Current Developments:
Variable Resolution - Boundary Conditions

R. Vacondio, A.J.C. Crespo
Acknowledgements

Ben Rogers, George Fourtakas, Peter K. Stansby, Paolo Mignosa, Jose M. Domínguez, M. Gómez-Gesteira, Jose L. Cercós-Pita
Part 1: Variable resolution
(SPHERIC Grand Challenge #4)
Rubble mound Breakwater: On a GPU GTX 480 the simulation with $2 \times 10^6$ particles ($\Delta x=0.15$ m) requires 21 hours of computational time for 55 seconds of physical time.

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

Motivation

- Despite the huge effort in parallelizing SPH codes (MPI+CUDA) long runtimes are still an issue.
- In Eulerian models variable resolution achieved 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:
- 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)
Static particle distribution with different mass

- 3D simulations of energy device under extreme wave conditions

<table>
<thead>
<tr>
<th>Numerical model</th>
<th>Uniform particle distrib</th>
<th>Variable mass ratio 1:8</th>
</tr>
</thead>
<tbody>
<tr>
<td># of particles</td>
<td>918’000</td>
<td>139’000</td>
</tr>
<tr>
<td>Computational time</td>
<td>7 days</td>
<td>1.5 days</td>
</tr>
<tr>
<td>Δx max (m)</td>
<td>0.02</td>
<td>0.04</td>
</tr>
<tr>
<td>Δx min (m)</td>
<td>0.02</td>
<td>0.02</td>
</tr>
</tbody>
</table>

Omidvar et al. IJNMF (2012)
WC-SPH variationally consistent scheme with variable $h$
WC-SPH formulation

Vacondio et al. 2013 CMAME:
- It is variationally derived
- It conserves both mass and momentum

\[
\frac{d \rho_i}{dt} = \sum_j m_j \left( \mathbf{u}_i - \mathbf{u}_j \right) \cdot \nabla W_j (\mathbf{x}_i, h_j) + 2 \delta h_i \sum_j m_j c_{ij} \left( \frac{\rho_i}{\rho_j} - 1 \right) \frac{r_{ij}}{r_{ij}^2 + \eta^2} \cdot \nabla W_j (\mathbf{x}_i, h_j)
\]

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

\[
\frac{d \mathbf{x}_i}{dt} = \mathbf{v}_i
\]

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

Time integration with Simplectic scheme, Wendland kernel, \( \delta - \text{SPH} \)
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

Splitting procedure (2)

Particle positions and smoothing length are defined as:

\[ h_k = \alpha h \]
\[ d_k = \varepsilon h \]
\[ m_k = \lambda m \]

Where $\alpha$ and $\varepsilon$ are parameters define Particle position and smoothing length

Density error minimization

Local density error: \[ e(\mathbf{x}) = \rho(\mathbf{x}) - \rho^*(\mathbf{x}) = m \sum_{k=1}^{M} m_k W_k(x, h_k) \]

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

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

\[ E^* = \min_{\lambda} \left\{ -C - 2\lambda^T \mathbf{b} + \lambda^T Q \lambda \right\} \]

\[ \lambda_k = \frac{m_k}{m_N} \]

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
Naïve approach: the Cube

- One particle split in 9 Daughter Particles
- First DP in the cube centre (position of the original particle) and 8 particles in the cube vertices
- Wendland kernel
- $\varepsilon$ and $\alpha$ parameters are varied between 0.3 and 0.9 to obtain a global density error matrix:

$$E^\prime = E^\prime (\varepsilon, \alpha)$$
Naïve approach: the Cube (2)

Non-dimensional global density error

\[
\text{Min. global density error is small, but just for } \alpha \approx 0.9
\]
Platonic solids (spherical symmetry):

- **Dodecahedron**
  - 20 vertices
  - $\varepsilon h$

- **Icosahedron**
  - 12 vertices
  - $\varepsilon h$
Dodecahedron (20 vertices)

Global density error for given $\alpha$, $\varepsilon$ is smaller than in the cube pattern
Icosahedron (12 vertices)

Global density error

$m_{\text{min}}/m_{\text{max}}$ ratio

Similar to the Dodecahedron, but with less particles
**Which is the best stencil?**

Cubic 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.

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)

Splitting

- List of particles to split
- Create daughter particles

Coalescing

- List of particles to coalesce
- New “bigger” particles are created
- Check to delete particles with the same “mother” (sequential)

Variable res. formulation overheads: $h$ and $m$ different for each particle, more memory access, and more floating point operation
Variable resolution
Test cases
2-D still water tank

\[ \Delta x_0 = 0.025 \text{ m}, \ (N_p=4800) \]

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 $\delta$–SPH

with $\delta$–SPH
Pressure field at time 5 s

without $\delta$-SPH

with $\delta$-SPH

after 54,000 steps

after 54,000 steps WITH deltaSPH
Vertical distribution of pressure at last instant (t=5s, after 54k steps)

without $\delta$--SPH

with $\delta$--SPH

uniform resolution

one level of splitting

two levels of splitting
Two simulations:
- high resolution, no adaptivity (with $\Delta x_0 = 0.01$ m)
- Initial coarse resolution ($\Delta x_0 = 0.02$ m), splitting activated close to the wall.

Wave height: 0.101m
Peak Period: 2.683s

Experimental campaign in Blankenberge Marina, Altomare et al. 2015(+) model scale 1:5
Forces obtained with experiments, SPH and SPH-adaptive
## Runtimes

<table>
<thead>
<tr>
<th></th>
<th>SPH</th>
<th>SPH-adaptivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta x_0$</td>
<td>0.01m</td>
<td>0.02m</td>
</tr>
<tr>
<td>Initial number of particles</td>
<td>135,883</td>
<td>35,670</td>
</tr>
<tr>
<td>Split particles</td>
<td>-</td>
<td>8,670</td>
</tr>
<tr>
<td>CPU runtime</td>
<td>29.82h</td>
<td>5.81h</td>
</tr>
<tr>
<td>GPU runtime</td>
<td>42.40 min</td>
<td>18.77 min</td>
</tr>
</tbody>
</table>

CPU speedup: 5.13 – GPU speedup: 2.26

Adaptivity overheads are more relevant for the GPU code (more registers, non coalesced memory access etc.)
2D-Falling sphere

- 2-D sphere with radius=1 m
- density=1,200 kg/m$^3$
- 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

\[ m_{\text{max}} = 0.5m_0 \]
\[ m_{\text{min}} = 0.0m_0 \]

\[ m_{\text{max}} = 1.0m_0 \]
\[ m_{\text{min}} = 0.8m_0 \]
SPHERIC Benchmark Test 2

Time: 0.00 s

Mass (kg)

0.00013 - 0.003375
SPHERIC Benchmark Case #2.

Two simulations:
- no adaptivity: blue
- Adaptivity: green

Water height

Pressure
**SPHERIC Benchmark Case #2.**

<table>
<thead>
<tr>
<th></th>
<th>SPH</th>
<th>SPH-adaptivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta x_0$</td>
<td>0.008m</td>
<td>0.015m</td>
</tr>
<tr>
<td>Initial number of particles</td>
<td>1,262,816</td>
<td>184,275</td>
</tr>
<tr>
<td>New daughters by splitting</td>
<td>-</td>
<td>$2.2 \times 10^6$</td>
</tr>
<tr>
<td>Coalesced particles</td>
<td>-</td>
<td>$3.9 \times 10^6$</td>
</tr>
<tr>
<td>CPU runtime</td>
<td>173 h</td>
<td>85 h</td>
</tr>
<tr>
<td>GPU runtime</td>
<td>3.60h</td>
<td>1.99h</td>
</tr>
</tbody>
</table>

**# of fluid particles**

CPU speedup: 2.04  
GPU speedup: 1.80  
This looks bad but:

\[
\frac{(\# \text{ part SPH})}{(\# \text{ part SPH-adapt})} = 4.2
\]
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
Part 2: Boundary Conditions
(SPHERIC Grand Challenge #3)
Different type of Boundary Conditions

(i) Boundary repulsive force
(ii) Fluid extensions to the solid boundary
(iii) Boundary integral (analytical or semi-analytical)

Dynamic boundaries (DBC) from group (i-ii)
Local Uniform STencil (LUST) from group (ii)
Boundary Integral (INTEGRAL) from group (iii)
Dynamic boundaries (DBC)

- same continuity equation as for the fluid particles
- computationally efficient
- Kernel truncation error which prevents convergence
- Over repulsion of fluid particles

\[
\frac{d\rho_i}{dt} = \sum_j \rho_i \frac{m_j}{\rho_j} \mathbf{v}_{ij} \cdot \nabla_i W_{ij}
\]
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
- Computationally more expensive than DB
The density of the fictitious particles is corrected hydrostatically based on the density of the fluid particle.

\[
\rho_k = \rho_i + \left[ \rho_0^7 \sqrt{\frac{\rho_0 g z_{ik}}{B}} \cdot n_v + 1 - \rho_0 \right],
\]

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.

\[
u_k = (u_i - u_v) \frac{x_{vk} \cdot n_v}{x_{iv} \cdot n_v} - u_v.
\]

**Momentum equation**

\[
\left< \frac{d \mathbf{u}}{d t} \right>_i = - \sum_{j \in \Omega_f} m_j \left( \frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} + \Pi_{ij} \right) \nabla W_{ij} - \sum_{k \in \Omega_b} m_k \left( \frac{P_i}{\rho_i^2} + \frac{P_k}{\rho_k^2} + \Pi_{ik} \right) \nabla W_{ik},
\]

**Continuity equation**

\[
\left< \frac{d \rho}{d t} \right>_i = \sum_{j \in \Omega_f} m_j (u_i - u_j) \cdot \nabla W_{ij} + \sum_{k \in \Omega_b} m_k (u_i - u_k) \cdot \nabla W_{ik}.
\]
Boundary Integral (INTEGRAL)

Numerical approximation of the integral

SPH interpolant

Continuous formulation

\[ \langle f(x) \rangle = \frac{1}{\gamma_h(x)} \int_{y \in \Omega} f(y) W_h(x - y) \, dy \]

\[ \gamma_h(x) := \int_{y \in \Omega} W_h(x - y) \, dy. \]

Discrete formulation

\[ \langle f \rangle_i = \frac{1}{\gamma_i} \left( \sum_{j \in \text{Fluid}} \frac{f_j}{\rho_j} W_{ij} m_j \right) \]

\[ \gamma_i = \sum_{j \in \text{Fluid}} \frac{1}{\rho_j} W_{ij} m_j \]

Integrals along the boundary are replaced by a sum of area elements.
Boundary Integral (INTEGRAL)

SPH differential operator

Continuous formulation

\[ \langle \mathcal{D}f(\mathbf{x}) \rangle = \frac{1}{\gamma_h(\mathbf{x})} \int_{y \in \Omega} \mathcal{D}f(\mathbf{y}) W_h(\mathbf{x} - \mathbf{y}) d\mathbf{y}, \quad \langle \mathcal{D}f(\mathbf{x}) \rangle = \frac{1}{\gamma_h(\mathbf{x})} \left( \int_{y \in \Omega} f(\mathbf{y}) \cdot \nabla W_h(\mathbf{y} - \mathbf{x}) d\mathbf{y} + \int_{y \in \partial \Omega} f(\mathbf{y}) \cdot \mathbf{n}(\mathbf{y}) W_h(\mathbf{y} - \mathbf{x}) d\mathbf{y} \right) \]

Discrete formulation

\[ \langle \mathcal{D}f \rangle_i = \frac{1}{\gamma_i} \left( \sum_{j \in \text{Fluid}} \frac{f_j}{\rho_j} \cdot \nabla W_{ij} m_j + \sum_{j \in \text{Boundary}} f_j \cdot n_j W_{ij} s_j \right) \]
Variable resolution

Test cases

Can you guess which is the first test case?
TEST 1: Still water with a wedge (2-D)

- A low value of viscosity is used ($\alpha_{\tau}=0.01$)
- No density filter
- $h/dp=1.3$
TEST 1: Still water with a wedge (2-D)

Convergence test

Results are analysed after 20 seconds of physical time (85,000 steps)
TEST 1: Still water with a wedge (2-D)

\[ dp = 0.04 \]
\[ Np = 672 \]

\[ dp = 1 \]
\[ Np = 11'135 \]
TEST 2: SPHERIC Benchmark Test Case #2

SPHERIC Benchmark Test Case #2

δ-SPH is now employed to obtain smoothed density distribution

\( D_p = 0.01, \; N_p = 800k, \; Time = 6.0 \; s \)
TEST 2: SPHERIC Benchmark Test Case #2

DBC 800k h/dp:1.3

LUST 800k h/dp:1.3

INTEGRAL 800k h/dp:4.0

Density
1002.5
1002.0
1001.0
1000.0
TEST 2: Dam break with obstacle (3-D)

Dp=0.01, Np=800k, Time= 0.40 s

\[ k_{\text{DBC}} = 1.2 \]
\[ k_{\text{LUST}} = 0.5 \]
\[ k_{\text{INTEGRAL}} \approx 0 \]
TEST 2: Dam break with obstacle (3-D)

Experimental and numerical water heights measured at the probes H3 and H4 with $dp=0.01\text{m}$ and $h/dp=1.3$
TEST 2: Dam break with obstacle (3-D)

Experimental and numerical pressures measured at the sensors P1 and P6 with \( dp=0.01\text{m} \) and \( h/dp=1.3 \)
TEST 2: Dam break with obstacle (3-D)

Experimental and numerical pressures measured at the sensors P1 and P6 with $h/dp=1.3, 2.6, 4.0$

**INTEGRAL**

$h/dp=4.0$
### TEST 2: Dam break with obstacle (3-D)

#### Performance Analysis

<table>
<thead>
<tr>
<th>Test</th>
<th>BC type</th>
<th>Np</th>
<th>h/dp</th>
<th>Runtime (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEST 2</td>
<td>DBC</td>
<td>100k</td>
<td>1.3</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>800k</td>
<td></td>
<td>1.47</td>
</tr>
<tr>
<td>Dam break 3-D</td>
<td>LUST</td>
<td>100k</td>
<td>1.3</td>
<td>0.18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>800k</td>
<td></td>
<td>1.99</td>
</tr>
<tr>
<td></td>
<td>INTEGRAL</td>
<td>100k</td>
<td>1.3</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>800k</td>
<td></td>
<td>2.53</td>
</tr>
</tbody>
</table>

The ratio h/dp needs to be increased till 4 to get good results for INTEGRAL.
Conclusions

A comparison of three different boundary conditions has been performed.

DBC can be applied to arbitrary 2-D and 3-D geometries, BUT a high repulsive force is generated acting on the fluid particles resulting in a separation distance.

LUST BC is more computationally expensive than DBC but more accurate and it addresses most of the issues of DBC.

INTEGRAL methodology requires large number of neighbours within the support (as discussed in the consistency notes and demonstrated in the test cases) to obtain good accuracy.