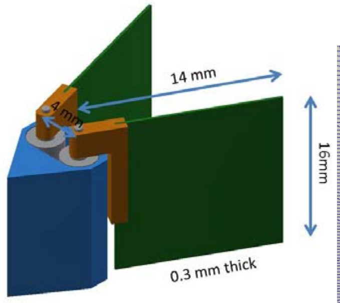


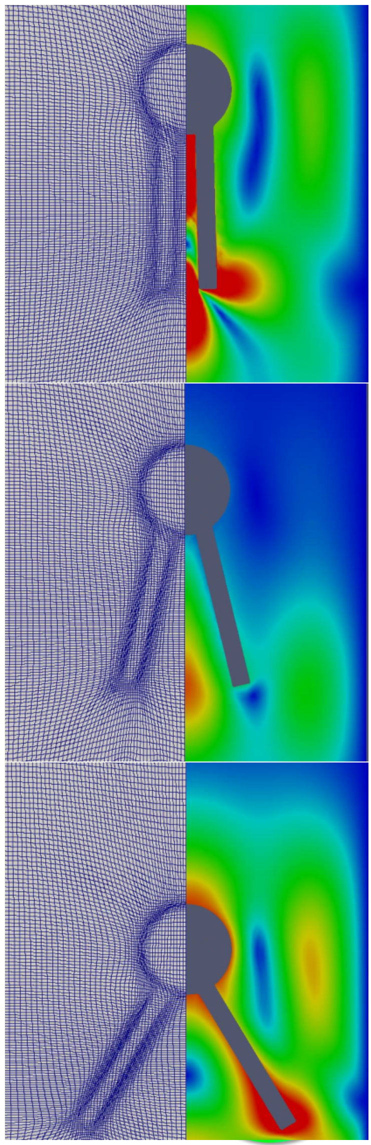


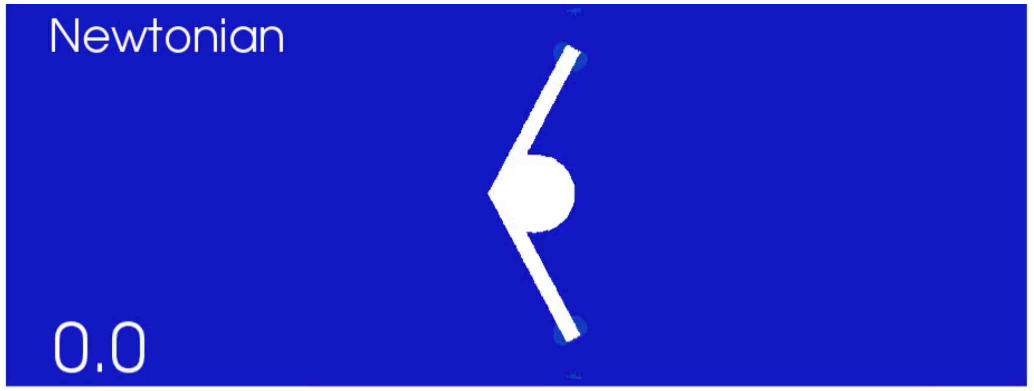
Microswimmer Example(II)

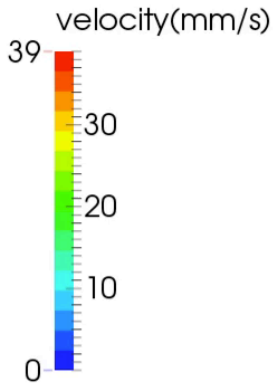
Application to microswimmers:

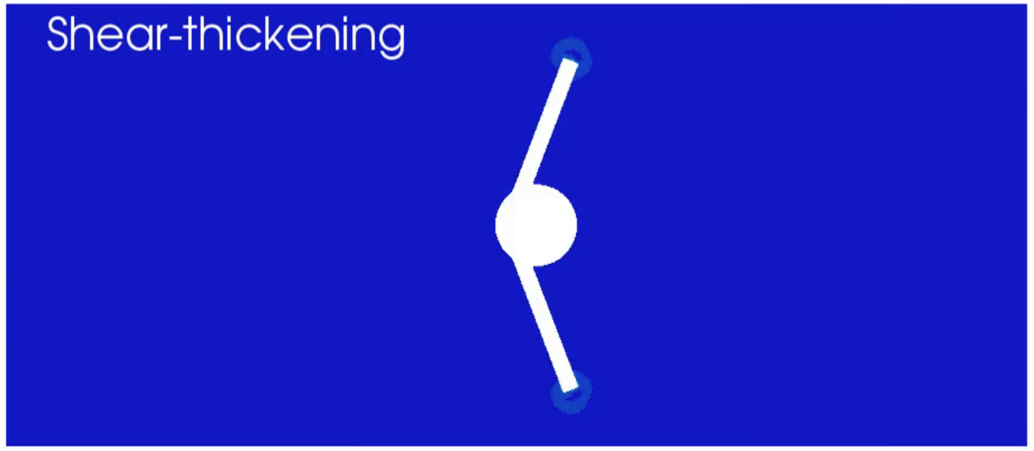
- Exp: Cooperation with Prof. Fischer (MPI IS Stuttgart)
- Analysis with respect to shear thickening/thinning
- Use of grid deformation to resolve *s/l* interface

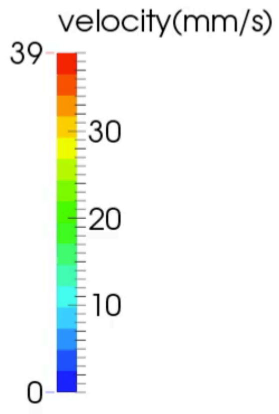






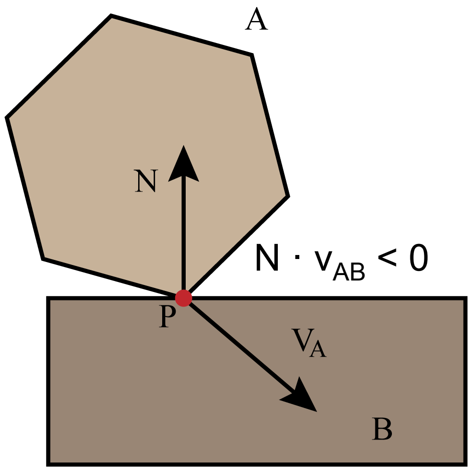


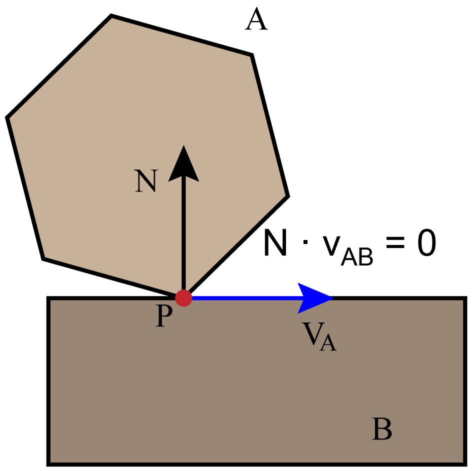


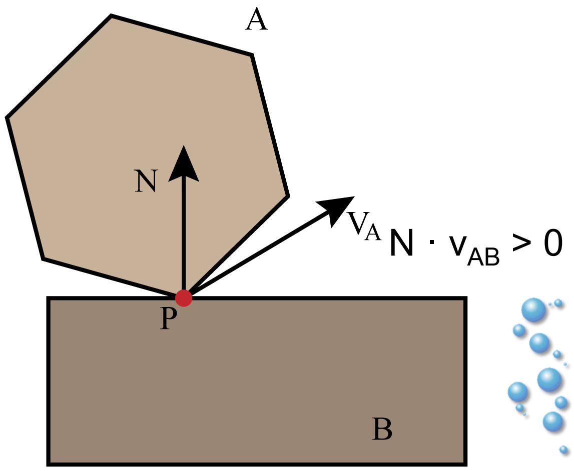


Contact/Collision Modelling

- Contact determination for rigid bodies A and B:
  - → Distance d(A,B)
  - → Relative velocity $v_{AB} = (v_A + \omega_A \times r_A - (v_B + \omega_B \times r_B))$
  - → Collision normal $N = (X_A(t) - X_B(t))$
  - → Relative normal velocity $N \cdot (v_A + \omega_A \times r_A - (v_B + \omega_B \times r_B))$
- distinguishes three cases of how bodies move relative to each other:
  - → Colliding contact : $N \cdot v_{AB} < 0$
  - → Separation : $N \cdot v_{AB} > 0$
  - → Touching contact : $N \cdot v_{AB} = 0$







Single Body Collision Model

technische universität dortmund

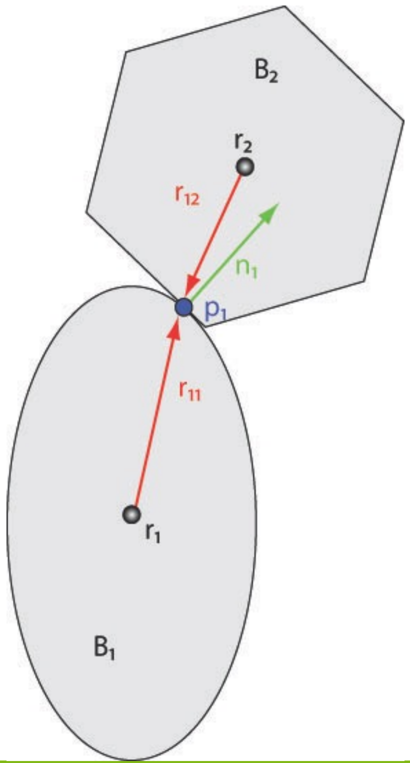
For a single pair of colliding bodies we compute the impulse f that causes the velocities of the bodies to change:

$$f = \frac{-(1+\varepsilon)\left(n_1(v_1 - v_2) + \omega_1(r_{11}\times n_1) - \omega_2(r_{12}\times n_1)\right)}{m_1^{-1} + m_2^{-1} + (r_{11}\times n_1)^T I_1^{-1}(r_{11}\times n_1) + (r_{12}\times n_1)^T I_2^{-1}(r_{12}\times n_1)}$$

Using the impulse f, the change in linear and angular velocity can be calculated:

$$v_1(t+\Delta t) = v_1(t) + \frac{fn_1}{m_1}, \quad \omega_1(t+\Delta t) = \omega_1(t) + I_1^{-1}(r_{11}\times fn_1)$$

$$v_2(t+\Delta t) = v_2(t) - \frac{fn_1}{m_2}, \quad \omega_2(t+\Delta t) = \omega_2(t) - I_2^{-1}(r_{12}\times fn_1)$$



Multi-Body Collision Model(I)

In the case of **multiple colliding bodies** with ***K* contact points** the impulses influence each other. Hence, they are combined into a **system of equations** that involves the following matrices and vectors:

- N: matrix of contact normals
- C: matrix of contact conditions
- M: rigid body mass matrix
- f: vector of contact forces ($f_i \geq 0$)
- f^{ext}: vector of external forces(gravity, etc.)

$$\underbrace{N^T C^T M^{-1} C N}_{A} \cdot \underbrace{\Delta t f^{t+\Delta t}}_{x} + \underbrace{N^T C^T (u^t + \Delta t M^{-1} + f^{ext})}_{b} \geq 0, f \geq 0$$

A problem of this form is called a **linear complementarity problem (LCP)** which can be solved with efficient iterative methods like the **Projected Gauss-Seidel solver (PGS)**.

Kenny Erleben,*Stable, Robust, and Versatile Multibody Dynamics Animation*

Multi-Body Collision Model (II)

technische universität dortmund

Sequential Impulses

- Apply pairwise impulses iteratively
- Normal impulse

$$P_n = \max\left(\frac{-\Delta\bar{\mathbf{v}}\cdot\mathbf{n}}{k_n}, 0\right)$$

- Tangential (frictional) impulse

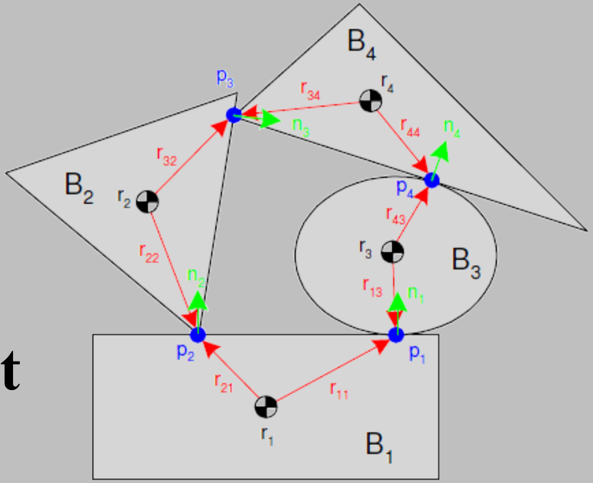
$$v_t = \Delta\mathbf{v}\cdot\mathbf{t}$$

$$-\mu P_n \le P_t \le \mu P_n$$

- Terminate when:
  - Impulses become small
  - Iteration limit is reached

$$P_t = \mathrm{clamp}\left(\frac{-\Delta\bar{\mathbf{v}}\cdot\mathbf{t}}{k_t}, -\mu P_n, \mu P_n\right)$$

$$k_t = \frac{1}{m_1} + \frac{1}{m_2} + \left[I_1^{-1}(\mathbf{r}_1\times\mathbf{t})\times\mathbf{r}_1 + I_2^{-1}(\mathbf{r}_2\times\mathbf{t})\times\mathbf{r}_2\right]\cdot\mathbf{t}$$

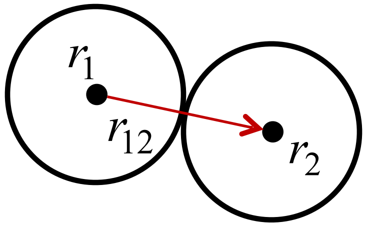


Details: Guendelman, *Nonconvex rigid bodies with stacking*

Multi-Body Collision Model (III)

Collision forces

- Use a DEM approach that can be easily evaluated in parallel
- Consider only the 3x3x3 neighbouring cells for each particle

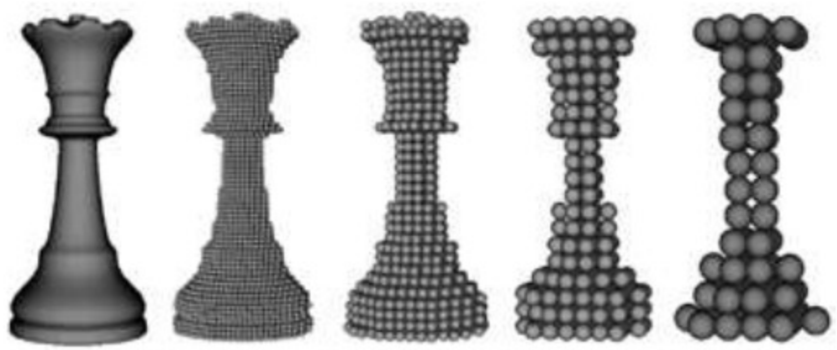


Forces acting on each particle

$$F_{i,s} = -k\left(d - |r_{ij}|\right)\frac{r_{ij}}{|r_{ij}|} \qquad F_{i,d} = \eta \cdot u_{ij}$$

$$F_{i,t} = k_t \cdot u_{ij,t} \qquad u_{ij,t} = u_{ij} - \left(u_{ij} \cdot \frac{r_{ij}}{|r_{ij}|}\right)\frac{r_{ij}}{|r_{ij}|}$$

k,η: material constants



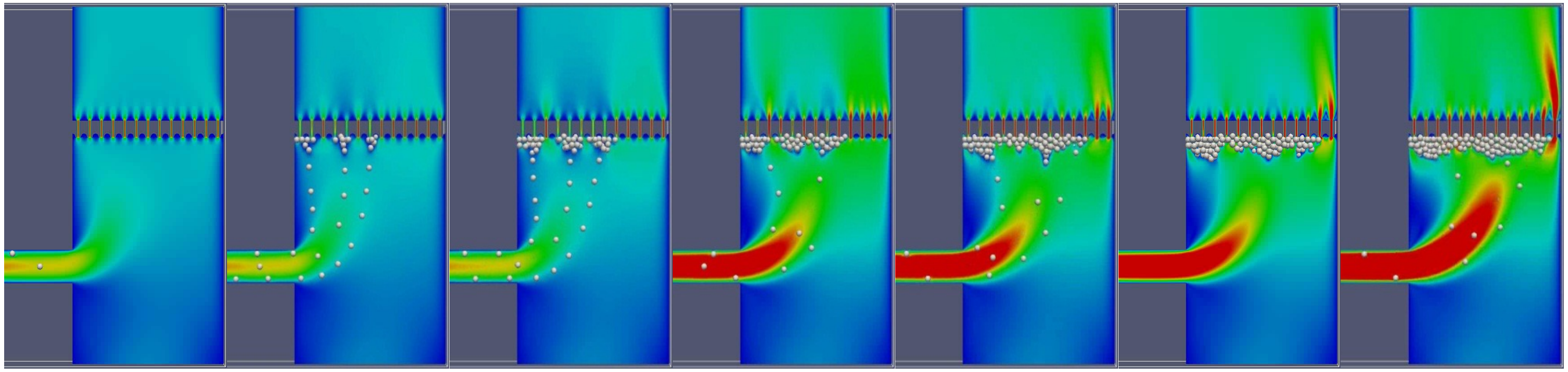
Sum up for each collision

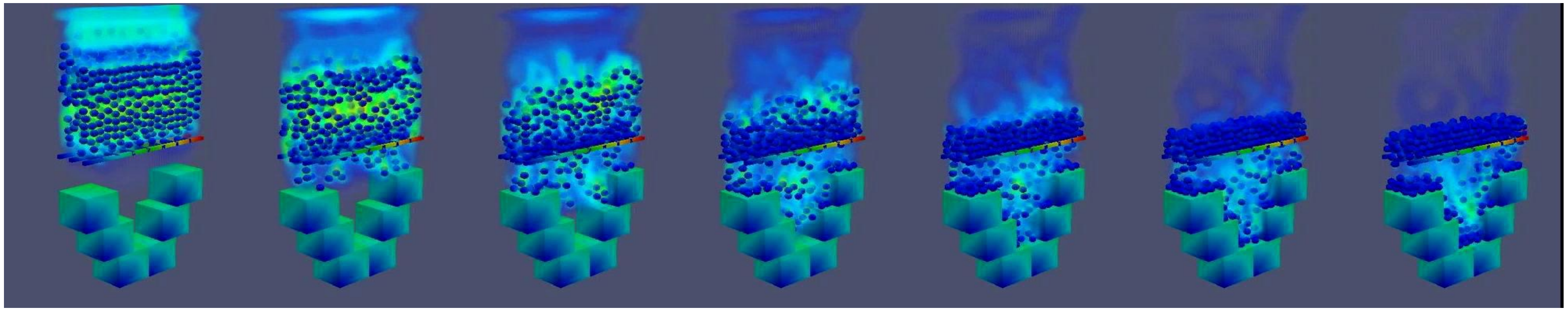
$$F_{i,c} = \sum_{\text{collisions}(i)} \left(F_{i,s} + F_{i,d} + F_{i,t}\right)$$

$$T_{i,c} = \sum_{\text{collisions}(i)} \left(r_i \times \left(F_{i,s} + F_{i,d} + F_{i,t}\right)\right)$$

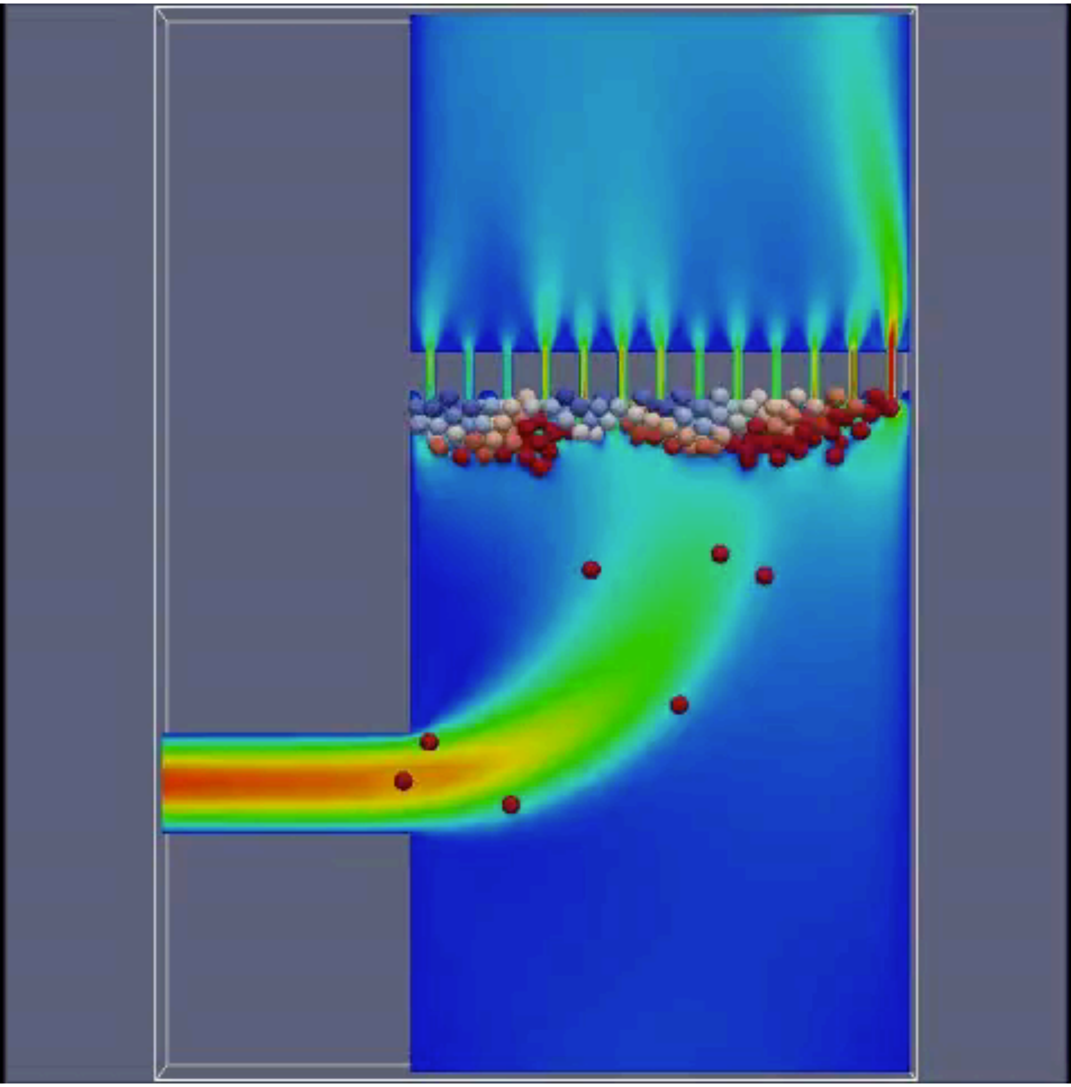
- Can be extended to rigid bodies
- Details: GPU Gems 3 (Takahiro Harada)

Examples

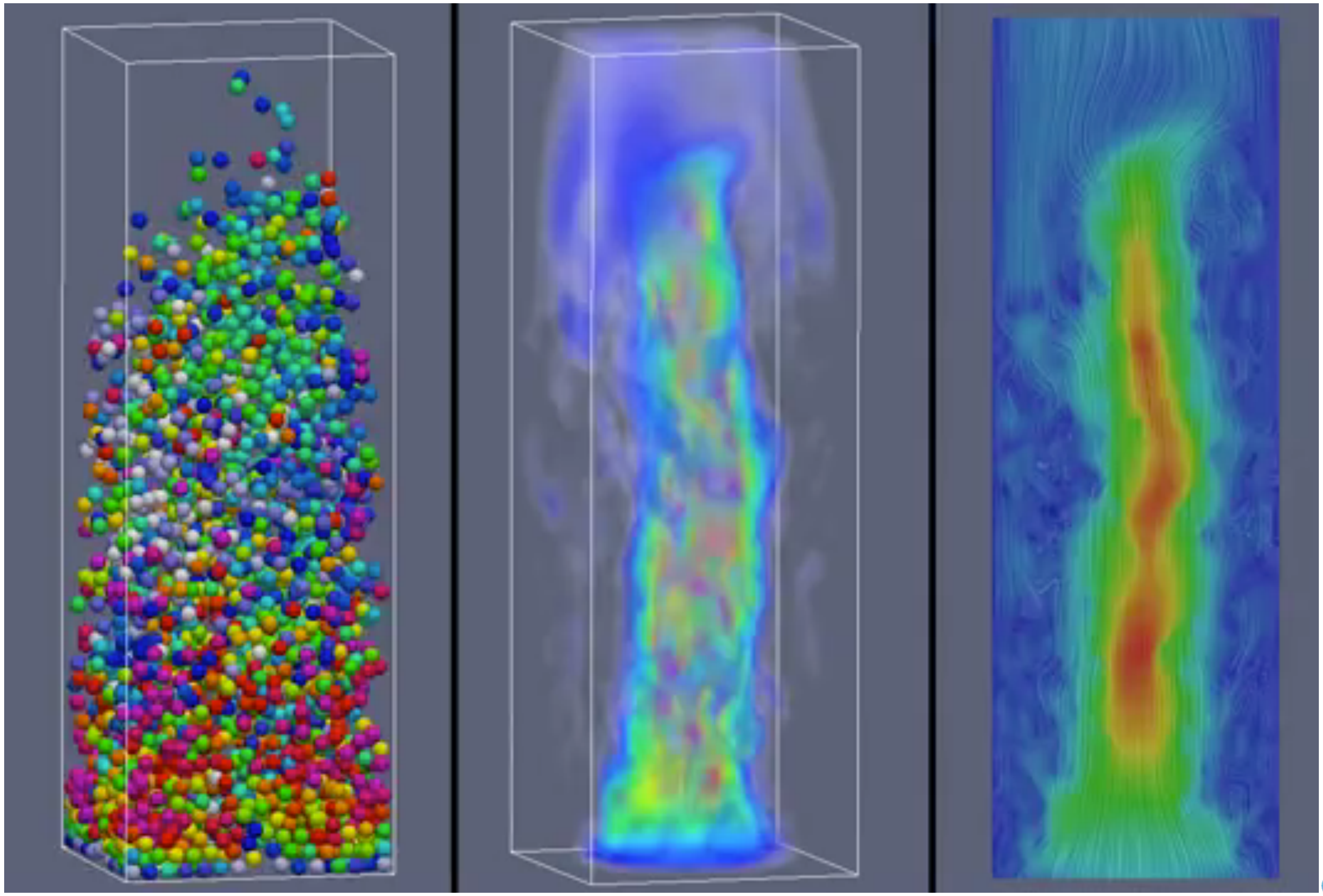




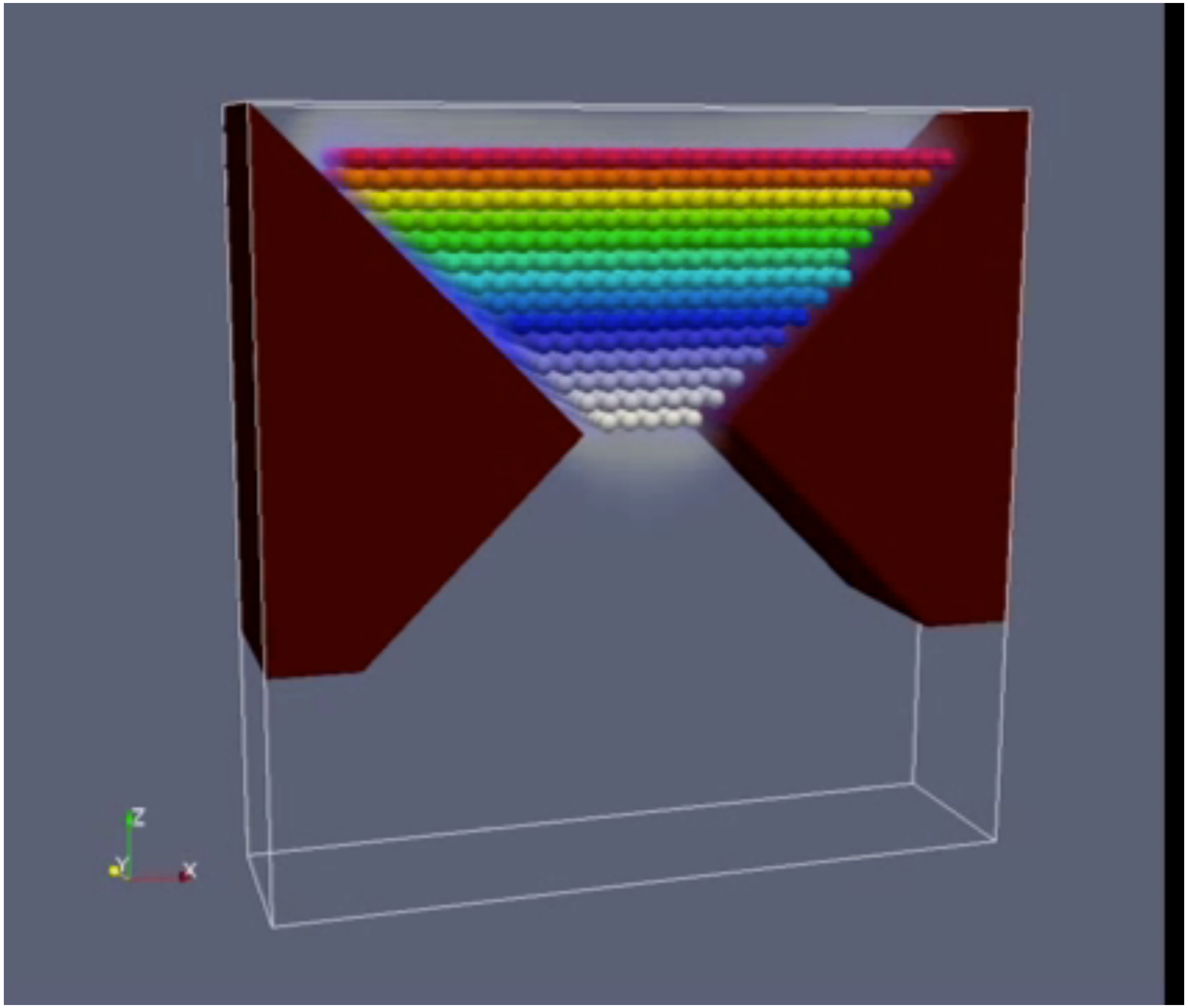
Examples



Fluidized Bed Example



DGS Configuration



Extensions & Future Activities

Fluidics

- Viscoelastic fluids
- Turbulence
- Multiphase problems
  - → Liquid-Liquid-Solid
  - → Liquid-Gas-Solid

Hardware-Oriented Numerics

- Improve parallel efficiency of collision detection and force computation on GPU
- Implement core CFD-Solver Modules on GPU
- Complete dynamic grid adaptation on GPU
- Hydrodynamic forces on GPU