



Current limitations and future developments

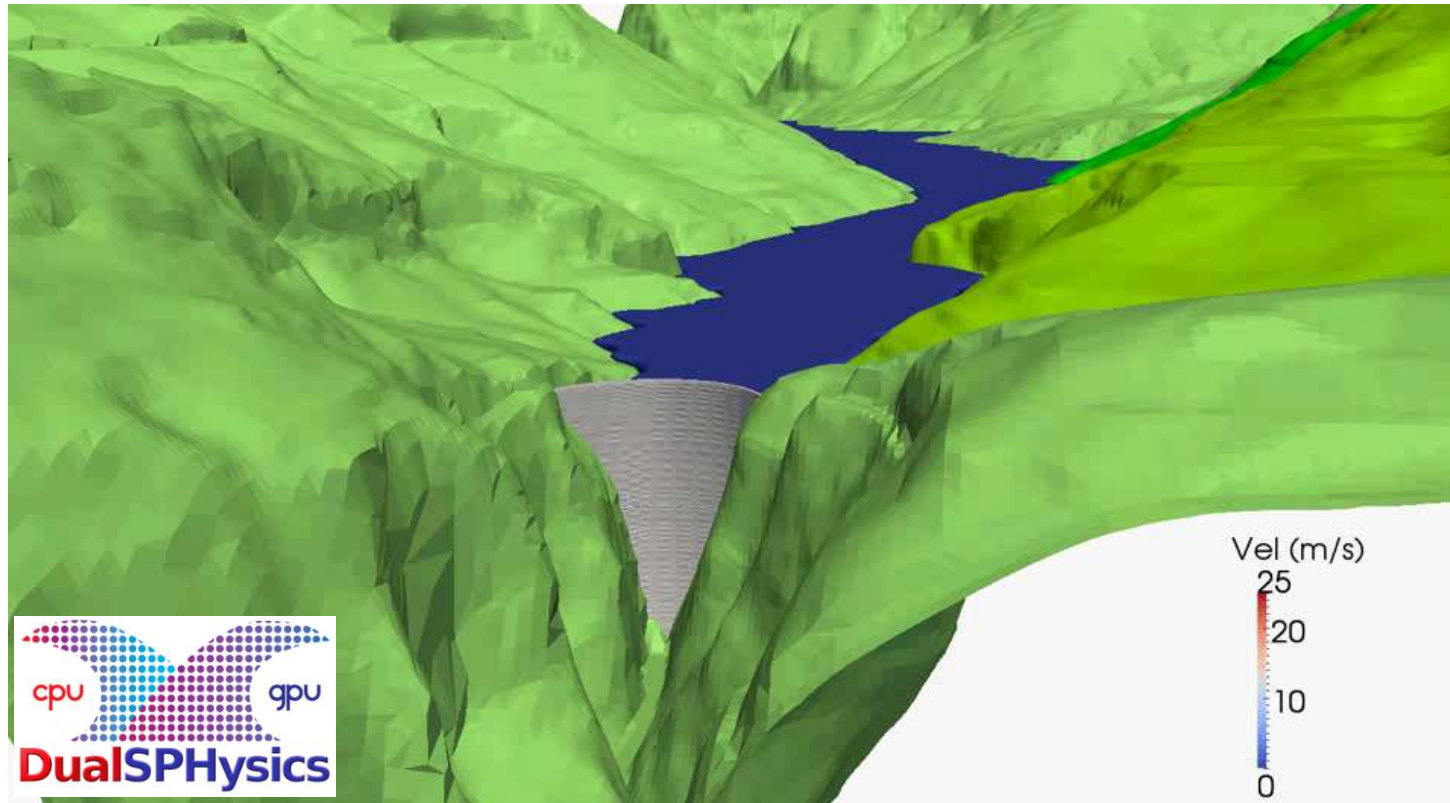
Dr Renato Vacondio
University of Parma
ITALY



2nd DualSPHysics User Workshop, 6-7 December 2016

Current limitations of DualSPHysics 4.0

Computational time for SPH with uniform resolution and 1 GPU

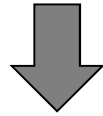


Vajont rockslide: On a GPU GTX 580, 4×10^6 particles ($\Delta x=5$ m)
requires 62 hours of computational time for 21 min of physical time

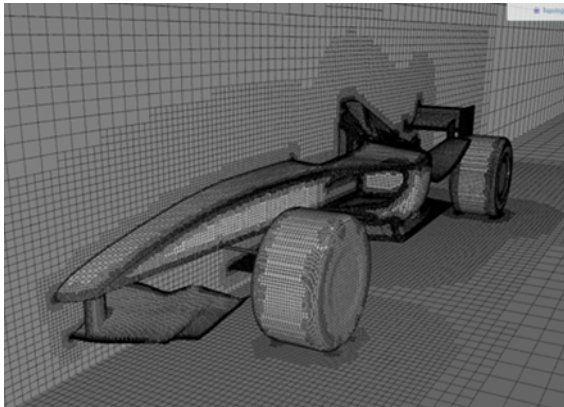
Vacondio et al. Advances in Water Resources (2013)

How do we overcome this limitation?

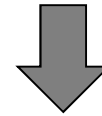
Eng Solution 1



Variable resolution



HPC Solution



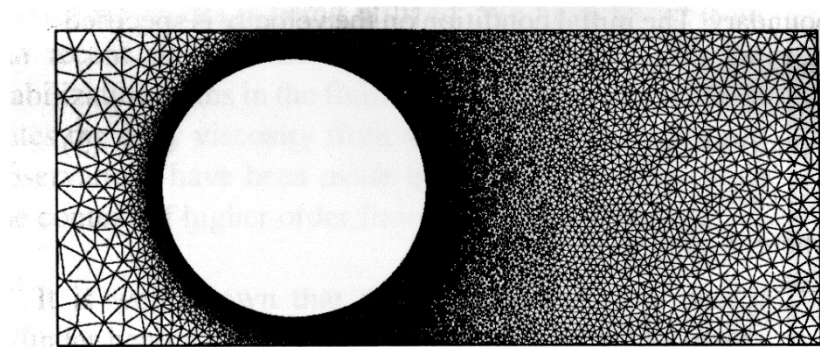
multiGPU



We need both

Motivation for adaptivity in SPH

- Most of the SPH codes are based on uniform resolution
- In Eulerian models variable resolution achieved many years ago using (dynamically adaptive) unstructured meshes:



Due to its Lagrangian nature, this is more challenging for SPH:

SPHERIC Grand Challenge #4: Can we achieve the same efficiency in SPH ?

Previous works about variable resolution in SPH :

Remeshing:

- *Koumoutsakos Ann. Rev. Fluid Mech. (2005)*: applied remeshing idea to SPH
- Multiblock space discretization: *Børve et al. JCP (2005)*

Static refinement:

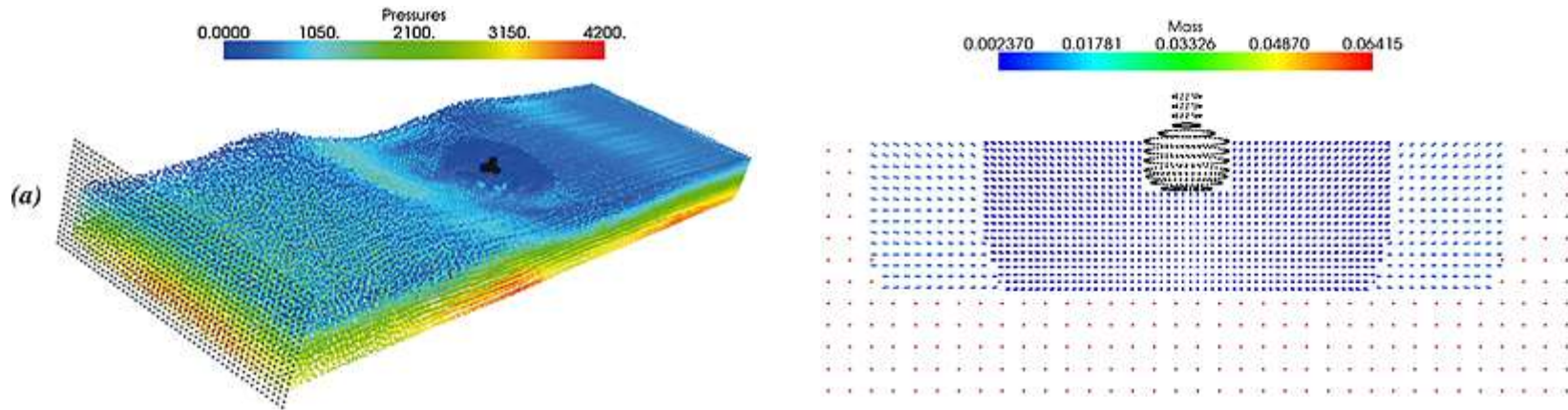
- Different initial resolution zones, no splitting: *Oger et al. JCP (2006)* and *Omidvar et al. IJNMF (2012)*.

Dynamic refinement:

- Particle insertion and removal in 1D: *Lastiwka et al. IJNMF (2005)*
- Particle splitting: *Feldman and Bonet IJNME (2007)*, *Lopez et al. Comput Mech (2013)*,
- Splitting and coalescing: *Barcarolo et al. JCP (2014)*, *Spreng et al. Comp. Part. Mech. (2014)*

Static particle distribution with different mass

- 3D simulations of energy device (Manchester Bobber) under extreme wave conditions



Numerical model	Uniform particle distrib	Variable mass ratio 1:8
# of particles	918'000	139'000
Computational time	7 days	1.5 days
Δx max (m)	0.02	0.04
Δx min (m)	0.02	0.02

Omidvar et al. IJNMF (2012)

WC-SPH variationally consistent scheme for adaptivity

WC-SPH formulation with variable h

Vacondio et al. 2013 CMAME:

- It is variationally derived
- It conserves both mass and momentum

$$\frac{d\rho_i}{dt} = \sum_j m_j (\mathbf{u}_i - \mathbf{u}_j) \cdot \nabla W_j(\mathbf{x}_i, h_j) + 2\delta h_i \sum_j m_j \bar{c}_{ij} \left(\frac{\rho_i}{\rho_j} - 1 \right) \frac{\mathbf{r}_{ij}}{\mathbf{r}_{ij}^2 + \eta^2} \cdot \nabla W_j(\mathbf{x}_i, h_j)$$

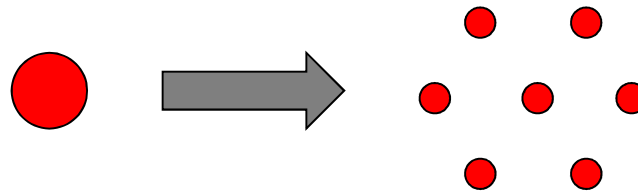
$$\frac{d\mathbf{v}_i}{dt} = \sum_j \frac{m_j}{\rho_j \rho_i} \left[p_i \nabla W_j(\mathbf{x}_i, h_j) - p_j \nabla W_i(\mathbf{x}_j, h_i) \right] + \sum_j \frac{m_j}{\rho_j \rho_i} (\Pi_{ij} \cdot \nabla W_j(\mathbf{x}_i, h_j)) + \mathbf{g}$$

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{v}_i$$

$$p = B \left[\left(\frac{\rho}{\rho_o} \right)^\gamma + 1 \right]$$

Time integration with Symplectic scheme, Wendland kernel, δ – SPH

Increasing the resolution: particle splitting



Splitting procedure (1)

Key idea: split one particle into M daughter particles.

- **Mass, position, velocity, density and smoothing length** must be defined for each daughter particle
- Mass, momentum and energy conservation should be enforced



- Number of daughter particles: ideal numbers is 4 in 2D (it doubles the resolution) but it is not very convenient (see later)
- to reduce the degrees of freedom: we defined a priori the stencil and the smoothing length of the daughter particles
- the mass distribution of the daughter particles is obtained by minimizing the density error

Feldman and Bonet IJNME (2007), Vacondio et al. IJNMF (2012)

Splitting procedure (2)

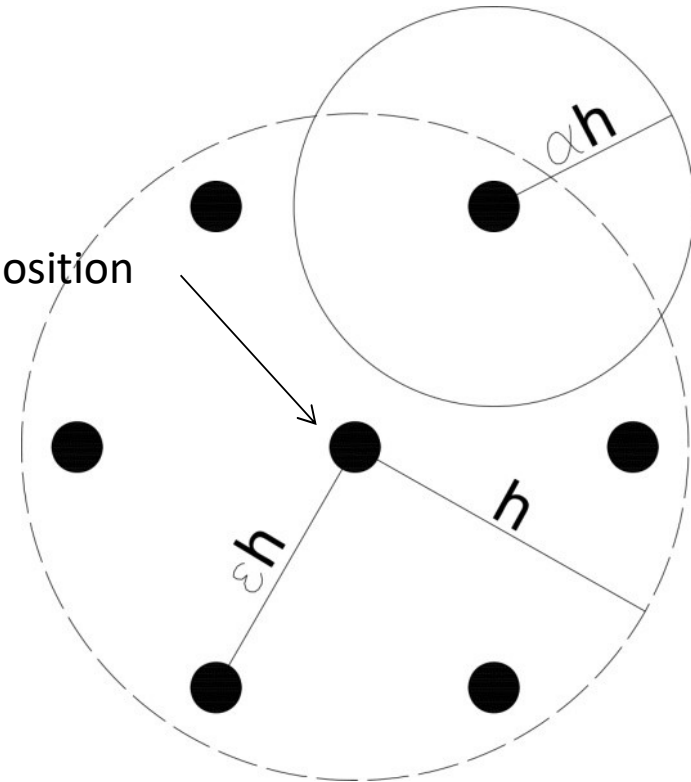
Particle positions and smoothing length are defined as.

$$h_k = \alpha h$$

$$d_k = \varepsilon h$$

$$m_k = \lambda_k m$$

Original particle position



Where α and ε are parameters

ε : particle position

α : smoothing length

Feldman and Bonet IJNME (2007), Vacondio et al. IJNMF (2011)

Density error minimization

Local density error:
$$e(\mathbf{x}) = \rho(\mathbf{x}) - \rho^*(\mathbf{x}) = m_N W_N(x, h_N) - \sum_{k=1}^M m_k^* W_k(x, h_k)$$

Global density error:
$$E = \int_{\Omega} e(\mathbf{x})^2 d\mathbf{x}$$

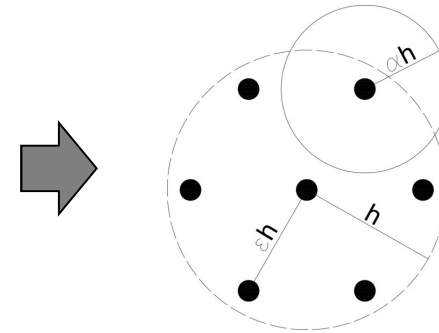
After some algebra ... the best mass distribution is calculated as follows:

$$E^* = \min_{\lambda} \left\{ \bar{C} - 2\boldsymbol{\lambda}^T \bar{\mathbf{b}} + \boldsymbol{\lambda}^T \mathbf{Q} \boldsymbol{\lambda} \right\} \quad \lambda_k = \frac{m_k}{m_N}$$

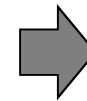
With constraint for mass conservation:
$$\sum_{j=1}^M \lambda_j = 1$$

State of the art of splitting in SPH

To dynamically vary the resolution in 2D: splitting and coalescing procedures are available

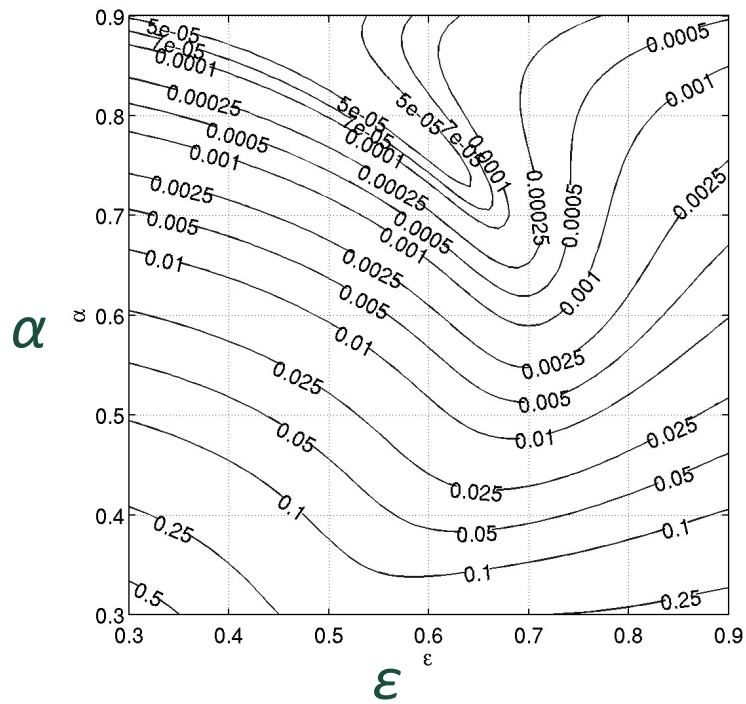
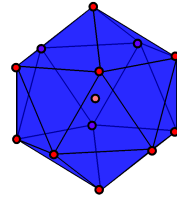


in 3D no literature available on splitting

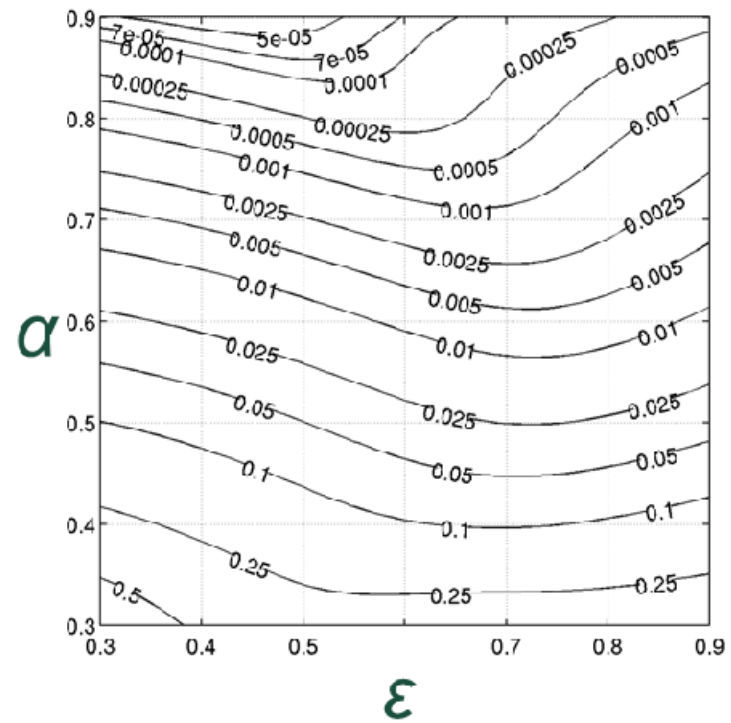
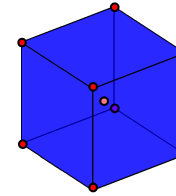


Global density error

Icosahedron



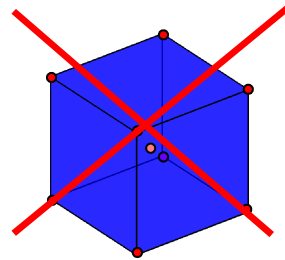
Cube



Which is the best stencil?

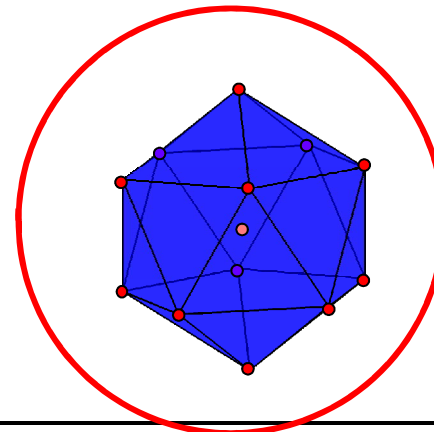
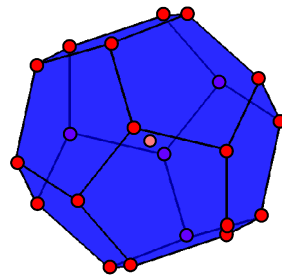
Cube

It is not the best stencil: error small only for $h_k=0.9 h_M$ This means a lot of neighbors in the high resolution zone.

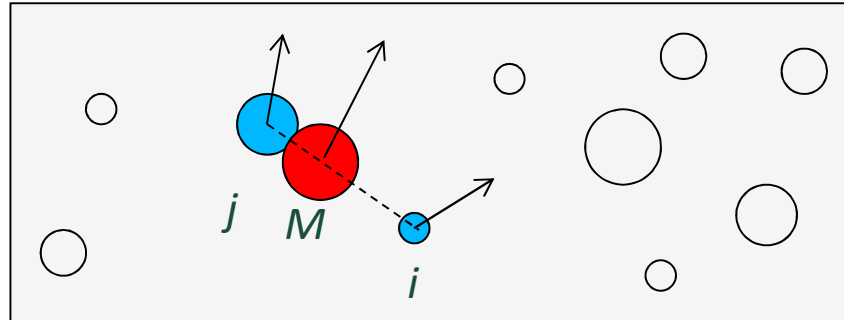


Platonic solids

The global density error matrix obtained for Icosahedron and Dodecahedron are similar, but the **Icosahedron is more efficient** because it creates less daughter particles (12 vertices instead of 20)



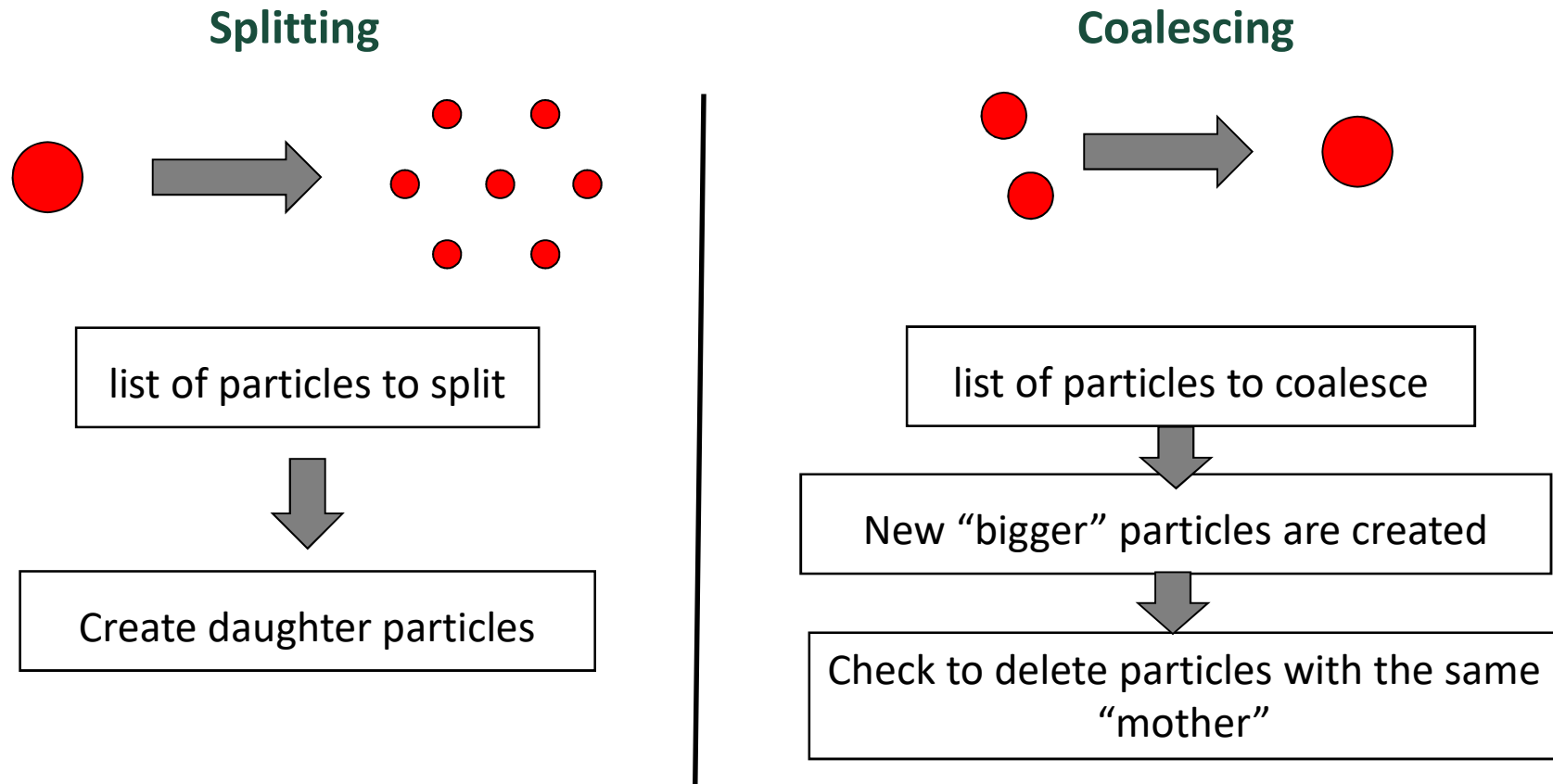
To Reduce the resolution: Particle Coalescing (merging)



The same algorithm used in 2D and 3D (*Vacondio et al. 2013 CMAME*):

- Particles are coalesced in pairs
- mass and momentum conservation gives mass position and velocity of the new particle M
- The smoothing length h_M is obtained by enforcing zero density error
- No further coalescing is possible for particle M in the same time iteration

Parallel implementation (CPU & GPU)



Variable res. formulation overheads: h and m different for each particle, more memory access, and more floating point operation

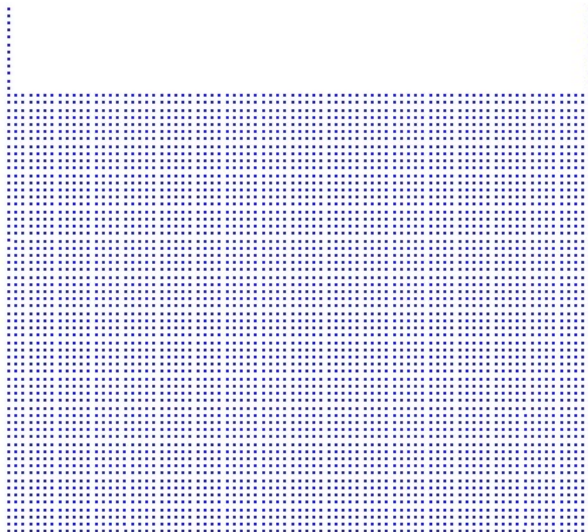
2-D still water tank

$\Delta x_0 = 0.025$ m, ($N_p = 4800$)

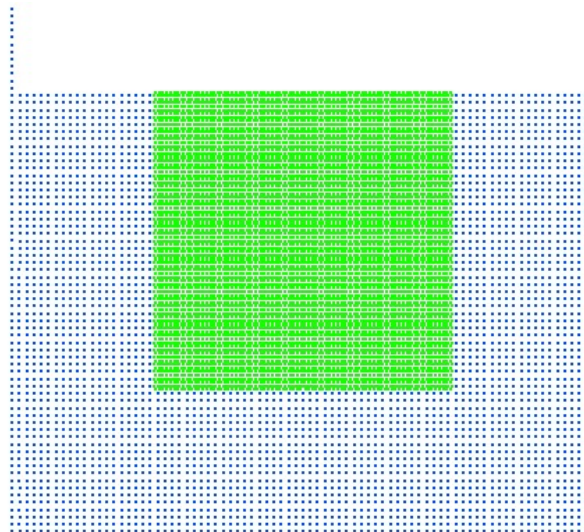
size of the box 2x1.5 m

Low artificial viscosity: $\alpha = 0.01$

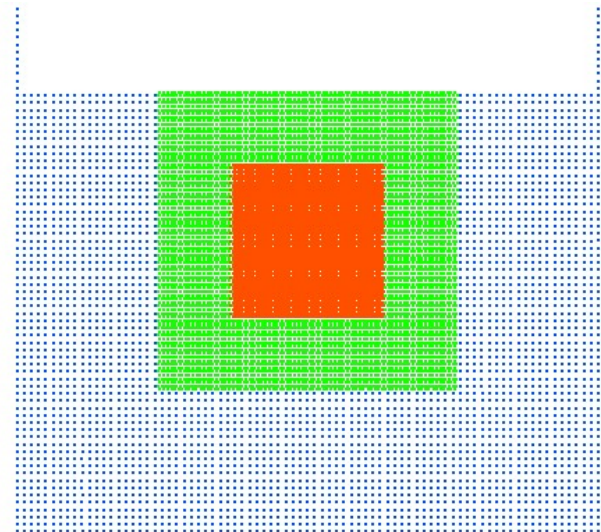
uniform resolution



one level of splitting



two levels of splitting



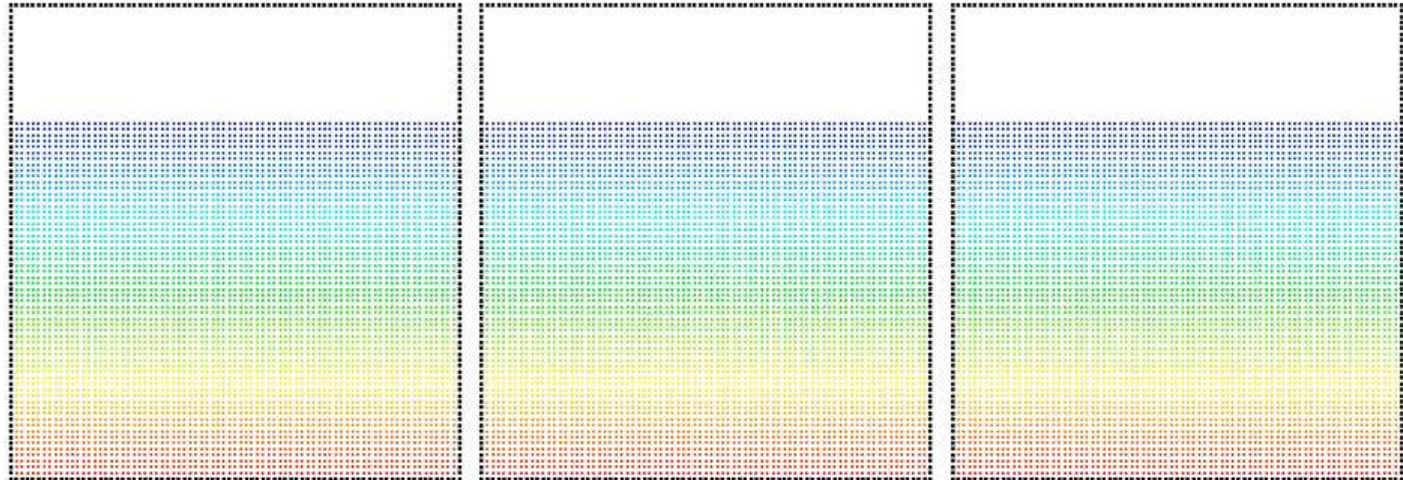
Pressure field

uniform resolution

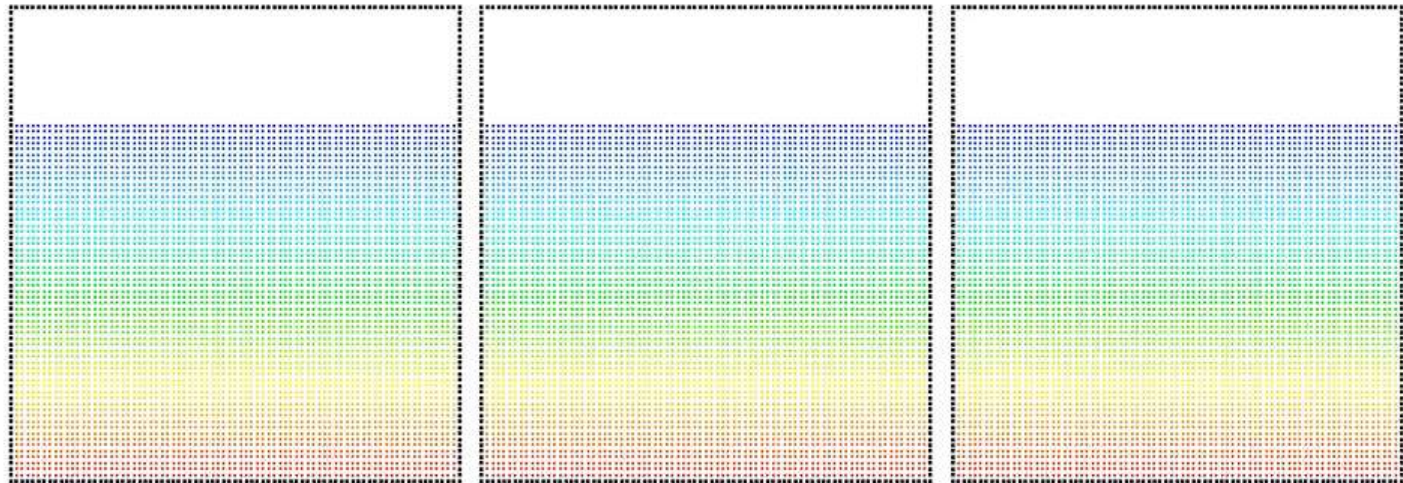
one level of splitting

two levels of splitting

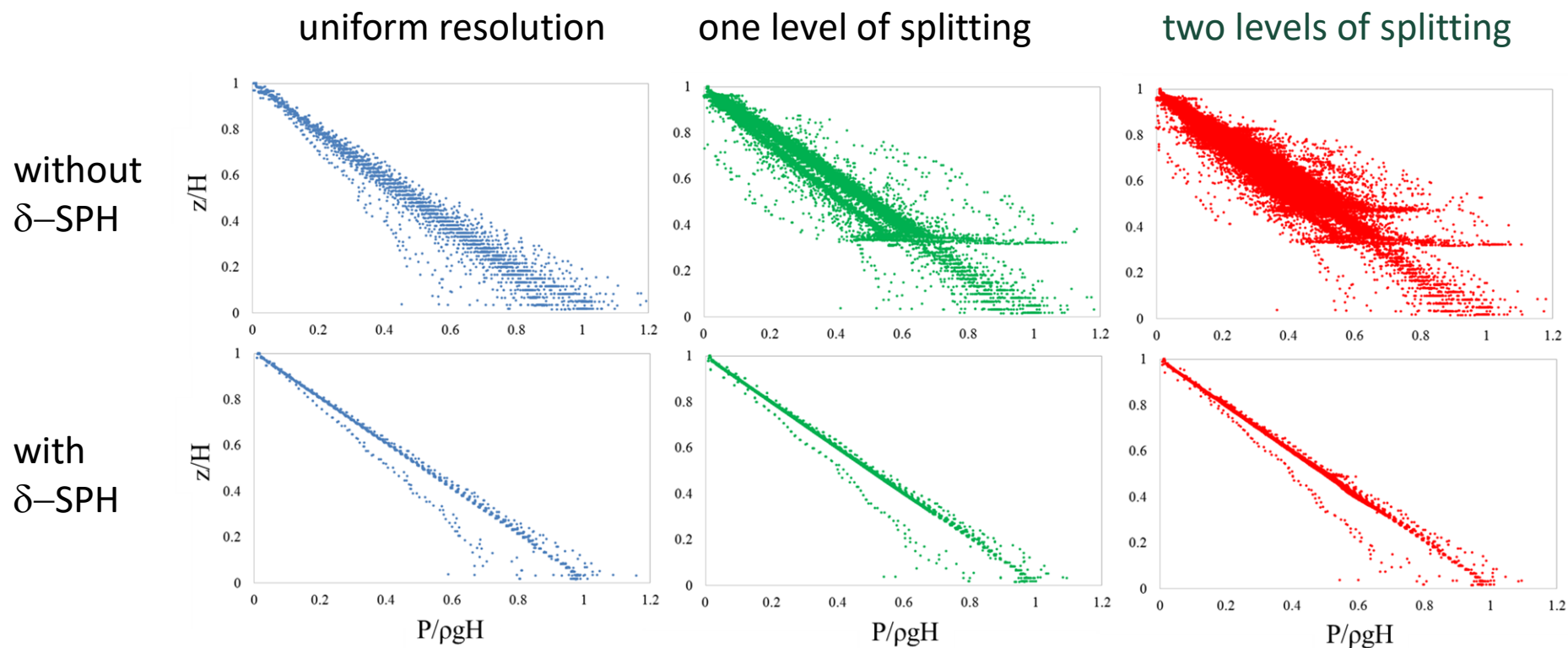
without
 δ -SPH



with
 δ -SPH



Vertical distribution of pressure at last instant ($t=5s$, after 54k steps)



2D-Falling sphere

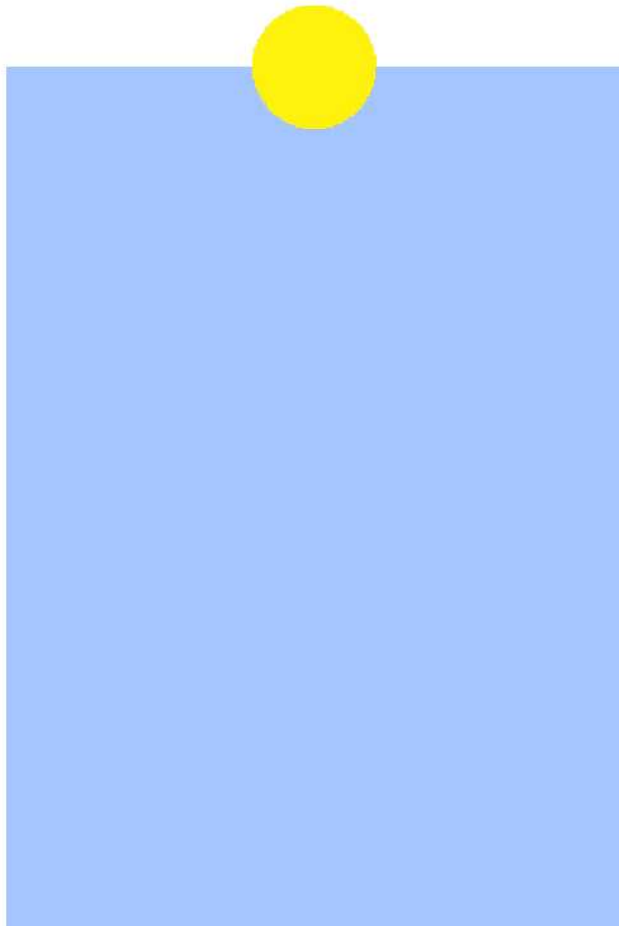


- 2-D sphere with radius=1 m
- density=1,200 kg/m³
- be compared against VOF: Fekken (2004)
- SPH with $\Delta x_0=0.03$ m and no adaptivity
- SPH with $\Delta x_0=0.05$ m and **dynamic** adaptivity

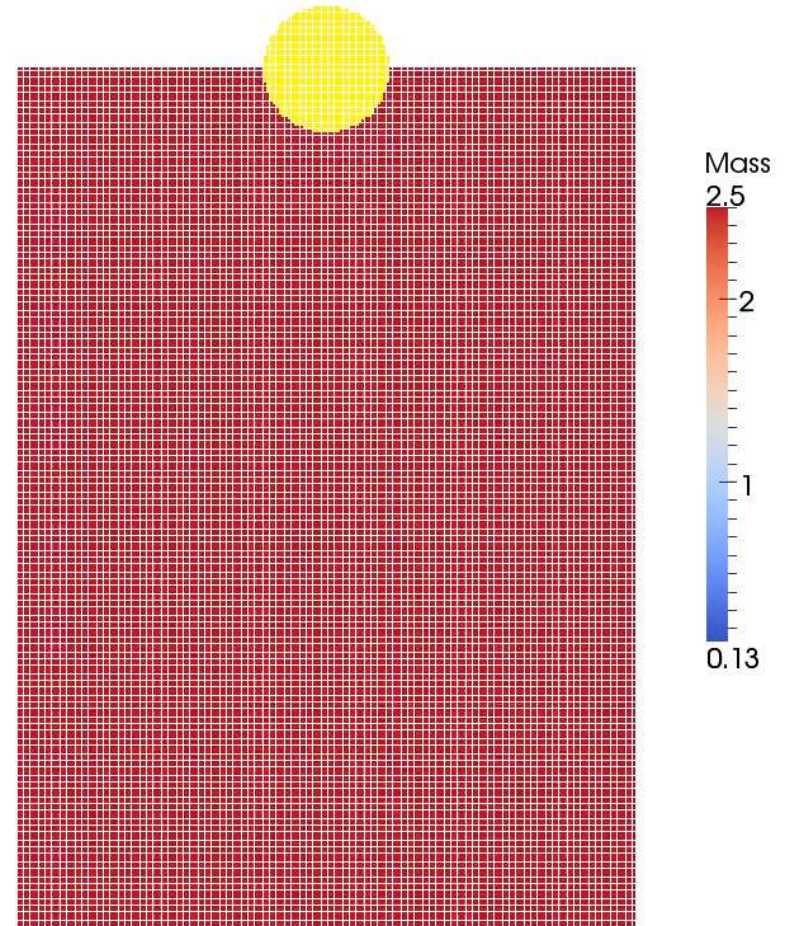
**dynamic adaptive region
is used**

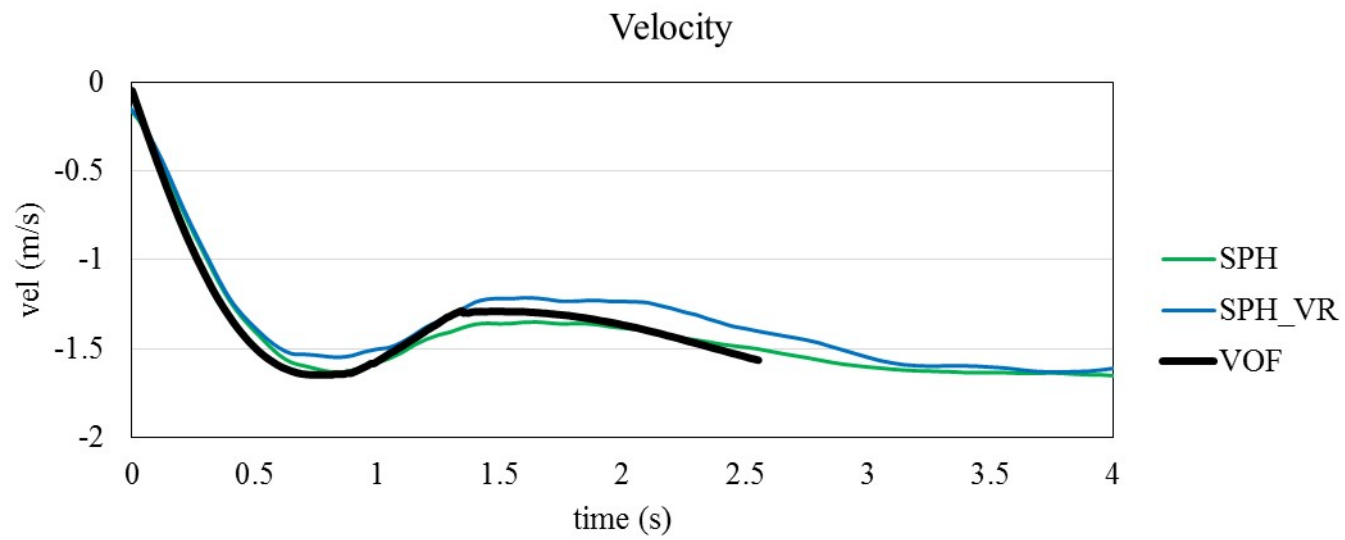
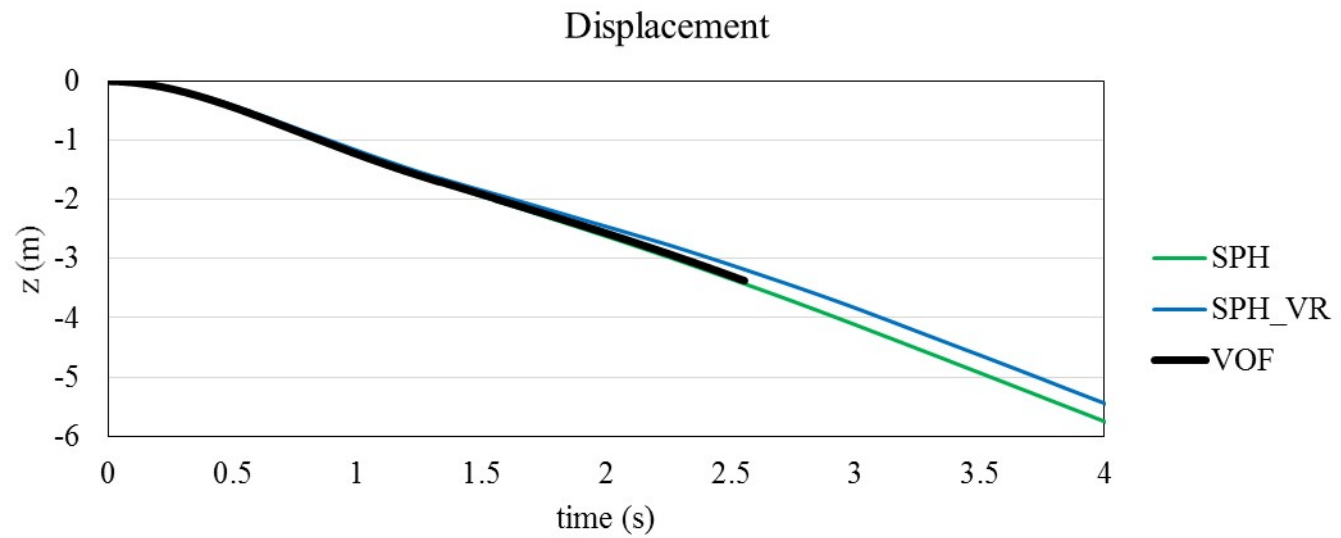
Fekken G. Numerical simulation of free surface flow with moving rigid bodies, Ph.D. Thesis, University of Groningen, 2004

High resolution $\Delta x_0=0.03$ m no adaptivity



Low Res $\Delta x_0=0.05$ m dynamic adaptivity

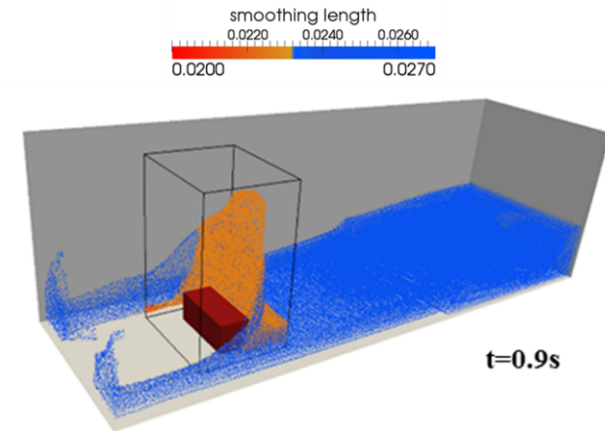




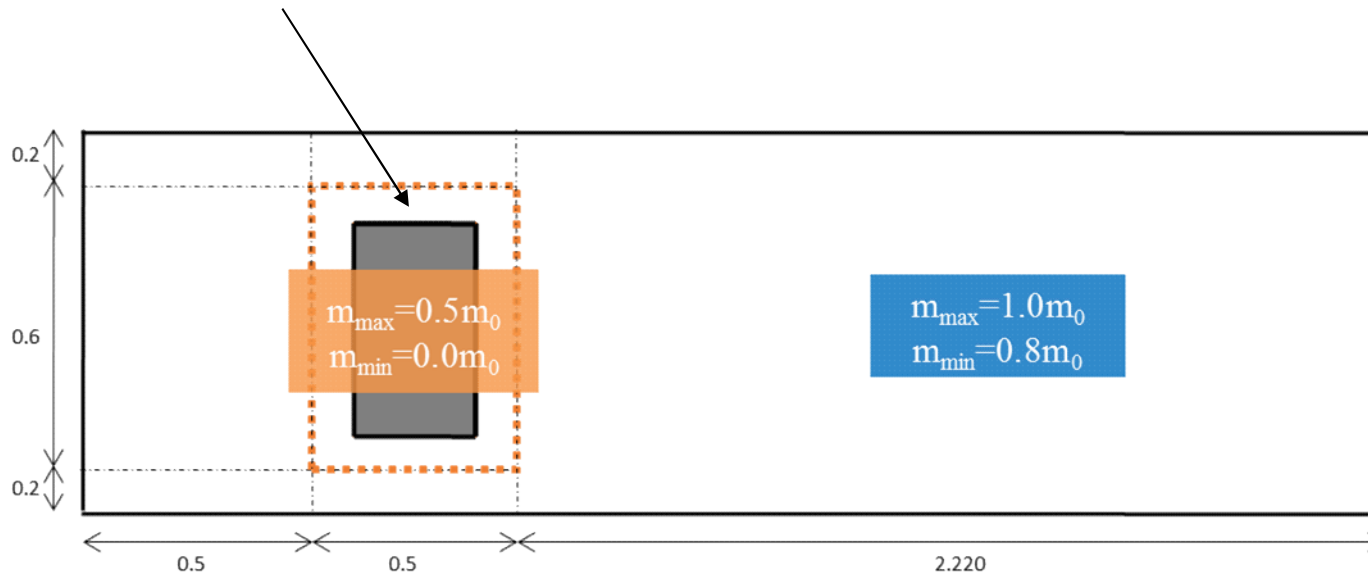
SPHERIC Benchmark Case #2.

Two simulations:

- No adaptivity, $\Delta x_0 = 0.008\text{m}$
- Adaptivity (splitting and coalescing) $\Delta x_0 = 0.015\text{m}$

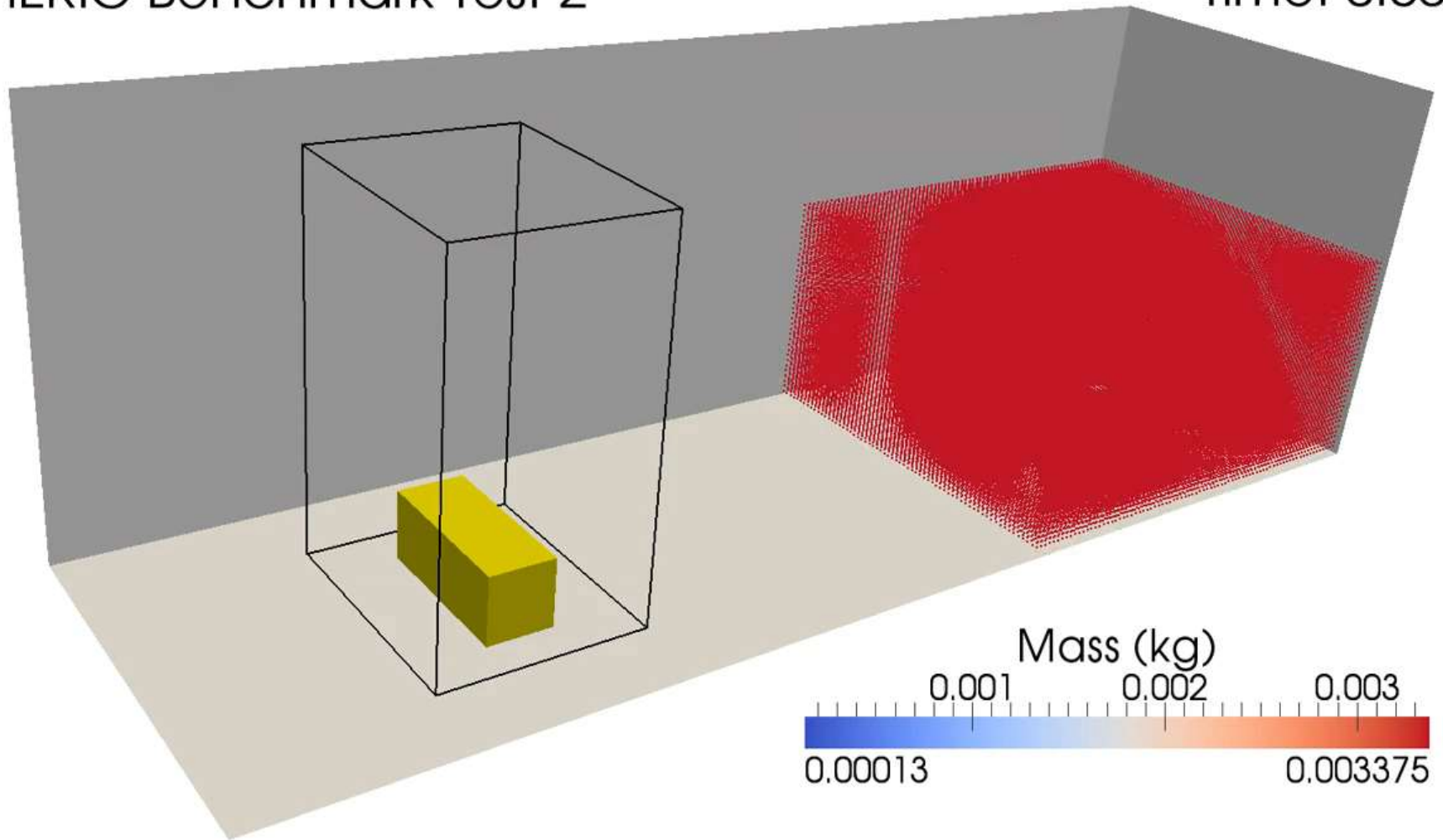


High resolution region



SPHERIC Benchmark Test 2

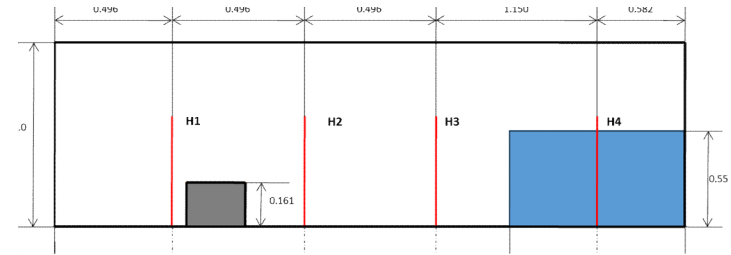
Time: 0.00 s



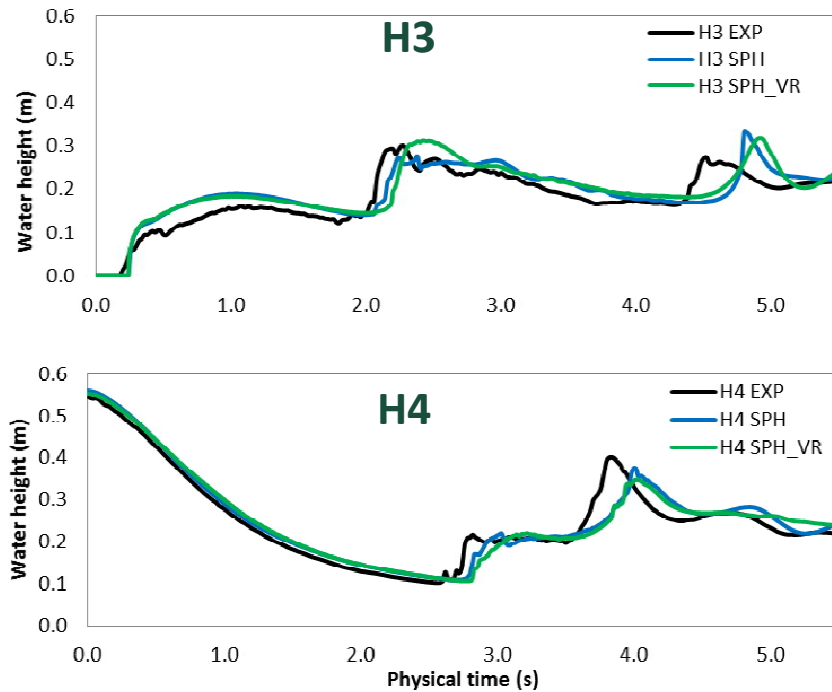
SPHERIC Benchmark Case #2.

Two simulations:

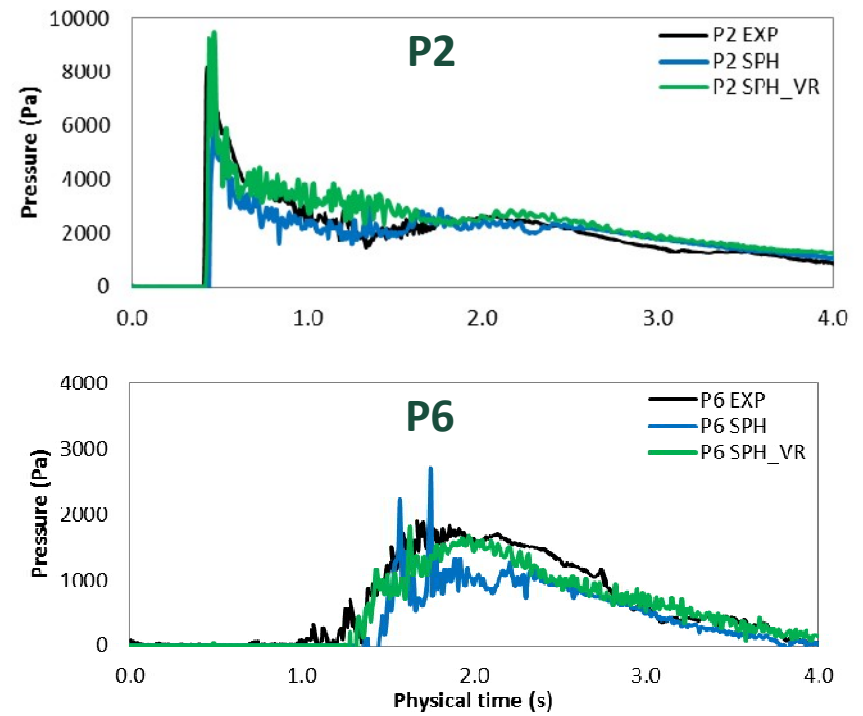
- no adaptivity: blue
- Adaptivity: green



Water height



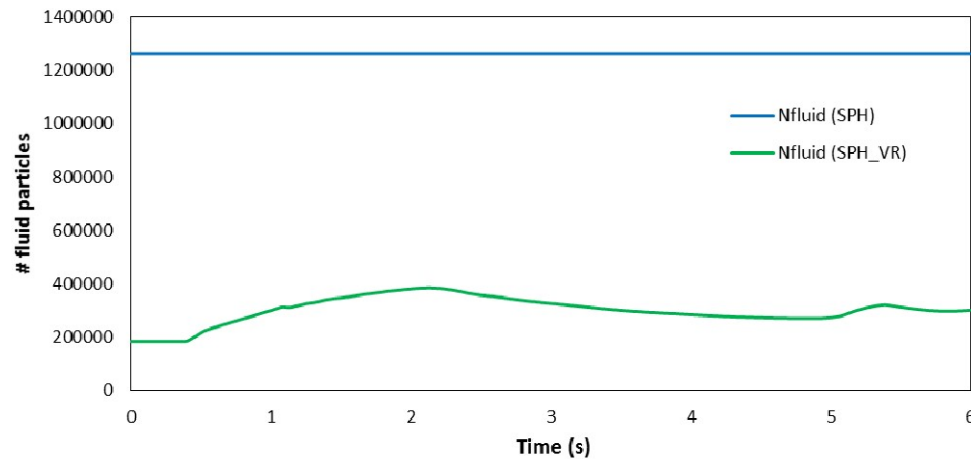
Pressure



SPHERIC Benchmark Case #2.

	SPH	SPH-adaptivity
Δx_0	0.008m	0.015m
Initial number of particles	$1.3 \cdot 10^6$	184 k
New daughters by splitting	-	$2.2 \cdot 10^6$
Coalesced particles	-	$3.9 \cdot 10^6$
CPU runtime	173 h	85 h
GPU runtime	3.60h	1.99h

of fluid particles



CPU speedup: 2.04

GPU speedup: 1.80

This looks bad but:

$$\frac{(\# \text{ part SPH})}{(\# \text{ part SPH-adapt})} = 4.2$$

Issue: in complex problem we don't know a priori where high resolution is necessary

Idea: control spatial resolution using a measure of the SPH interpolation error.

Yes, but what is a "good" measure of the SPH interpolation error?



L. Da Vinci Sistine Chapel

Taylor expansion of a scalar function f

$$f(\mathbf{x}) = f_i + \nabla f_i \cdot (\mathbf{x} - \mathbf{x}_i) + O(\mathbf{x} - \mathbf{x}_i)^2$$

Multiplying this by the kernel function $W_i(\mathbf{x})$ and integrating we obtain:

$$\int f(\mathbf{x})W_i d\mathbf{x} = f_i \int W_i d\mathbf{x} + \nabla f_i \cdot \int (\mathbf{x} - \mathbf{x}_i)W_i d\mathbf{x}$$

The SPH approximation is first order consistent if:

$$\int f(\mathbf{x})W_i d\mathbf{x} = f_i \cdot 1 + \nabla f_i \cdot \mathbf{0} + O(\mathbf{x} - \mathbf{x}_i)^2$$

We can repeat the same demonstration multiplying the same Taylor expansion by $\partial_x W_i(\mathbf{x})$

$$\int f(\mathbf{x})\partial_x W_i d\mathbf{x} = f_i \cdot \mathbf{0} + \partial_x f_i \cdot 1 + \partial_y f_i \cdot \mathbf{0} + \partial_z f_i \cdot \mathbf{0} + O(\mathbf{x} - \mathbf{x}_i)^2$$

Liu & Liu App. Num. Math. 2006

Criteria for automatic adaptivity in SPH

An SPH interpolation of an arbitrary field quantity f is First order consistent when:

$$\mathbf{C}_{0,0}(\mathbf{r}_a) = \sum \frac{m_b}{\rho_b} W_{ab} = 1$$

$$\mathbf{C}_{0,1}(\mathbf{r}_a) = \sum \frac{m_b}{\rho_b} (\mathbf{r}_a - \mathbf{r}_b) W_{ab} = \mathbf{0}$$

Similarly SPH interpolation for ∇f_i is first order consistent if:

$$\mathbf{C}_{1,0}(\mathbf{r}_a) = \sum \frac{m_b}{\rho_b} \nabla_a W_{ab} = \mathbf{0}$$

$$\mathbf{C}_{1,1}(\mathbf{r}_a) = \sum \frac{m_b}{\rho_b} (\mathbf{r}_a - \mathbf{r}_b) \nabla_a W_{ab} = \mathbf{1}$$

- Those conditions are never fulfilled by a normal SPH interpolation for irregular particle distribution
- The higher is the error on the moments, the less accurate is the SPH interpolation

Liu et al., 2003.

Criteria for automatic adaptivity in SPH

consistency order matrix of SPH interpolation:

$$\mathbf{C}_a = \begin{bmatrix} \mathbf{C}_{0,0}(\mathbf{r}_a) & \mathbf{C}_{1,0}(\mathbf{r}_a) \\ \mathbf{C}_{0,1}(\mathbf{r}_a) & \mathbf{C}_{1,1}(\mathbf{r}_a) \end{bmatrix}$$

If $\mathbf{C}_a = \mathbf{I}$, then SPH interpolation
first-order accurate



'error approximation'

$$\varepsilon_a^A = |\det(\mathbf{C}_a) - 1|$$



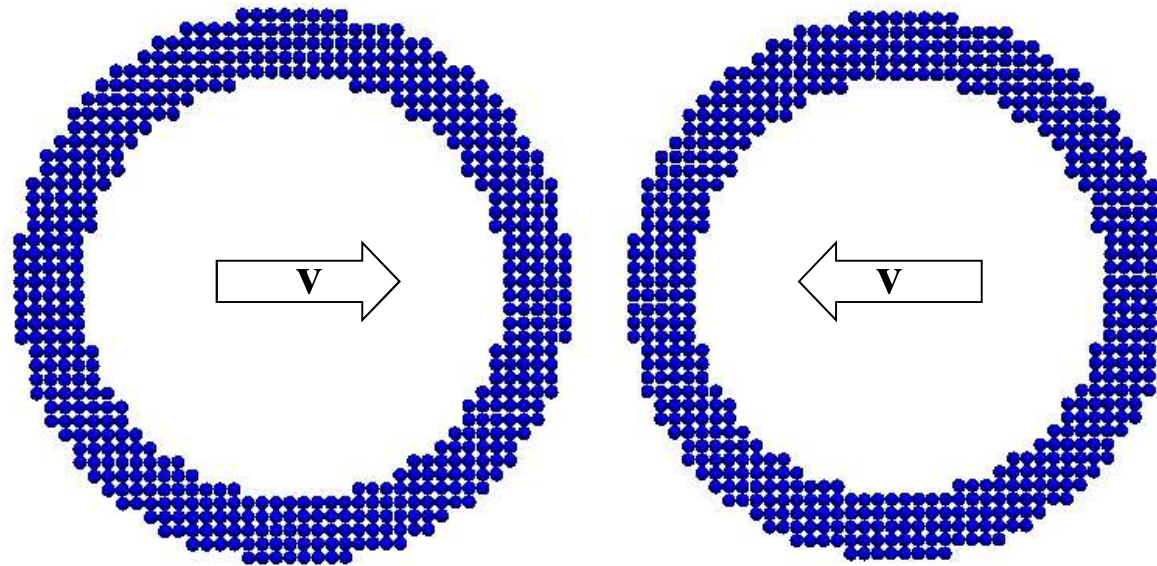
error-approximation-based adaptivity criterion

$$\eta_a^A |_{t+\Delta t} = \left| \frac{\varepsilon_a^A}{\bar{\varepsilon}^A} |_{t+\Delta t} - \frac{\varepsilon_a^A}{\bar{\varepsilon}^A} |_t \right| \quad \text{with} \quad \bar{\varepsilon}^A = \frac{1}{n} \sum_{k=0}^n \varepsilon_k^A$$

Sprengr et al., 2016 SPHERIC Proceedings.

Colliding Rubber Rings

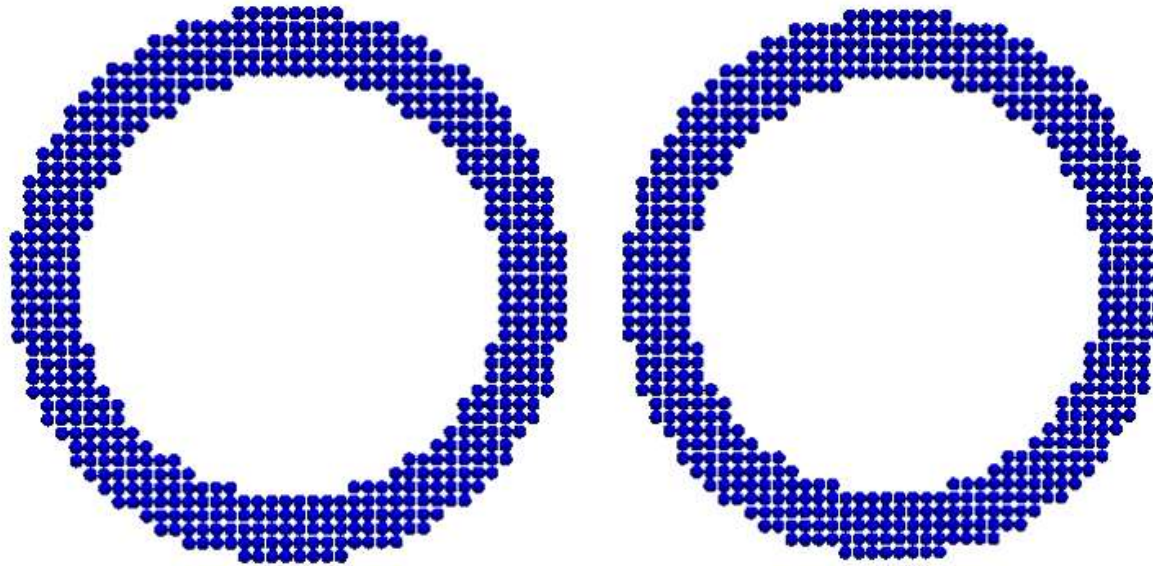
- solid mechanics simulation
- highly dynamic character, large deformations
- **static resolution** (approximately 1,100 particles)



Colliding Rubber Rings

relative approximated error

$$\eta_a^A |_{t+\Delta t} = \left| \frac{\varepsilon_a^A}{\varepsilon^A} |_{t+\Delta t} - \frac{\varepsilon_a^A}{\varepsilon^A} |_t \right|$$



● original particle ● refined particle

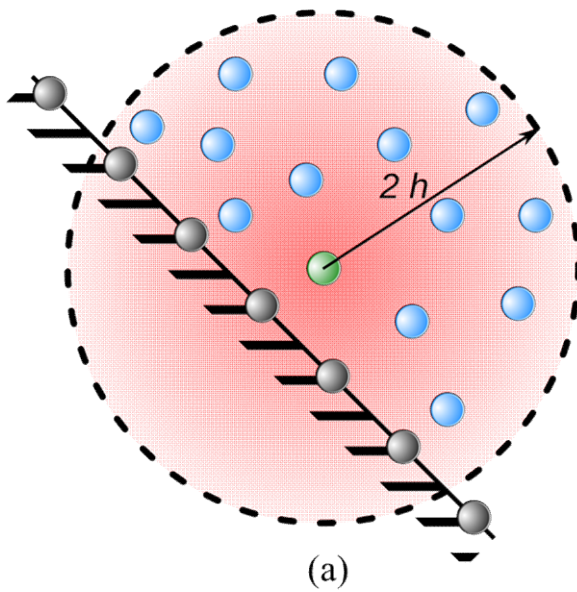
Conclusion on variable resolution

- Open Source Parallel SPH code with variable resolution and adaptivity has been presented
- Both 2D and 3D
- OpenMP and CUDA versions of the code have been developed, Speedup / overheads have been discussed
- Code validated in 2D and 3D against experiments and numerical simulation
- Formulation is adapted for particles with different size with negligible errors at interface between different resolution
- A novel criteria to automatically adjust the resolution have been presented

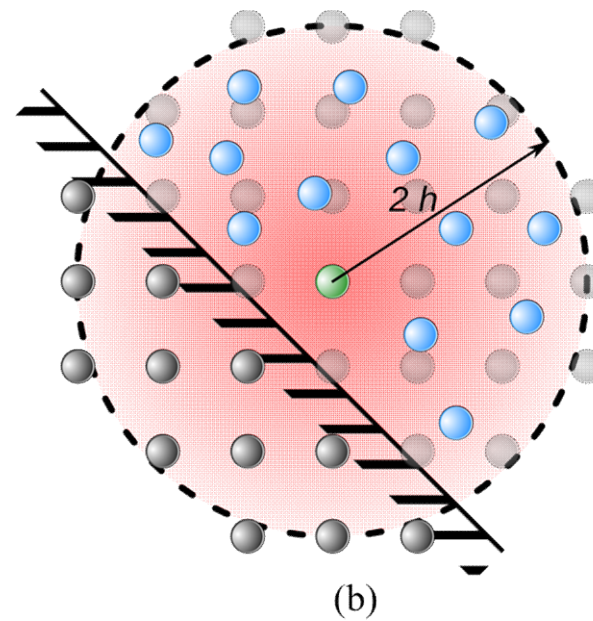
Wall Boundary conditions

Different type of Boundary Conditions

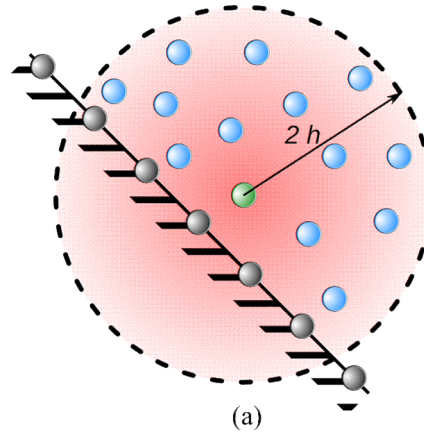
Dynamic boundaries (DBC)



Local Uniform STencil (LUST)



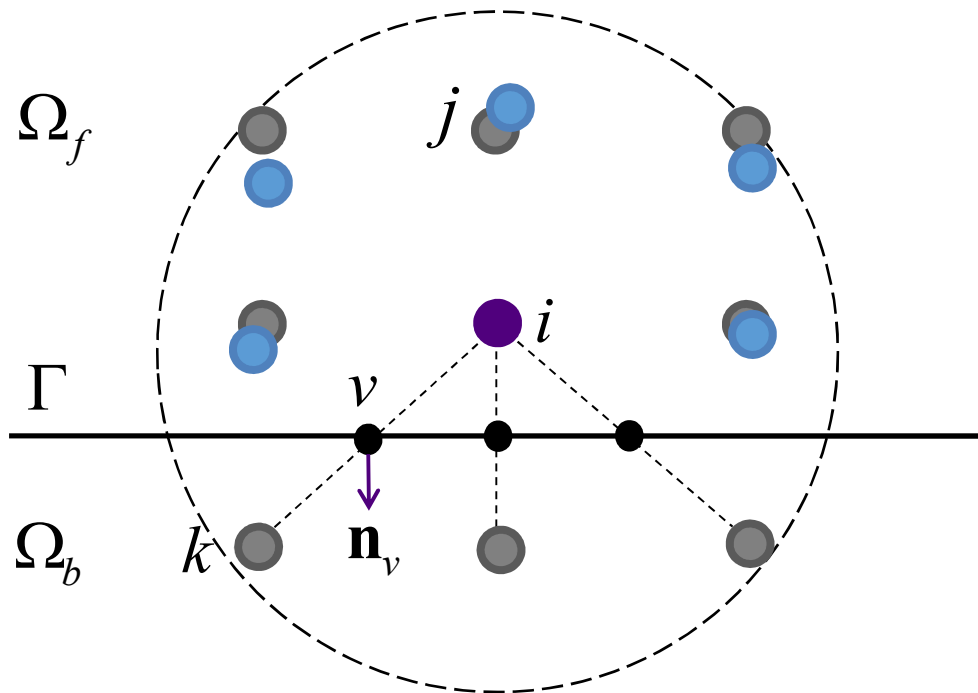
Dynamic boundaries (DBC)



- same continuity equation as for the fluid particles
- computationally efficient
- Kernel truncation error which prevents convergence
- Over repulsion of fluid particles
- Non-slip boundary condition cannot be imposed

$$\frac{d\rho_i}{dt} = \sum_j \rho_j \frac{m_j}{\rho_j} \mathbf{v}_{ij} \cdot \nabla_i W_{ij}$$

Local Uniform Stencil (LUST) Concept



- Regular stencil of fictitious particles is centered around fluid particles

- Fictitious particles in the fluid domain are deleted.

- The remaining fictitious particles, are used to solve cont. and momentum equations

- No kernel truncation
- It can deal with complex boundary
- Approximately first order consistent
- **Resolution independent (ideal for variable resolution)**

Local Uniform Stencil (LUST)

The density of the fictitious particles is corrected hydrostatically based on the density of the fluid particle.

$$\rho_k = \rho_i + \left[\rho_0 \sqrt[7]{\frac{\rho_0 g_z \mathbf{x}_{ik} \cdot \mathbf{n}_v}{B} + 1} - \rho_0 \right],$$

The pressure is then evaluated through the EOS.

The velocities of the fictitious particles are assigned according to Takeda et al.'s anti-symmetric mirroring formulation.

$$\mathbf{u}_k = (\mathbf{u}_i - \mathbf{u}_v) \frac{\mathbf{x}_{vk} \cdot \mathbf{n}_v}{\mathbf{x}_{iv} \cdot \mathbf{n}_v} - \mathbf{u}_v.$$

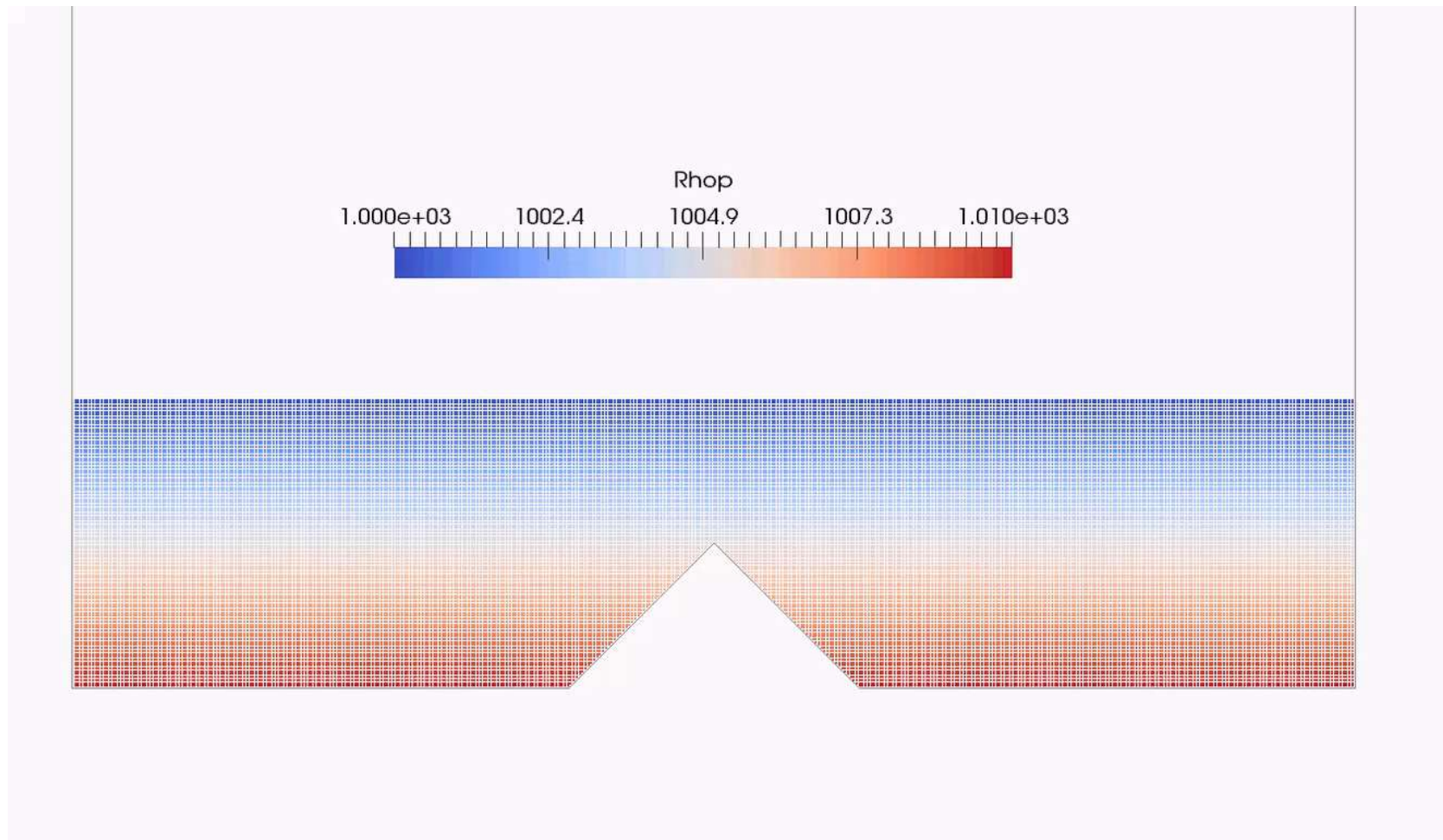
Momentum equation

$$\left\langle \frac{d\mathbf{u}}{dt} \right\rangle_i = - \sum_{j \in \Omega_f} m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} + \Pi_{ij} \right) \nabla W_{ij} - \sum_{k \in \Omega_b} m_k \left(\frac{P_i}{\rho_i^2} + \frac{P_k}{\rho_k^2} + \Pi_{ik} \right) \nabla W_{ik},$$

Continuity equation

$$\left\langle \frac{d\rho}{dt} \right\rangle_i = \sum_{j \in \Omega_f} m_j (\mathbf{u}_i - \mathbf{u}_j) \cdot \nabla W_{ij} + \sum_{k \in \Omega_b} m_k (\mathbf{u}_i - \mathbf{u}_k) \cdot \nabla W_{ik}.$$

Still water - pressure



$\Delta x_0 = 0.005$ m , $h = 1.3 \Delta x_0$

$N_p = 40'000$

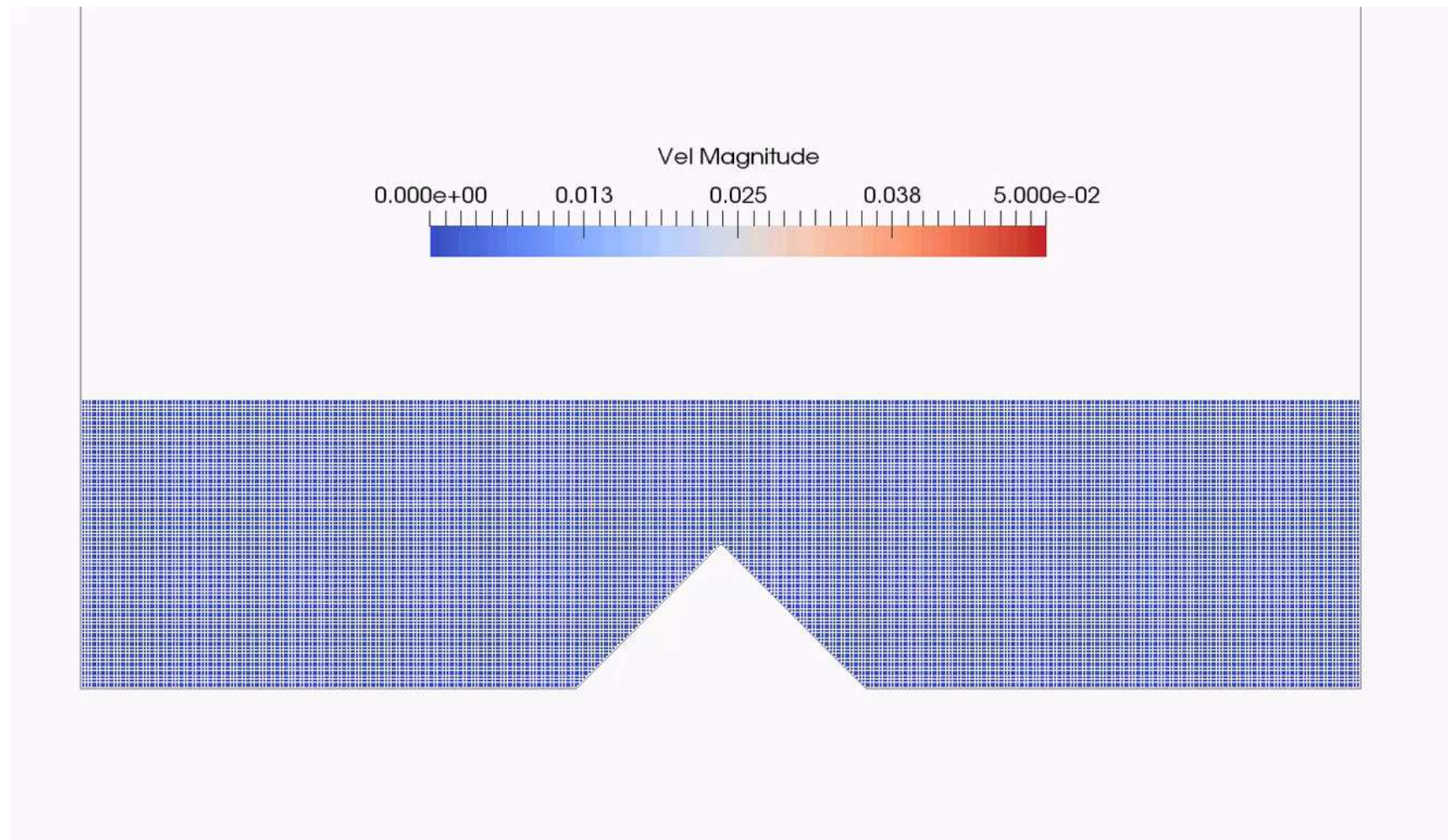
Size: $h = 0.5$ m $L = 2.2$ m

Laminar viscosity = 0.001 m²/s

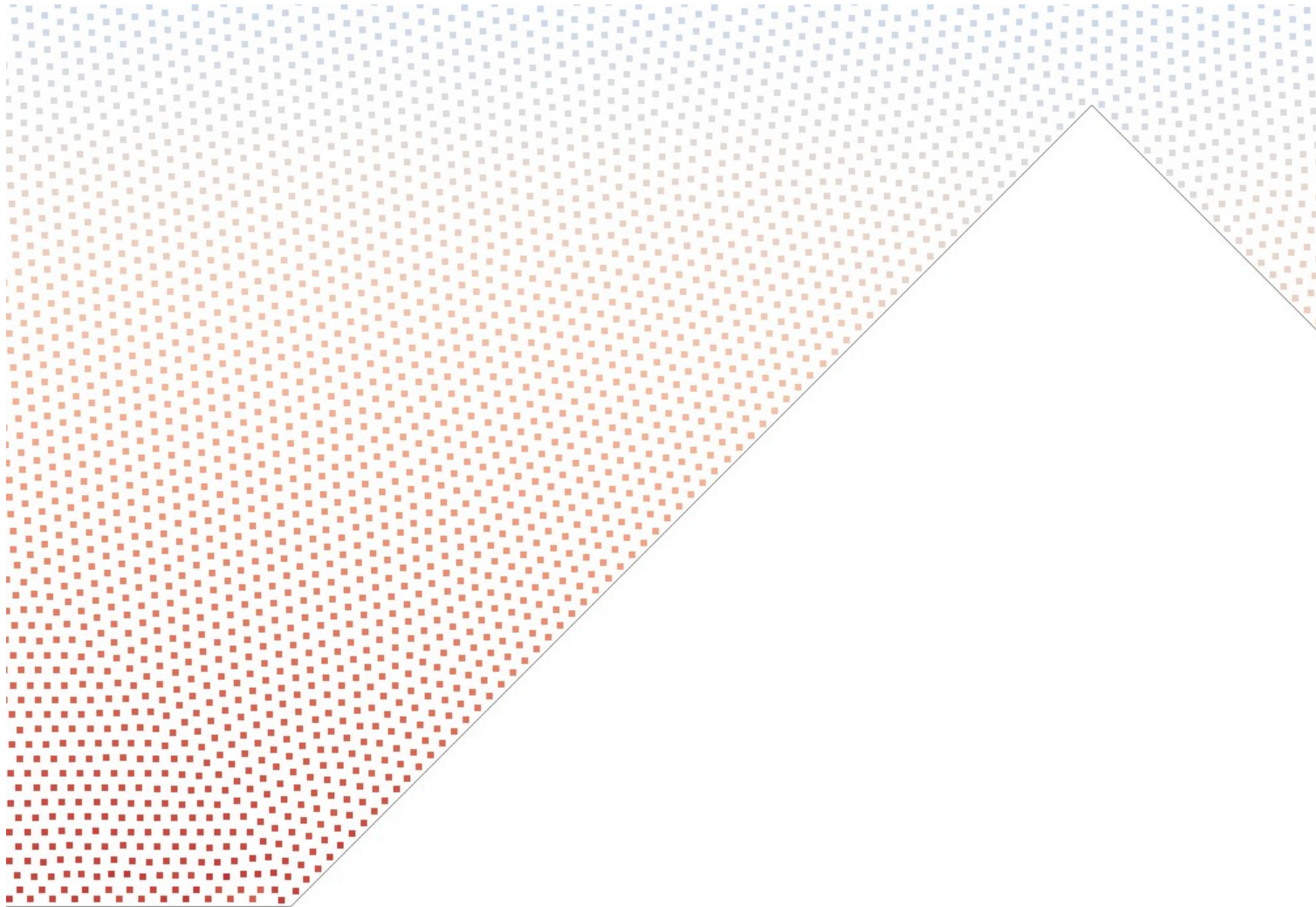
Total time = 10 s

170'000 time steps

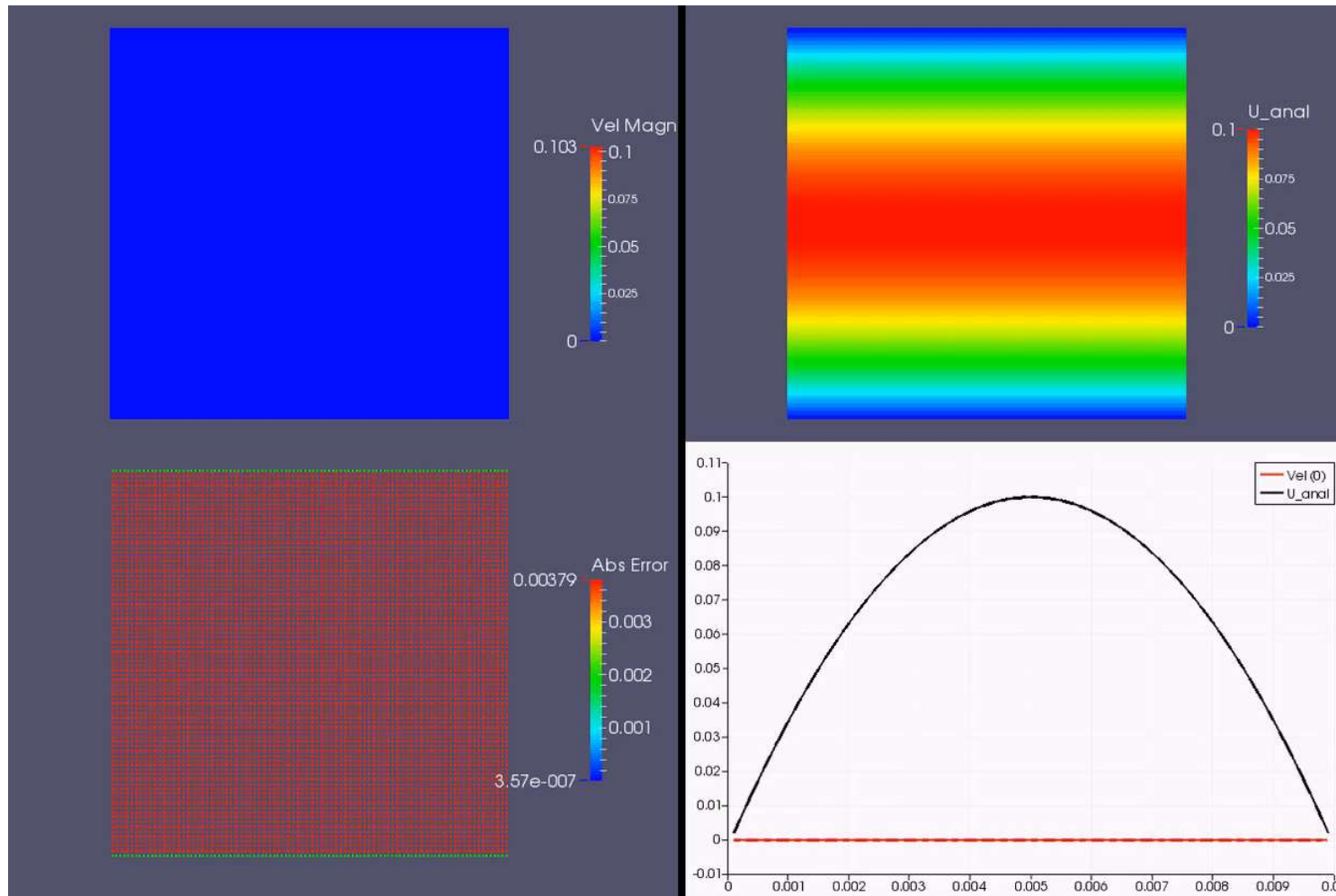
Still water - velocity



Still water – particle distribution zoom



Poiseuille flow in 2D



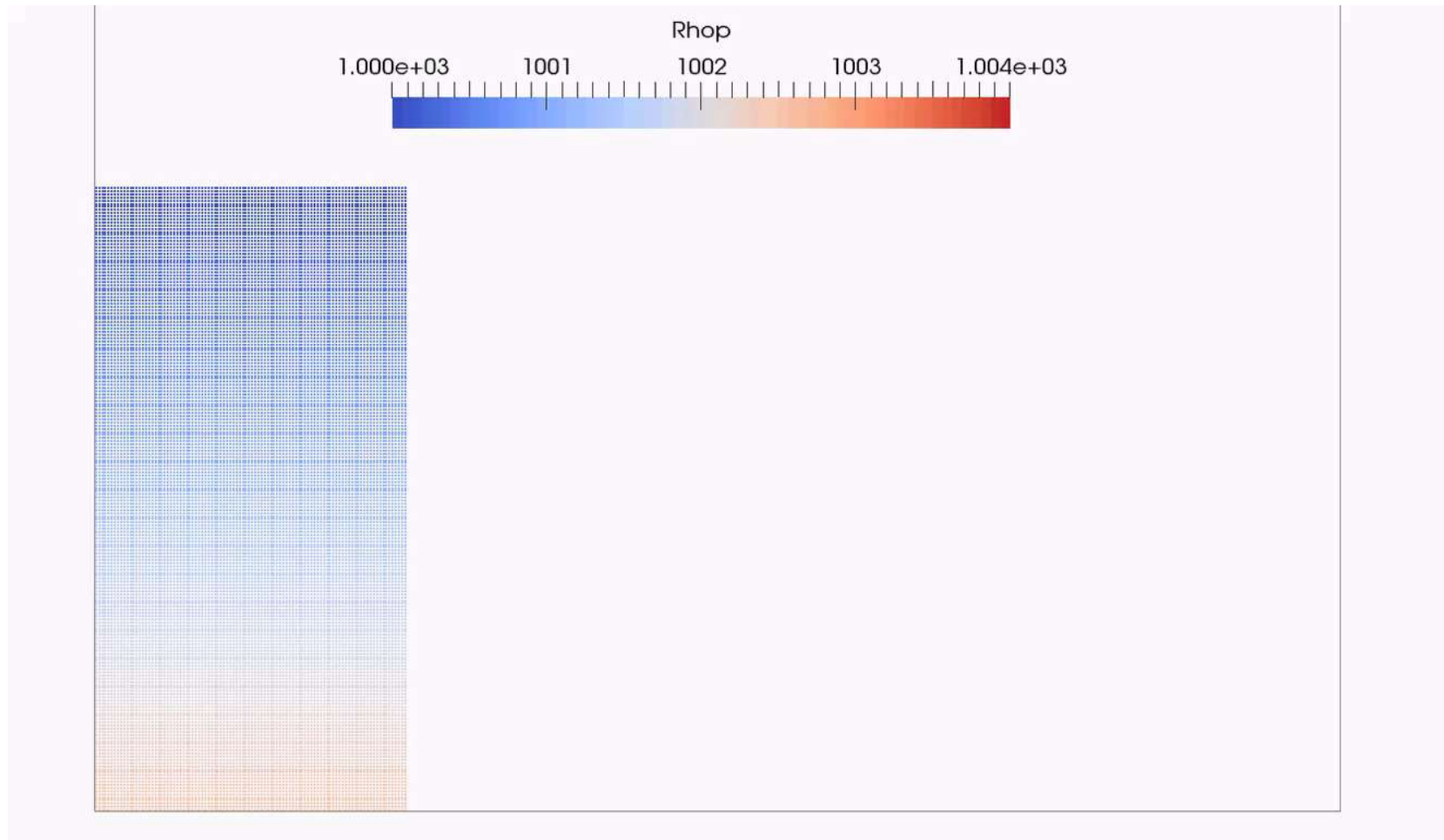
$\Delta x_0 = 0.0001 \text{ m}$, $h = 1.3 \Delta x_0$

$N_p = 10'000$

$Re = 10$

Total time = 10 s (1M time steps)

Dam Break (first attempt)



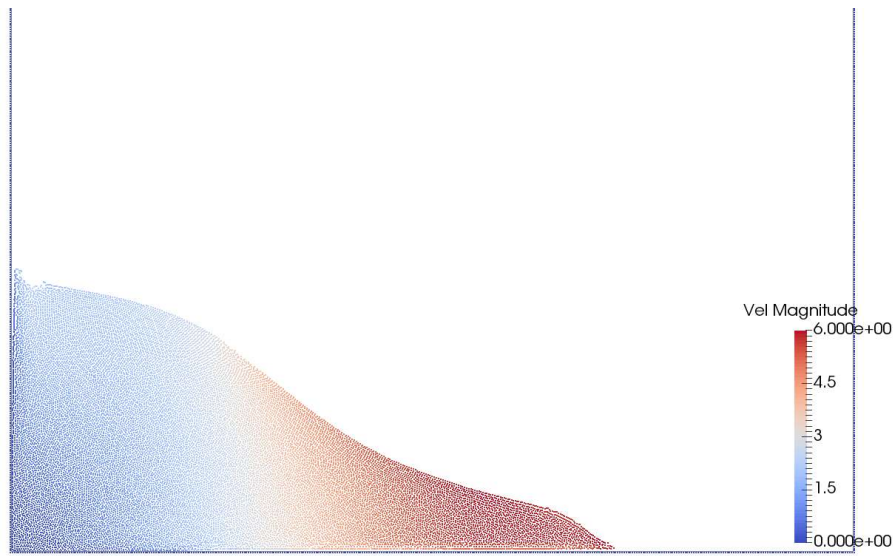
$\Delta x_0 = 0.01 \text{ m}$, $h = 1.3 \Delta x_0$

$N_p = 20'000$

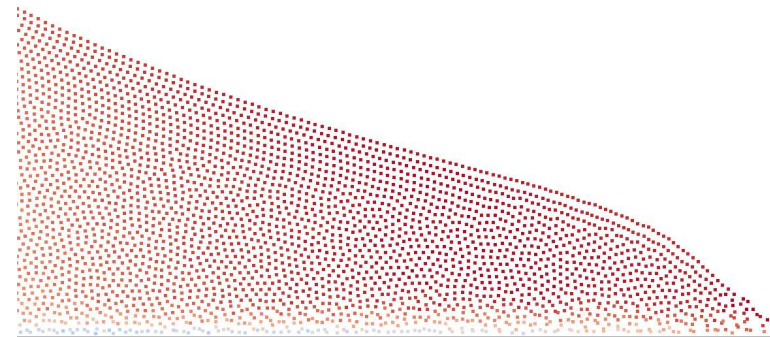
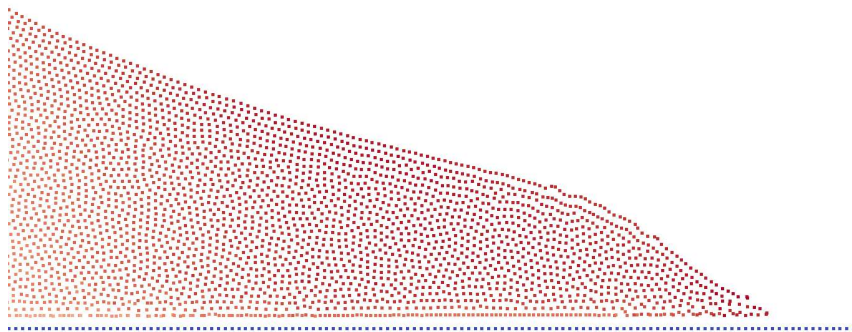
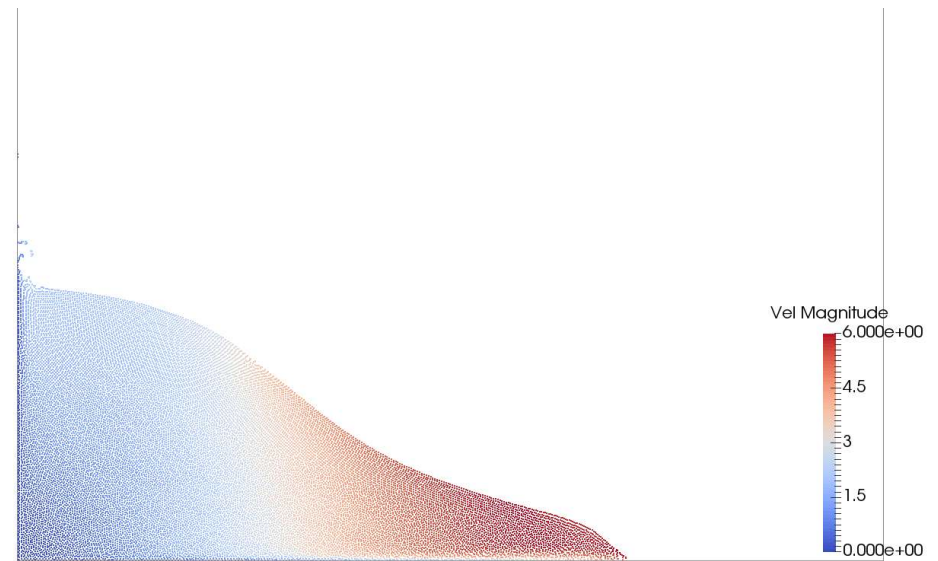
Laminar viscosity = $0.001 \text{ m}^2/\text{s}$

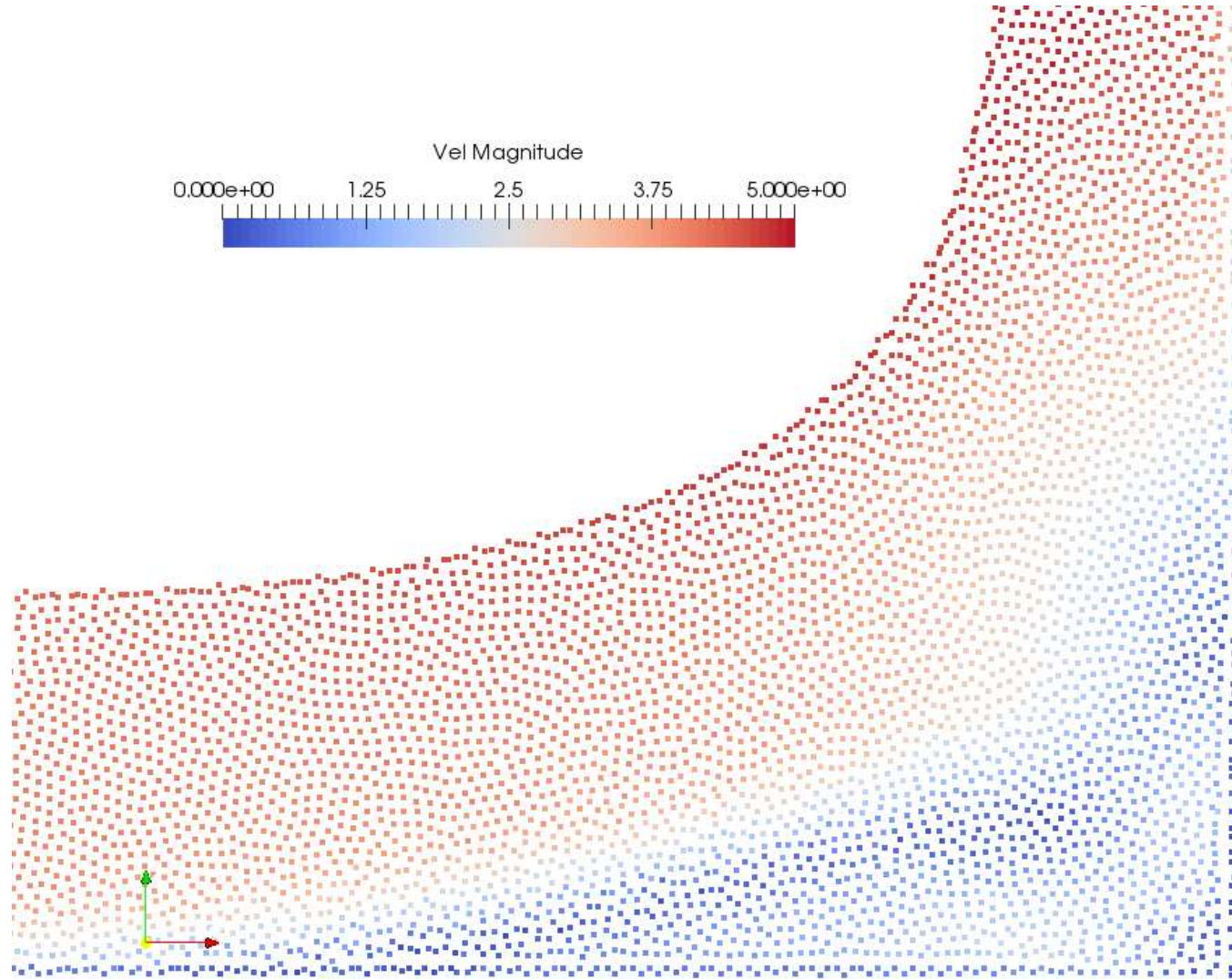
Dam Break (first attempt)

Dynamic Boundaries



LUST





Open Boundary conditions

State of the Art of Open Boundary Conditions in SPH (GC #3)

Two Approaches:

- Buffer layers (see Lastiwka et al., 2009, Federico et al., 2012)
- Semi-analytical boundary conditions method (see Kassiotis et al. 2013, Leroy, 2014)

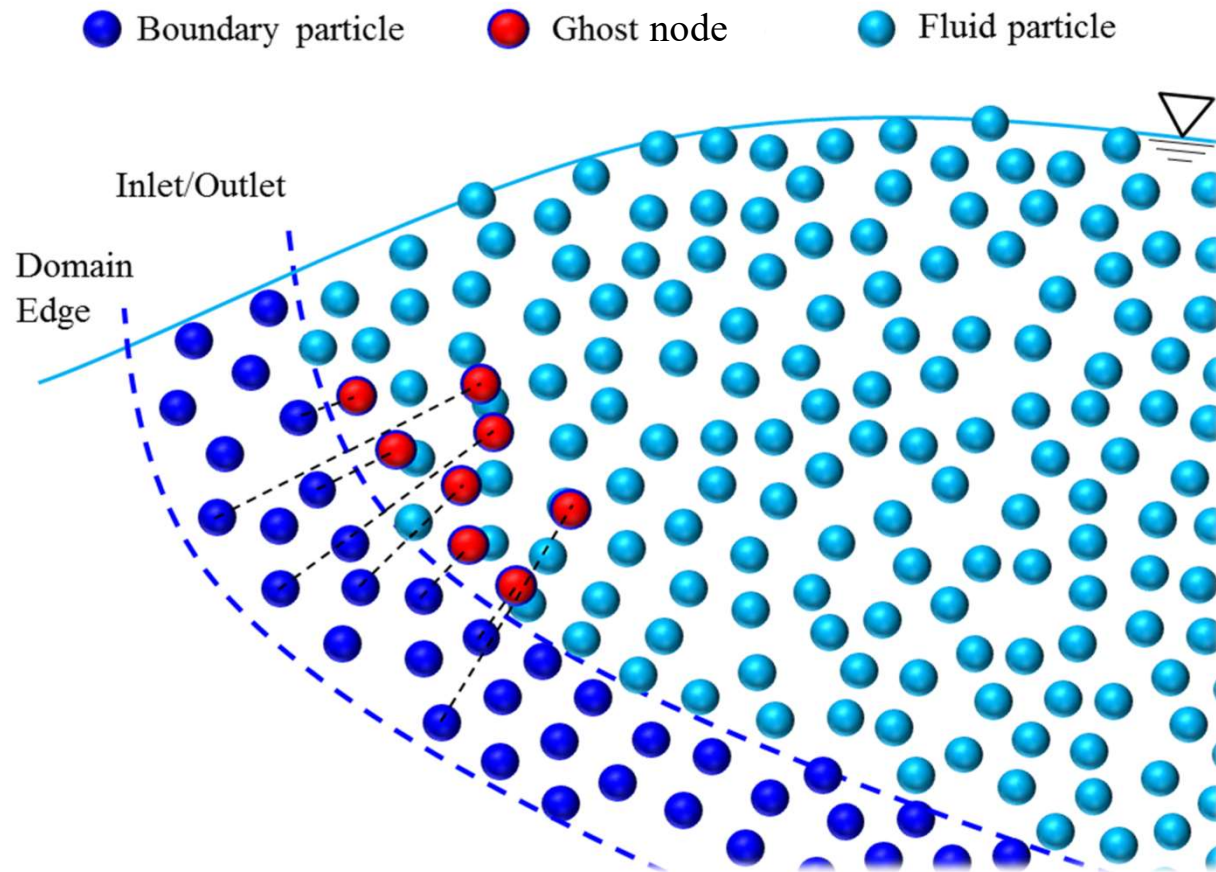
Motivations:

- Boundary conditions belong to the SPHERIC Grand Challenge # 3
- Open boundary conditions (OBCs) are essential to simulate real cases with SPH

Objective:

- To create a **parallel open-source code with open boundary conditions** in 2-D and 3-D to simulate real engineering problems

Buffer Layers



- The ghost nodes are positioned by mirroring the normal distance of OBPs
- Quantities of the ghost nodes are computed with corrected SPH interp.
- Properties of the ghost nodes are mirrored back to OBC (1st order T.S. app.)

Buffer Layers

First order correction start from the Taylor expansion of the function $f(\mathbf{x})$:

$$f(\mathbf{x}) = f_i + \nabla f_i \cdot (\mathbf{x} - \mathbf{x}_i) + O(\mathbf{x} - \mathbf{x}_i)^2$$

Multiplying this by the kernel function $W_i(\mathbf{x})$ we obtain:

$$\int f(\mathbf{x})W_i d\mathbf{x} = f_i \int W_i d\mathbf{x} + \nabla f_i \cdot \int (\mathbf{x} - \mathbf{x}_i)W_i d\mathbf{x}$$

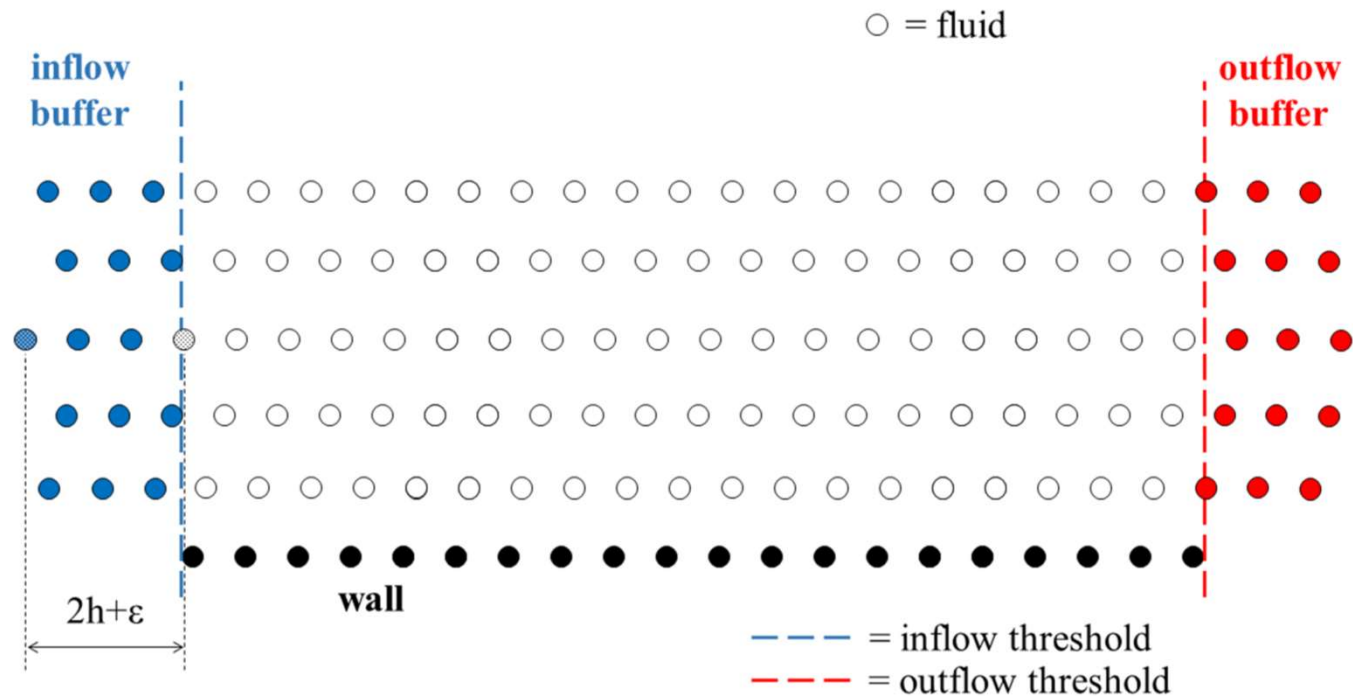
Whereas multiplying by the kernel derivatives $\partial_\beta W_i(\mathbf{x})$ we obtain 3 additional equations:

$$\int f(\mathbf{x})\partial_\beta W_i d\mathbf{x} = f_i \int \partial_\beta W_i d\mathbf{x} + \nabla f_i \cdot \int (\mathbf{x} - \mathbf{x}_i)\partial_\beta W_i d\mathbf{x}$$

Solving this linear system of 4 equations (in 3D) we get the corrected values of f_i , and its derivatives $\partial_\beta f$ (4 unknowns).

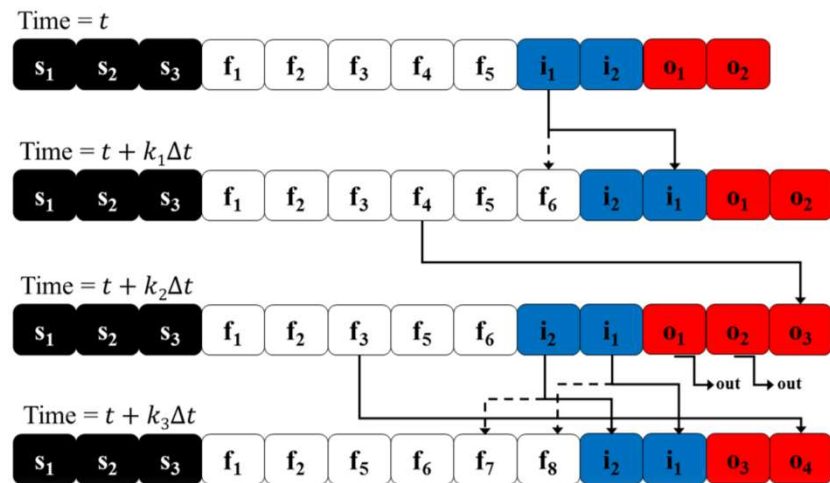
This ensure **first order consistency** also for disordered set of particles.

Buffer Layers



- At the inflow velocity is imposed and pressure is extrapolated
- At the outflow velocity is extrapolated and pressure should be imposed but, due to the pressure oscillation of WC-SPH, more accurate results are obtained by extrapolating also pressure
- The algorithm structure allows the treatment of backflow

Parallel Implementation (CPU)



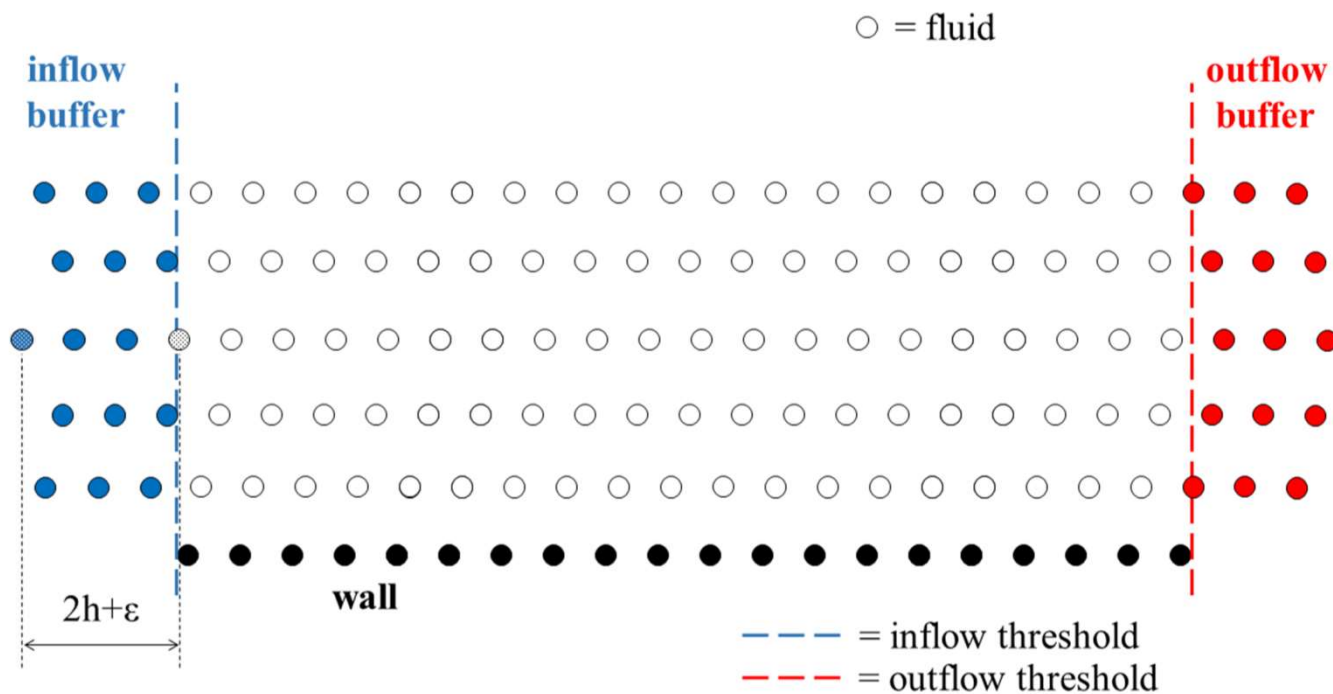
Single array for all particles

Extra memory is pre-allocated for the creation of new particles, and allocated when necessary on the fly

Code type of particles changes when crossing an inlet or outlet

Five types of transitions:
I-F, F-O, O-F, F-Out, O-Out

2-D Open Channel Flow



Main parameters:

$$\Delta x_0 = 0.05 \text{ m}$$

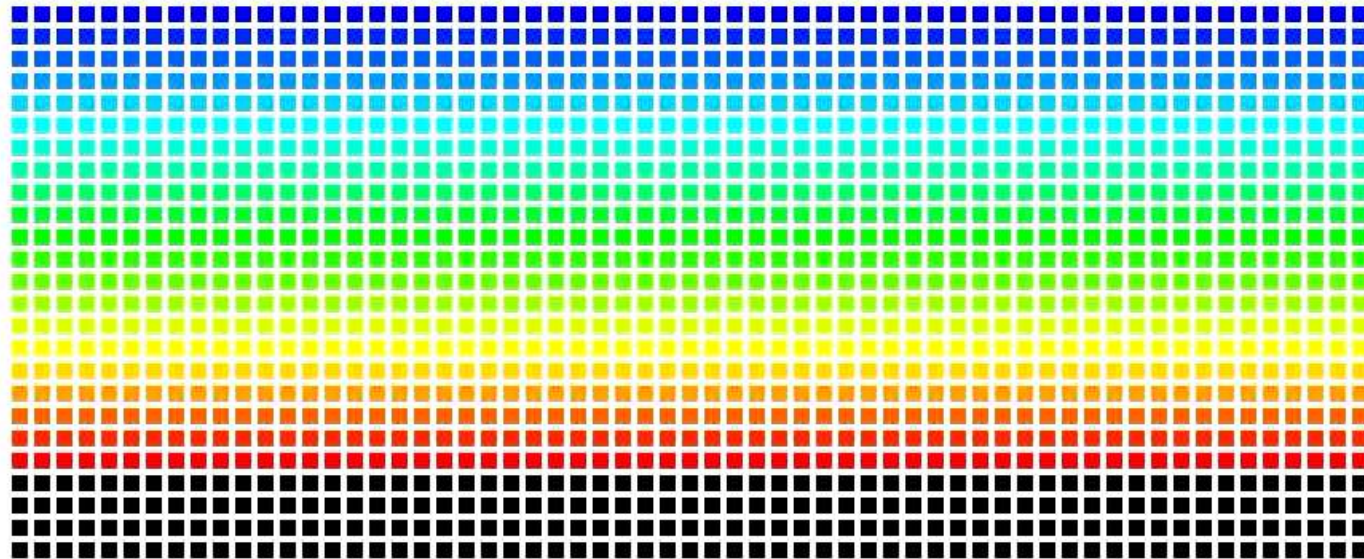
$$N_p = 1,800$$

$$\text{Size: } h = 1 \text{ m } L = 3 \text{ m}$$

$$\alpha = 0.01$$

2-D Open Channel Flow: pressure field

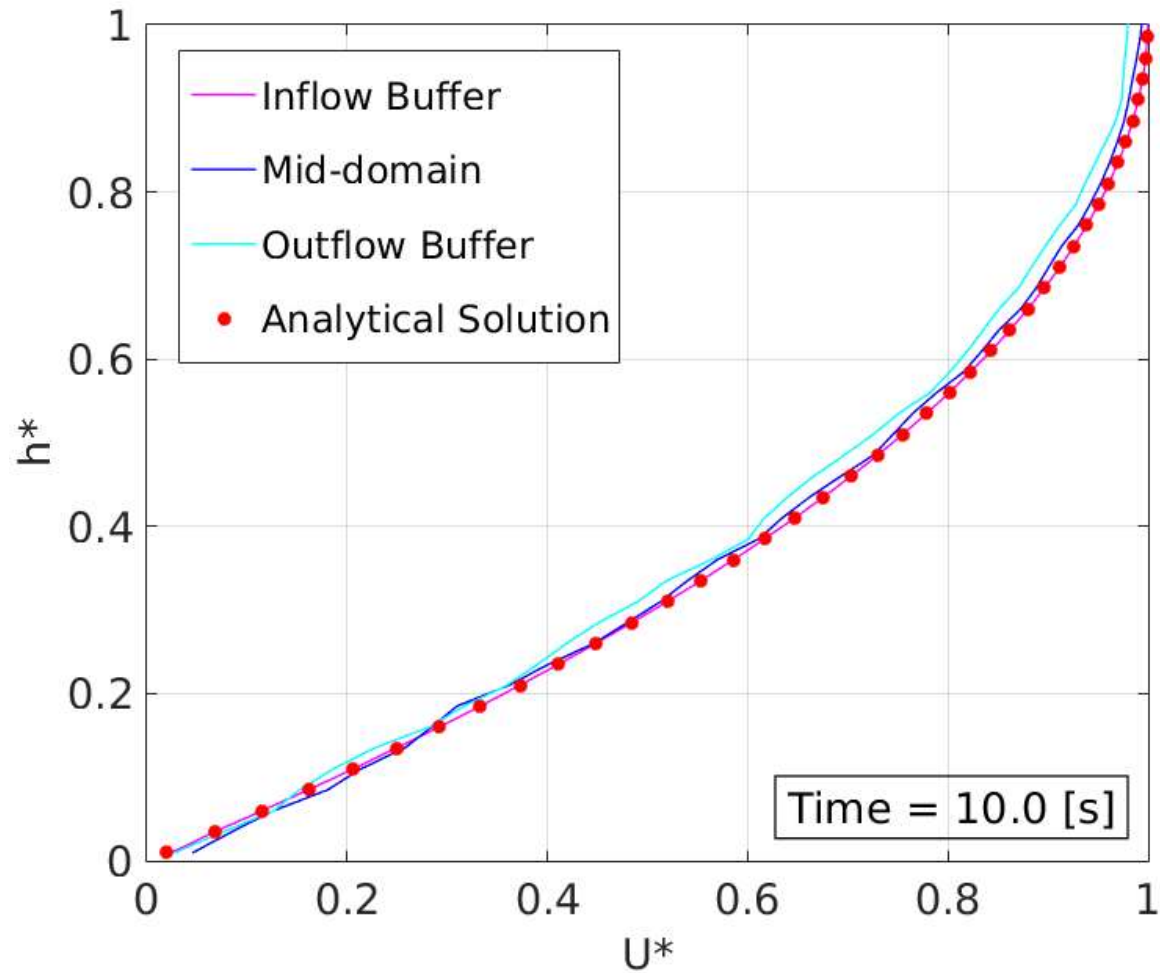
Pressure [Pa]



Time: 0.00 [s]

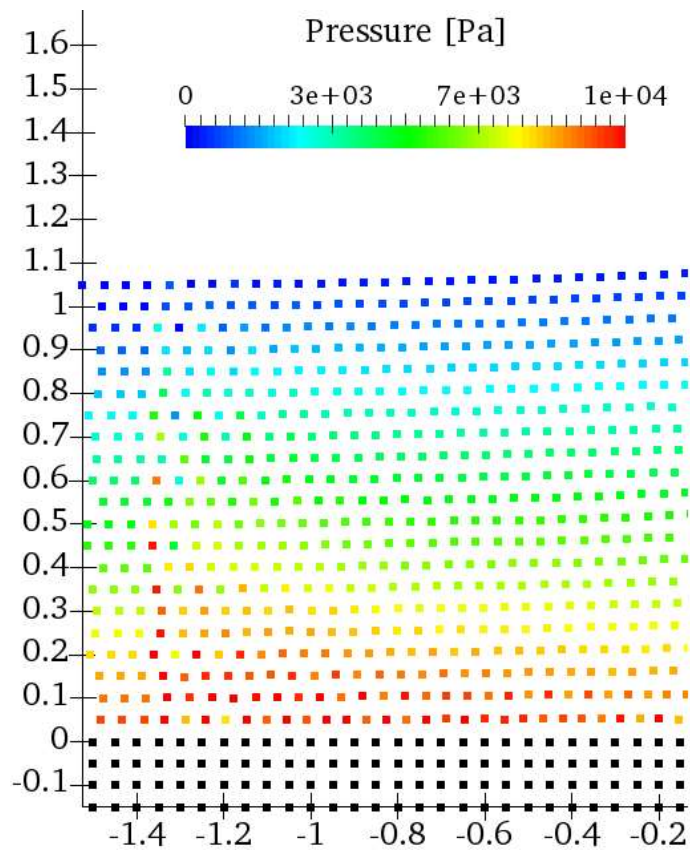
Inflow velocity is assigned, all the rest is extrapolated from ghost nodes.

2-D Open Channel Flow: velocity profile



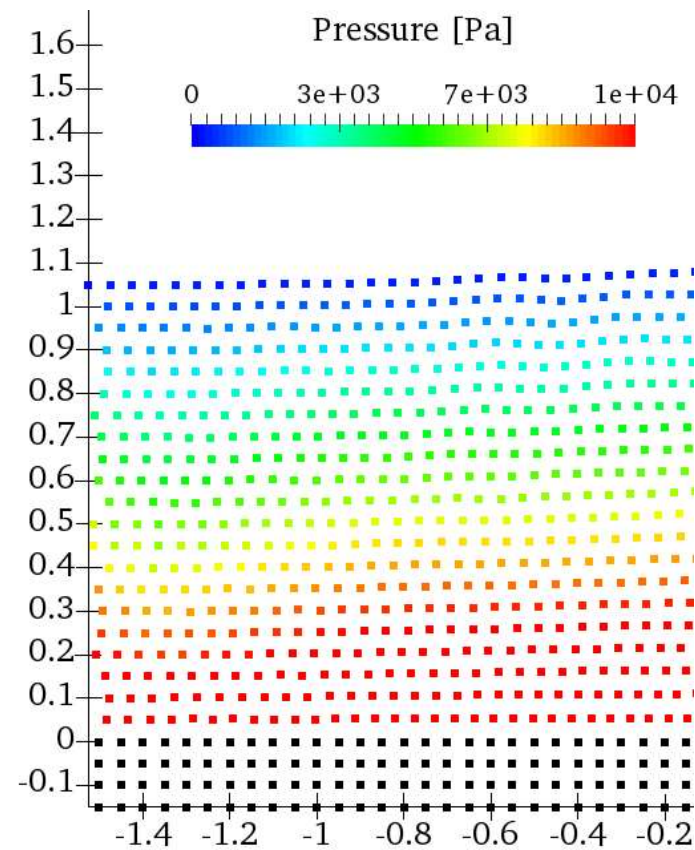
2-D Open Channel Flow: Pressure Field

a) Density imposed



Time: 10.00 [s]

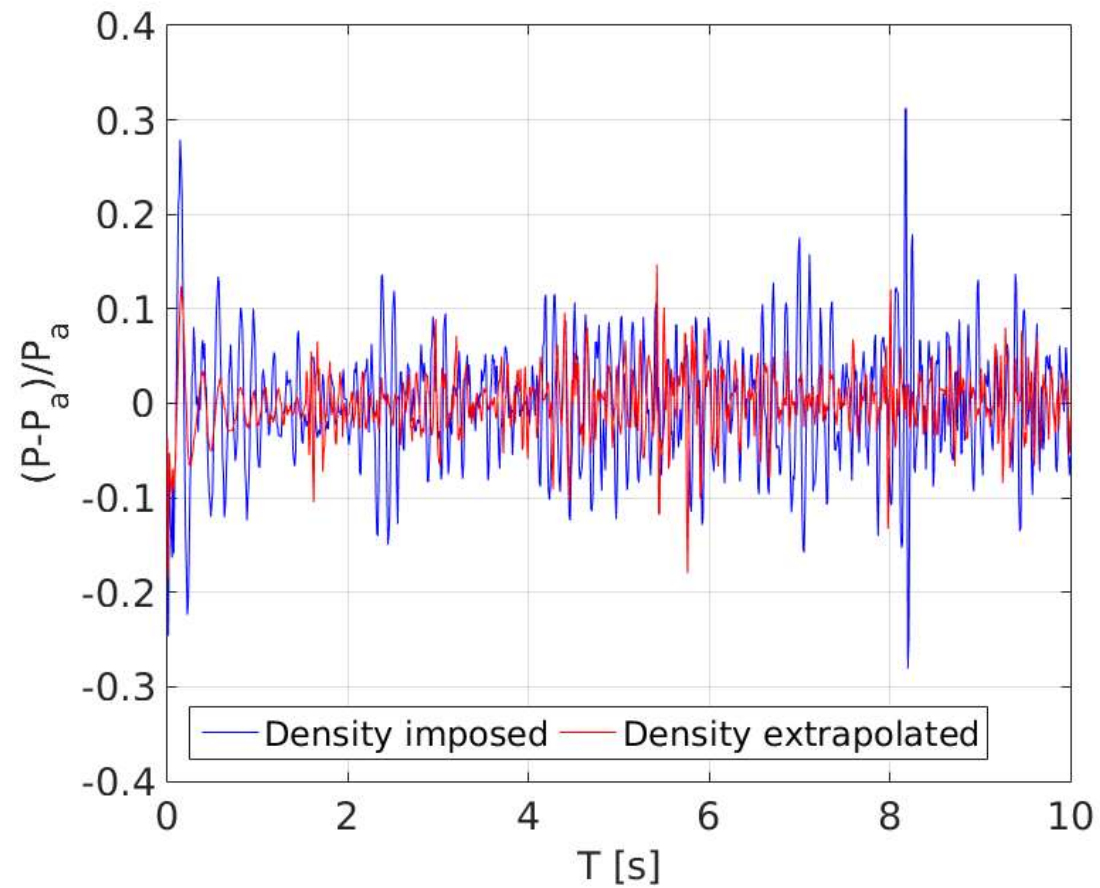
b) Density extrapolated



Time: 10.00 [s]

2-D Open Channel Flow: Pressure Field

Pressure against time in a point close to the inflow



2-D Eisbach River flow



Main parameters:

$$\Delta x_0 = 0.05 \text{ m}$$

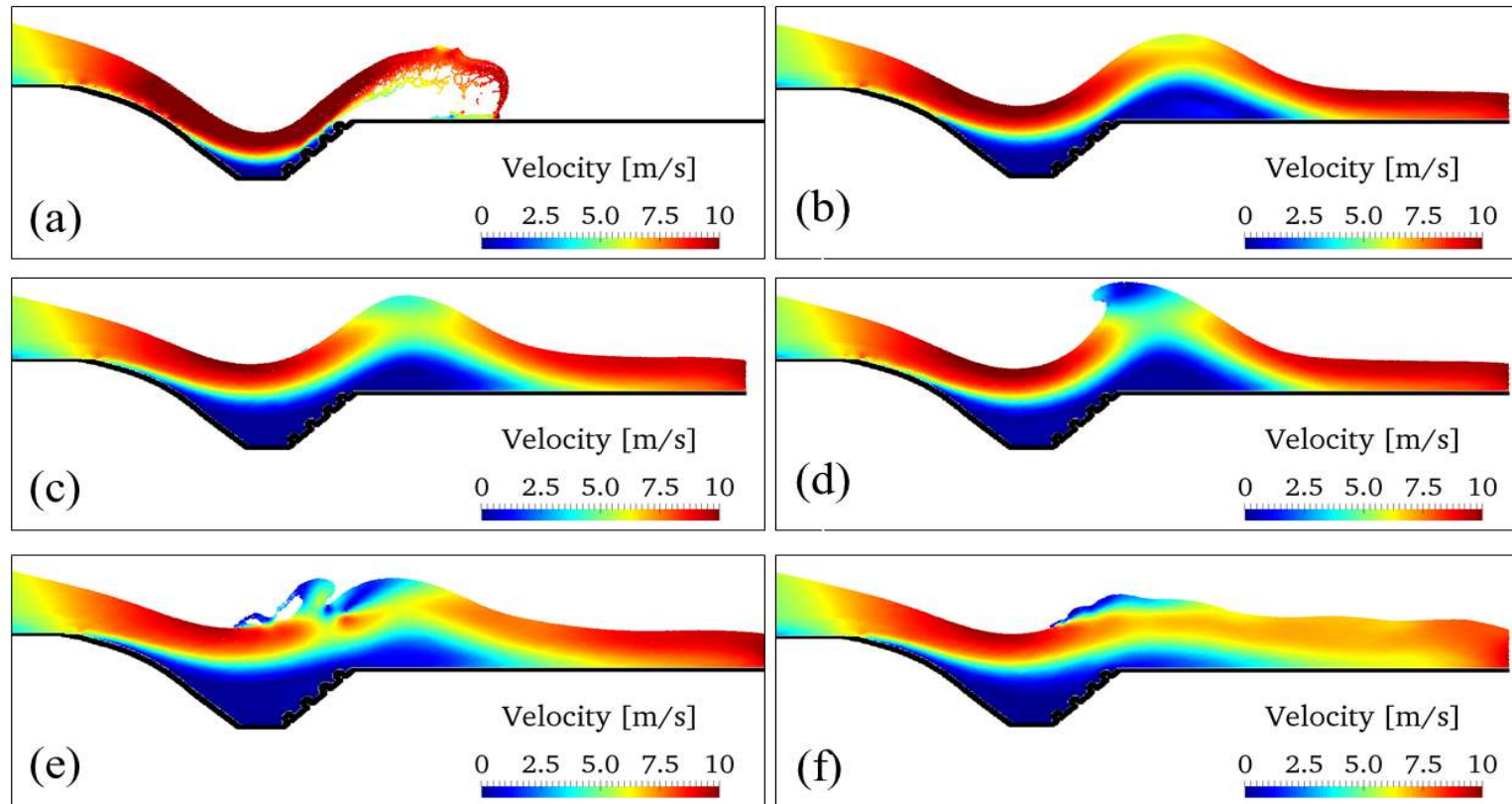
$$h_{\text{avg}} = 2 \text{ m}$$

$$h_{\text{max}} = 7 \text{ m}$$

$$L = 46 \text{ m}$$

$$\alpha = 0.01$$

2-D Eisbach River flow



a) $t=4$ s, b) $t=14$ s, c) $t=16$ s, d) $t=24$ s, e) $t=35$ s, f) $t=41$ s

2-D Eisbach River flow

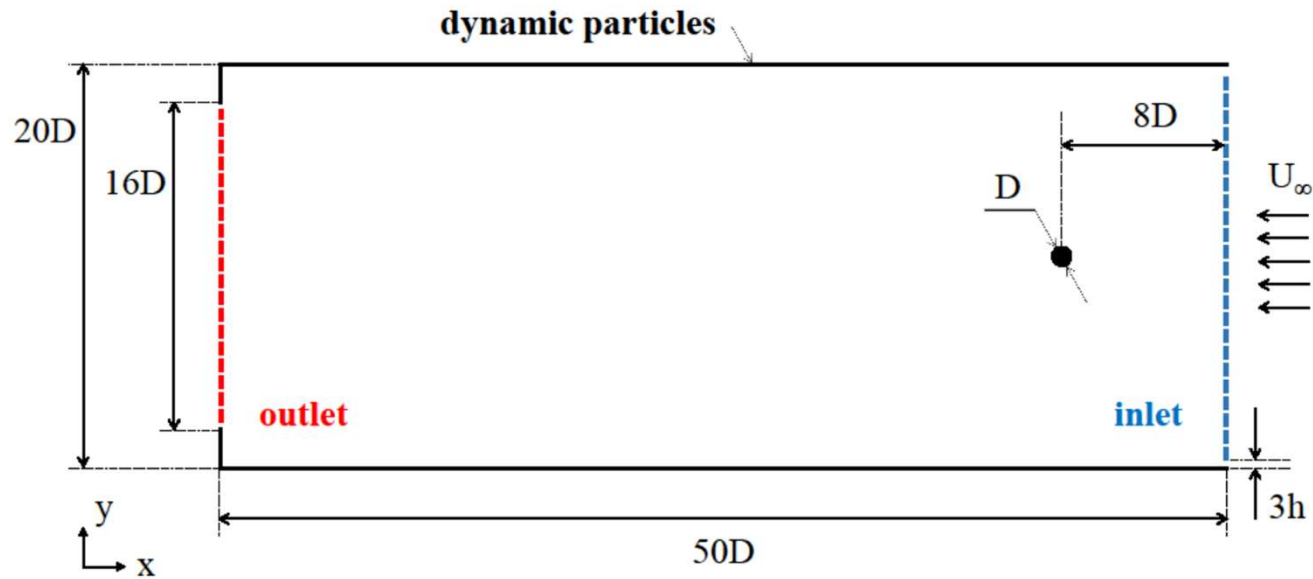
Simulation Info

	SPH
Initial number of SPH particles	5,900
Average SPH particles per time step	425,000
SPH particles created per second of simulation	8,000
CPU time per second of simulation [min]	25
CPU time for open boundary treatment [%]	2 ca.



**The addition of open boundary conditions
causes a negligible increase in computational
time**

2-D Channel Flow Past a Cylinder



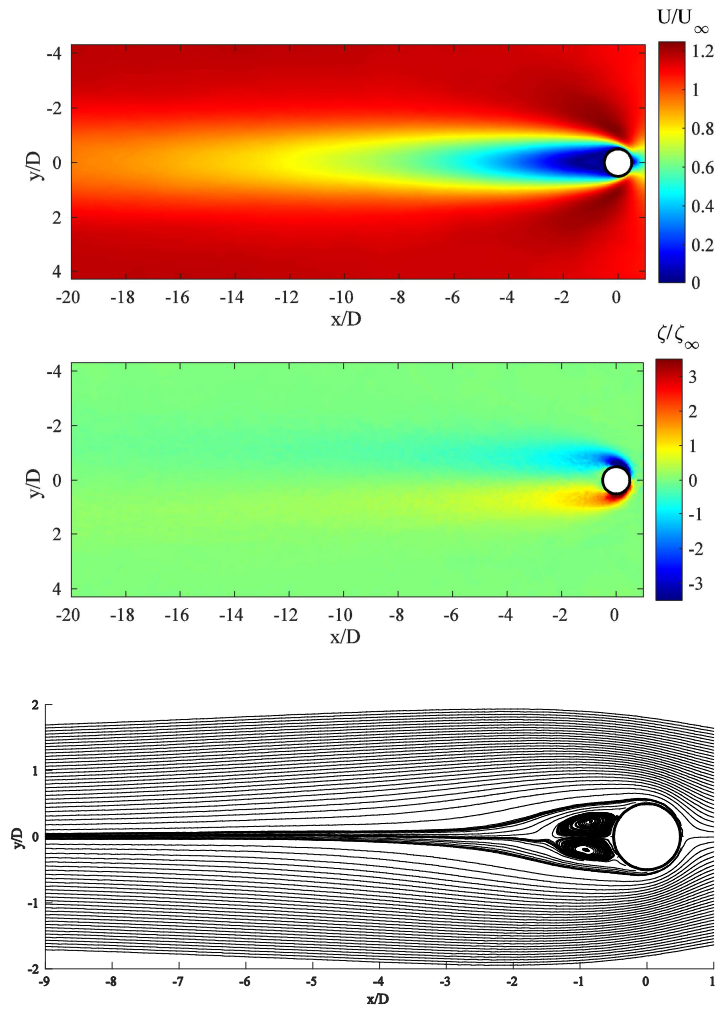
Main parameters:

$$\Delta x_0 = 1/200 D$$

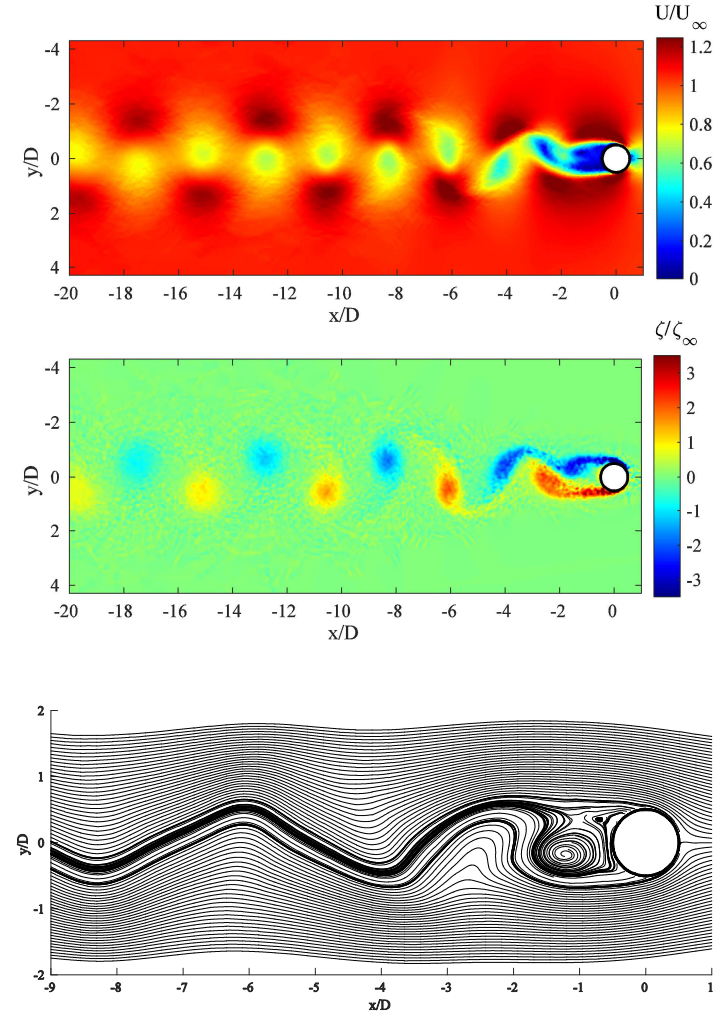
$$N_p = 400,000$$

$$Re \in [20 - 500]$$

2-D Channel Flow Past a Cylinder



a) $Re = 20$



b) $Re = 200$

2-D Channel Flow Past a Cylinder

	Separation Bubble	Separation Angle	Strouhal Nr.
Re = 20	0.92 D	41°	-
Re = 50	2.25 D	53°	-
Re = 100	-	-	0.174
Re = 200	-	-	0.205
Re = 500	-	-	0.235

Excellent agreement of SPH data with many literature results, e.g. Calhoun (2002); Coutanceau and Bouard (1977); Dennis and Chang (1970); Fornberg(1980), and Vacondio et al. (2013).

Conclusion on Open Boundary Conditions

- Open Source Parallel SPH code with variable resolution and adaptivity has been presented
- OpenMP version of the code has been developed in both 2- and 3-D
- Efficacy and efficiency have been discussed
- Code successfully applied to 2D and 3D cases

Next releases

- v4.1 Next months

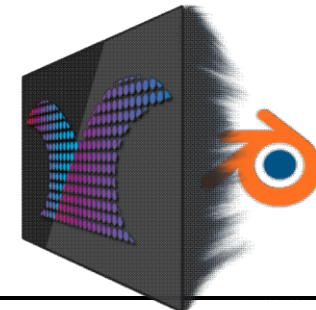
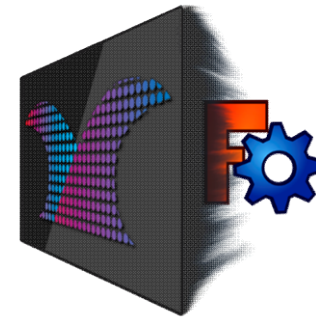
- Fixing bugs
- More comments on .h files
- Improvements on post-processing tools
- New MeasureBoxes
- Latest 2nd order wave generation
- New testcase: Poiseuille flow

- DesignSPHysics

- Python scripts for **FREECAD**

- VisualSPHysics

- Python scripts for **BLENDER**



Next releases

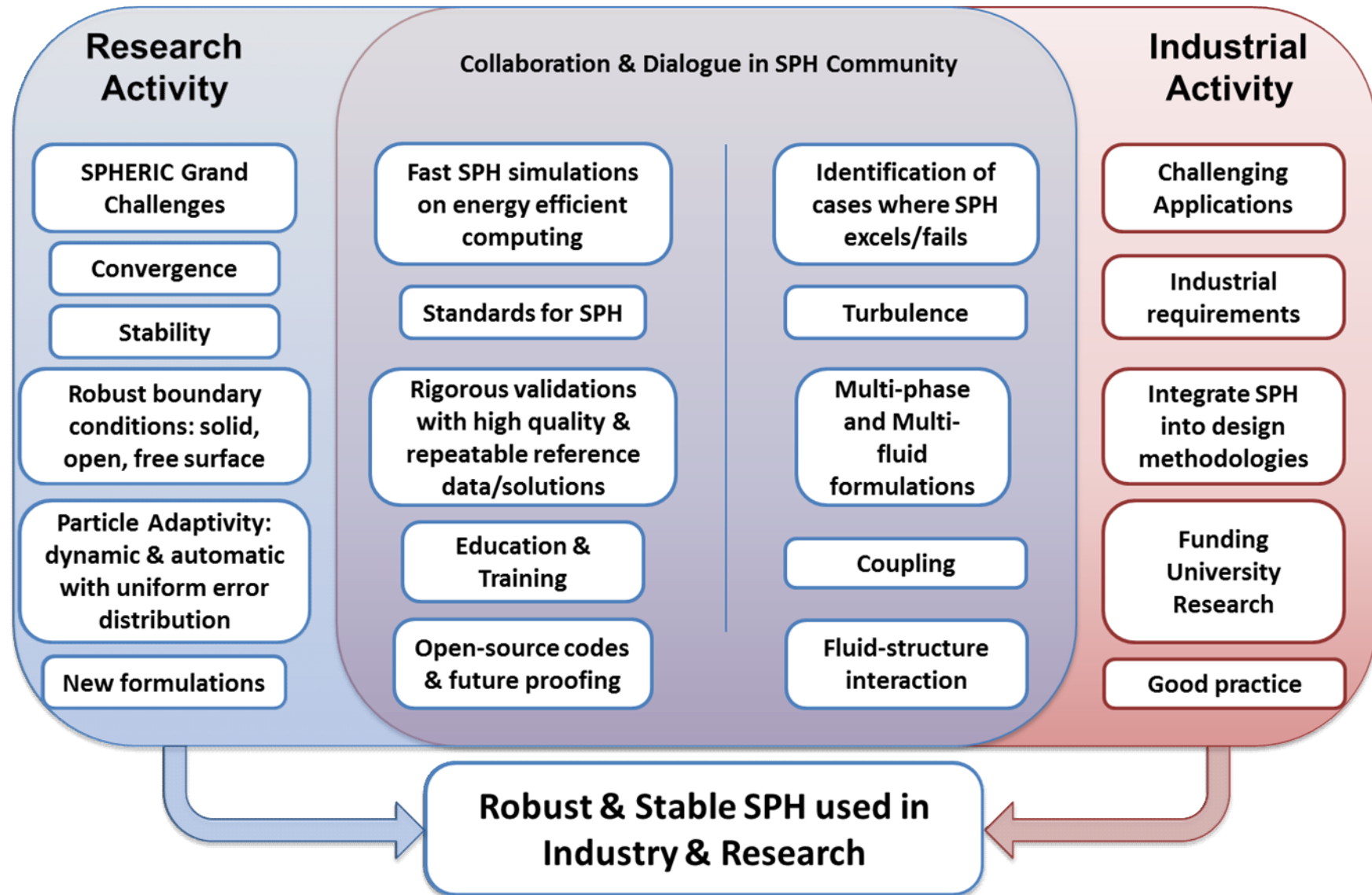
- v4.2 End of 2017
 - New wall Boundary Conditions
 - Open boundaries
 - Chrono implementation for rigid bodies
 - MultiGPU (several GPUs hosted on single node)

Acknowledgements

Ben Rogers, George Fourtakas, Peter K. Stansby, Paolo Mignosa, F. Spreng, A. Tafuni, Jose M. Domínguez, M. Gómez-Gesteira, A J. Crespo



SPH Vision for Free-surface flow:



Violeau & Rogers (2016), "SPH for free-surface flow: past, present and future", *Journal of Hydraulic Research*.



Kingfisher, Massimo Zanotti, Ghedi (BS)

Thank you
renato.vacondio@unipr.it