GPUSPH Theory Guide

version 4.0 — October 2016

Contents

1 Introduction

In the present document the SPH formulations realized in GPUSPH will be described. While this section is as complete as necessary we recommend to read the cited literature in order to gain a full understanding of the respective methods.

The basic premise is that GPUSPH approximates solutions to the Navier-Stokes equations which are given by

$$\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} = -\frac{1}{\rho}\boldsymbol{\nabla}p + \nabla \cdot \left[\nu(\boldsymbol{\nabla}\boldsymbol{v} + \boldsymbol{\nabla}\boldsymbol{v}^T)\right] + \boldsymbol{g}, \tag{1}$$

where $\boldsymbol{v}$ is the velocity, ρ the density, p the pressure, ν the kinematic viscosity and $\boldsymbol{g}$ an external force. These equations are coupled with the continuity equation

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = -\rho\nabla \cdot \boldsymbol{v}. \tag{2}$$

In order to close the system of equations a weakly-compressible formulation is chosen which uses the Cole Equation of State given by

$$p = \frac{c_0\rho_0}{\xi}\left[\left(\frac{\rho}{\rho_0}\right)^{\xi} - 1\right], \tag{3}$$

where c_0 is the speed of sound, ρ_0 the reference density and ξ is the polytropic index, equal to 7 in the case of water.

2 The SPH approximation

At the heart of the SPH method is an interpolation that defines a physical quantity at a certain position. This interpolation consists in two steps: a kernel approximation followed by a particle approximation.

2.1 Kernel approximation

Let f be a field then

$$f(\boldsymbol{r}) = \int f(\boldsymbol{r}')\delta(\|\boldsymbol{r} - \boldsymbol{r}'\|)\mathrm{d}\boldsymbol{r}', \tag{4}$$

where δ is the Dirac delta distribution. The continuous SPH approximation can be written as

$$<f>(\boldsymbol{r}) = \int_{\Omega} f(\boldsymbol{r}')W(|\boldsymbol{r} - \boldsymbol{r}'|, h)\mathrm{d}\boldsymbol{r}', \tag{5}$$

3

where the Dirac delta function was replaced by a weight or kernel function W, Ω is the computational domain and h is the smoothing length. For this approximation to be consistent, two conditions are required:

- $\lim_{h\to 0} W(.,h) = \delta(.)$, *i.e.* the kernel function converges to the delta function,

- $\int W(\boldsymbol{r} - \boldsymbol{r}', h)\mathrm{d}\boldsymbol{r}' = 1$, *i.e.* the kernel function has unitary integral.

Additional constraints are imposed to the SPH kernels, in order to simplify the physical modelling:

- $W(\boldsymbol{r} - \boldsymbol{r}', h) = W(\boldsymbol{r}' - \boldsymbol{r}, h)$, *i.e.* the kernel is symmetric,

- the kernel has compact support.

From now one we will assume that the kernel fulfill this additional properties.

2.2   Particle approximation

The kernel approximation of Eq. (5) does not yet contain any spatial discretization. The spatial discretization is based on a the subdivision of the simulation domain using particles with each particle representing a small volume of the simulation domain. Each particle carry his one physical quantities (mass, density, velocity, ...). SPH being a purely Lagrangian method there is no need for any connectivity between particles.

For a given particle a, $\boldsymbol{r}_a$ will denote the particle position and f_a the value of f at the particle.

Now that the space has been discretized we can apply the particle approximation to the kernel one : approximating Eq. (5) by the following sum

$$[f](\boldsymbol{r}_a) = \sum_{b\in\mathcal{F}} \frac{m_b}{\rho_b} f_b W_{ab}, \tag{6}$$

where $\mathcal{F}$ is the set of all particles, m the mass and $W_{ab} = W(|\boldsymbol{r}_a - \boldsymbol{r}_b|/h, h)$. It should be noted that $[f] \neq f$.

3   SPH kernels

Currently GPUSPH features four different kernels.

- CUBICSPLINE: the cubic spline kernel

- `QUADRATIC`: the quadratic splain kernel

- `WENDLAND`: the quintic Wendland kernel

- `GAUSSIAN`: the Gaussian kernel

The formulae for the different kernels and their derivatives are given below. The variable q is defined by:

$$q = \frac{|\boldsymbol{r} - \boldsymbol{r'}|}{h} \tag{7}$$

The B-splines kernels are widely used in the SPH literature. The quadratic spline is defined, in three-dimensions, by:

$$W(q, h) = \frac{\alpha_2}{h^3} f_2(q) \tag{8}$$

with:

$$f_2(q) = \begin{cases} \dfrac{1}{4}q^2 - q + 1 & 0 \le q \le 2 \\ 0 & 2 < q \end{cases} \tag{9}$$

where $\alpha_2 = \frac{15}{16\pi}$ and its derivative by

$$f_2'(q) = \begin{cases} \dfrac{1}{2}q - 1 & 0 \le q \le 2 \\ 0 & 2 < q \end{cases} \tag{10}$$

The cubic spline is defined, in three-dimensions, by:

$$W(q, h) = \frac{\alpha_3}{h^3} f_3(q) \tag{11}$$

with:

$$f_3(q) = \begin{cases} 1 - \dfrac{3}{2}q^2 + \dfrac{3}{4}q^3 & 0 \le q \le 1 \\ \dfrac{1}{4}(2 - q)^3 & 1 \le q \le 2 \\ 0 & 2 < q \end{cases} \tag{12}$$

where $\alpha_3 = 1/\pi$ and its derivative by

$$f_3'(q) = \begin{cases} -3q + \dfrac{9}{4}q^2 & 0 \le q \le 1 \\ -\dfrac{3}{4}(2 - q)^2 & 1 \le q \le 2 \\ 0 & 2 < q \end{cases} \tag{13}$$

The Wendland kernel is given by

$$W(q, h) = \begin{cases} \dfrac{\alpha_W}{h^3} \left(1 - \dfrac{q}{2}\right)^4 (1 + 2q) & 0 \le q \le 2 \\ 0 & if \, 2 < q \end{cases} \tag{14}$$

where $\alpha_W = 21/(16\pi)$ and its derivative by

$$|\boldsymbol{\nabla} W(q, h)| = \begin{cases} \dfrac{-5\alpha_W}{h^5} q \left(1 - \dfrac{q}{2}\right)^3 & 0 \le q \le 2 \\ 0 & 2 < q \end{cases} \tag{15}$$

The truncated Gaussian kernel is defined by

$$W(q, h) = \begin{cases} \dfrac{\alpha_G}{h^3} \left(e^{-q^2} - e^{-\left(\frac{3}{h}\right)^2}\right) & 0 \le q \le 3 \\ 0 & 3 < q \end{cases} \tag{16}$$

where $\alpha_G = \dfrac{1}{\pi\left(\sqrt{\pi} - 36e^{-9}\right)}$.

$$|\boldsymbol{\nabla} W(q, h)| = \begin{cases} -2q\dfrac{\alpha_G}{h^3} \left(e^{-q^2} - e^{-\left(\frac{3}{h}\right)^2}\right) & 0 \le q \le 3 \\ 0 & 3 < q \end{cases} \tag{17}$$

3.1 First-order derivatives

In order to approximate derivatives the continuous SPH interpolation Eq. (5) is used as follows

$$< \boldsymbol{\nabla} f > (\boldsymbol{r}) = \int_\Omega \boldsymbol{\nabla} f(\boldsymbol{r}') W(|\boldsymbol{r} - \boldsymbol{r}'|, h) \mathrm{d}\boldsymbol{r}' \tag{18}$$

Applying integration by parts yields

$$< \boldsymbol{\nabla} f > (\boldsymbol{r}) = - \int_\Omega f(\boldsymbol{r}') \boldsymbol{\nabla}_{\boldsymbol{r}'} W(|\boldsymbol{r} - \boldsymbol{r}'|, h) \mathrm{d}\boldsymbol{r}' + \int_\Omega \boldsymbol{\nabla}(f(\boldsymbol{r}') W(|\boldsymbol{r} - \boldsymbol{r}'|, h)) \mathrm{d}\boldsymbol{r}' \tag{19}$$

As W is symmetric the kernel is antisymmetric and thus

$$\boldsymbol{\nabla}_{\boldsymbol{r}'} W(|\boldsymbol{r} - \boldsymbol{r}'|, h) = -\boldsymbol{\nabla}_{\boldsymbol{r}} W(|\boldsymbol{r} - \boldsymbol{r}'|, h) \tag{20}$$

Furthermore, using Stokes theorem the last term in Eq. (19) can be rewritten as integral over the boundary of the computational domain, denoted by $\partial\Omega$ which yields

$$< \boldsymbol{\nabla} f > (\boldsymbol{r}) = \int_\Omega f(\boldsymbol{r}') \boldsymbol{\nabla}_{\boldsymbol{r}} W(|\boldsymbol{r} - \boldsymbol{r}'|, h) \mathrm{d}\boldsymbol{r}' - \int_{\partial\Omega} (f(\boldsymbol{r}') W(|\boldsymbol{r} - \boldsymbol{r}'|, h)) \boldsymbol{n}_{\boldsymbol{r}'} \mathrm{d}\boldsymbol{\Gamma}(\boldsymbol{r}') \tag{21}$$

where $\boldsymbol{n_{r'}}$ is the inward pointing normal. As a convention for the remainder of this document all normals will point inward the domain. If the domain Ω is unbounded then the last term of Eq. (21) will vanish due to the compact support of the kernel W. Thus, in a continuous sense the first derivative can be approximated as

$$< \boldsymbol{\nabla} f > (\boldsymbol{r}) = \int_{\Omega} f(\boldsymbol{r}')\boldsymbol{\nabla}_r W(|\boldsymbol{r} - \boldsymbol{r}'|, h)\mathrm{d}\boldsymbol{r}' \tag{22}$$

The discretization in space again replaces the integral by a sum over all particles and so

$$[\boldsymbol{\nabla} f](\boldsymbol{r}_a) = \sum_{b\in\mathcal{F}} V_b f_b \boldsymbol{\nabla}_a W_{ab} \tag{23}$$

where $V_b = m_b/\rho_b$ is the volume of a particle b. It is common in SPH practice to write the gradient operator slightly different as it is used to compute the pressure gradient and thus Newtons third law of equal but opposite forces should be obeyed. This can be achieved by defining the SPH gradient as

$$\mathbf{G}_a(f) = [\boldsymbol{\nabla} f](\boldsymbol{r}_a) + [\boldsymbol{\nabla} f(\boldsymbol{r}_a)](\boldsymbol{r}_a) \tag{24}$$

where the latter derivative is that of a constant, thus equal to zero in the continuous framework. However, the SPH discretisation of the gradient as defined by equation (23) does not yield zero when applied to constants. The second term will thus have a non-zero contribution in this newly-defined SPH gradient.

The principle of energy conservation requires the gradient and divergence operators to be skew-adjoint, *i.e.*

$$< \mathbf{G}_a(f), \boldsymbol{F} >= - < f, D_a(\boldsymbol{F}) > \tag{25}$$

where $< .,. >$ is a scalar product on all particles. As a result of this constraint the divergence can be shown to be

$$D_a(\boldsymbol{F}) = [\nabla \cdot \boldsymbol{F}]_a - [\nabla \cdot \boldsymbol{F}_a]_a \tag{26}$$

This allows to write the gradient and divergence as

$$\mathbf{G}_a(f) = \sum_{b\in\mathcal{F}} V_b(f_b + f_a)\boldsymbol{\nabla}_a W_{ab} \tag{27}$$

and

$$D_a(\boldsymbol{F}) = \sum_{b\in\mathcal{F}} V_b(\boldsymbol{F}_b - \boldsymbol{F}_a) \cdot \boldsymbol{\nabla}_a W_{ab} \tag{28}$$

respectively.

7

3.2 Second-order derivatives

In theory second-order derivatives could simply be derived from first order ones, but that would cause second derivatives to appear which are numerically unstable. Thus, the preferred approach is to use a combination of a SPH first order derivative and a finite difference first order derivative.

The goal of this section is to derive a discretization for $\boldsymbol{\nabla} \cdot (f \boldsymbol{\nabla} \otimes \boldsymbol{F})$. To approximate the interior gradient, a Taylor Series of $\boldsymbol{F}$ can be used as follows

$$\boldsymbol{F}_b = \boldsymbol{F}_a - (\boldsymbol{\nabla}_a \otimes \boldsymbol{F})^T \cdot \boldsymbol{r}_{ab} + \mathcal{O}(\boldsymbol{r}_{ab}) \tag{29}$$

where $\boldsymbol{r}_{ab} = \boldsymbol{r}_a - \boldsymbol{r}_b$, a convention that will be used throughout this document also for quantities different from $\boldsymbol{r}$. The above can be rewritten to obtain an expression for the interior gradient

$$(\boldsymbol{\nabla}_a \otimes \boldsymbol{F})^T \cdot \frac{\boldsymbol{r}_{ab}}{|\boldsymbol{r}_{ab}|} \approx \frac{1}{|r_{ab}|} \boldsymbol{F}_{ab} \tag{30}$$

and similarly

$$(\boldsymbol{\nabla}_b \otimes \boldsymbol{F})^T \cdot \frac{\boldsymbol{r}_{ab}}{|\boldsymbol{r}_{ab}|} \approx \frac{1}{|r_{ab}|} \boldsymbol{F}_{ab} \tag{31}$$

As the kernel is radially symmetric

$$\boldsymbol{\nabla}_a W_{ab} = \frac{\boldsymbol{r}_{ab}}{|\boldsymbol{r}_{ab}|} |\boldsymbol{\nabla}_a W_{ab}| \tag{32}$$

The Divergence similar to the one given by Eq. (28) but with a $+$ instead of the $-$ yields

$$\boldsymbol{\nabla} \cdot (f \boldsymbol{\nabla} \otimes \boldsymbol{F}) = \sum_{b \in \mathcal{F}} V_b (f_b \boldsymbol{\nabla} \otimes \boldsymbol{F} + f_a \boldsymbol{\nabla} \otimes \boldsymbol{F}) \frac{\boldsymbol{r}_{ab}}{|\boldsymbol{r}_{ab}|} |\boldsymbol{\nabla} W_{ab}| \tag{33}$$

which, with Eqs. (30), (31) and (32), can be used to define the Laplacian

$$\mathbf{L}_a(f, \boldsymbol{F}) = \boldsymbol{\nabla} \cdot (f \boldsymbol{\nabla} \otimes \boldsymbol{F}) = \sum_{b \in \mathcal{F}} V_b (f_b + f_a) \boldsymbol{F}_{ab} \frac{1}{|\boldsymbol{r}_{ab}|} |\boldsymbol{\nabla} W_{ab}| \tag{34}$$

3.3 Discretization of the Navier-Stokes equations

With the definition of the three operators $\mathbf{G}$, D and $\mathbf{L}$ as given by Eqs. (27), (28) and (34), respectively, the Navier-Stokes equations (1) and (2) can be discretized in space as

$$\frac{\mathrm{d}\boldsymbol{v}_a}{\mathrm{d}t} = -\frac{1}{\rho_a} \mathbf{G}_a(p) + \mathbf{L}_a(\nu, \boldsymbol{v}) + \boldsymbol{g} \tag{35}$$

$$\frac{\mathrm{d}\rho_a}{\mathrm{d}t} = -\rho_a D_a(\boldsymbol{v}) \tag{36}$$

8

The final step to a full discretization is the implementation of a time integration scheme. GPUSPH currently features a predictor-corrector scheme that is given by

$$\boldsymbol{u}^{n+1/2} = \boldsymbol{u}^n + \frac{\Delta t}{2} NS(\boldsymbol{u}^n) \tag{37}$$

$$\boldsymbol{u}^n = \boldsymbol{u}^n + \Delta t NS(\boldsymbol{u}^{n+1/2})$$

where $\boldsymbol{u}$ is the state vector and NS are the right-hand sides of the discretized Navier-Stokes equations. Furthermore, the superscripts refer to the time instances. Written out in detail the full discretization reads

$$\boldsymbol{v}_a^{n+1/2} = \boldsymbol{v}_a^n + \frac{\Delta t}{2}\left(-\frac{1}{\rho_a^n}\mathbf{G}_a(p^n) + \mathbf{L}_a(\nu^n, \boldsymbol{v}^n) + \boldsymbol{g}\right) \tag{38}$$

$$\boldsymbol{r}_a^{n+1/2} = \boldsymbol{r}_a^n + \frac{\Delta t}{2}\boldsymbol{v}_a^n$$

$$\rho_a^{n+1/2} = \rho_a^n - \frac{\Delta t}{2}\rho_a^n D_a(\boldsymbol{v}^n)$$

$$p_a^{n+1/2} = \frac{c_0\rho_0}{\xi}\left[\left(\frac{\rho_a^{n+1/2}}{\rho_0}\right)^\xi - 1\right]$$

$$\boldsymbol{v}_a^{n+1} = \boldsymbol{v}_a^n + \Delta t\left(-\frac{1}{\rho_a^{n+1/2}}\mathbf{G}_a(p^{n+1/2}) + \mathbf{L}_a(\nu^{n+1/2}, \boldsymbol{v}^{n+1/2}) + \boldsymbol{g}\right)$$

$$\boldsymbol{r}_a^{n+1} = \boldsymbol{r}_a^n + \Delta t\,\boldsymbol{v}_a^{n+1/2}$$

$$\rho_a^{n+1} = \rho_a^n - \Delta t\,\rho_a^{n+1/2}D_a(\boldsymbol{v}^{n+1/2})$$

$$p_a^{n+1} = \frac{c_0\rho_0}{\xi}\left[\left(\frac{\rho_a^{n+1}}{\rho_0}\right)^\xi - 1\right]$$

3.3.1   Alternative continuity equation

The density can alternatively be computed by

$$[\rho]_a = \sum_{b\in\mathcal{F}}V_b\rho_b W_{ab} \tag{39}$$

which simplifies to

$$[\rho]_a = \sum_{b\in\mathcal{F}}m_b W_{ab} \tag{40}$$

9

To be consisten with the time-stepping and to preserve initial density values the following time dependent version is used

$$\rho_a^{n+1} = \rho_a^n - \left(\sum_{b \in \mathcal{F}} m_b W_{ab} \right)^n + \left(\sum_{b \in \mathcal{F}} m_b W_{ab} \right)^{n+1} \tag{41}$$

where the sums inside the brackets are with respect to the positions of the particles at the respective time step indicated by the superscript. It can be shown that this is equivalent to the continuity equation based on the divergence (see Vila (1999)). However, as the formulation depends only on the particle position it is numerically more stable.

4 Wall boundary conditions

Several different boundary conditions are available for the SPH method and a few selected ones are implemented in GPUSPH. Currently the following options are implemented

- `LJ_BOUNDARY`: Lennard-Jones boundary conditions

- `MK_BOUNDARY`: Monaghan-Kajtar boundary conditions

- `SA_BOUNDARY`: Semi-analytical boundary conditions

- `DYN_BOUNDARY`: Dynamic boundary conditions

In the following these boundary conditions and their formulations will be described in detail.

4.1 Lennard-Jones boundaries

With this type of boundaries, the walls are discretised with particles and repulsive forces are imposed between boundary particles and fluid particles (that move according to the SPH equations). The repulsive force employed derives from the Lennard-Jones potential (see Monaghan (2005)).
This method is computationally cheap, but may lead to spurious behaviours of the particles close to the walls. Indeed, none of the consistency issues related to boundaries are addressed by this method and although the impermeability of the walls is ensured, the SPH equations are inaccurately solved close to the boundaries. One effect is that the fluid does not remain still near the walls in a hydrostatic case.

4.2 Dynamic boundaries

Section under construction.

4.3 Semi-analytical boundaries

The semi-analytical wall boundary conditions developed by Ferrand et al. (2012) have shown promising results in the simulation of flows with complex boundaries using the Smoothed Particle Hydrodynamics (SPH) method. Recent efforts have pushed these boundary towards practical applications (see Mayrhofer et al. (2014); Leroy et al. (2014)). While the accuracy of these boundary conditions is outstanding, one of their downsides is their comparably high computational cost. The domain Ω
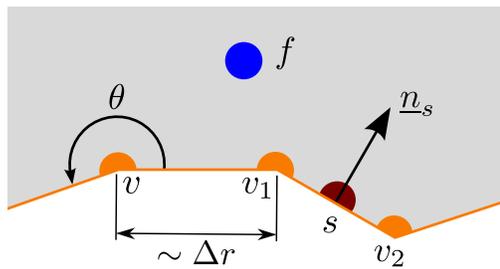

Figure 1: Different particle types

which is the fluid domain is discretized using three different sets of particles:

1. $f \in \mathcal{F}$: the fluid particles,

2. $p \in \mathcal{P}$: the vertex particles,

3. $s \in \mathcal{S}$: the boundary elements.

The boundary segments are triangles, in the 3-D case, that are located at the boundary $\partial\Omega$ of the domain. At each vertex of such a boundary elements is a vertex particle that has a mass that is related to the boundary shape as shown in Fig. 1. These particles, in a finite volume sense, represent the near-wall cells and are moving only if the solid wall is. The fluid particles on the other hand are typical SPH particles that move in a Lagrangian fashion and occupy Ω. The union of fluid and vertex particles will be denoted with $\mathcal{P}$.

11

The SPH interpolation in the context of the semi-analytical boundary conditions is given by

$$[f]_a = \frac{1}{\gamma_a} \sum_{b \in \mathcal{P}} V_b f_b W_{ab} \tag{42}$$

where γ_a is the kernel renormalization parameter defined as

$$\gamma_a = \int_\Omega W(|\boldsymbol{r} - \boldsymbol{r_b}|) \tag{43}$$

which is computed following the idea by Violeau et al. (2014). However, instead of analytically evaluating the integral over the boundary element, a 5th-order quadrature rule is used. Note that currently this is implemented only for the Wendland kernel and so the semi-analytical boundary conditions can only be used using this kernel.

The differential operators gradient and divergence are given as

$$\mathbf{G}_a(p) = \frac{1}{\gamma_a} \sum_{b \in \mathcal{P}} V_b(p_a + p_b)\boldsymbol{\nabla}W_{ab} - \frac{1}{\gamma_a} \sum_{s \in \mathcal{S}} (p_a + p_s)\boldsymbol{\nabla}\gamma_{as} \tag{44}$$

$$D_a(\boldsymbol{v}) = \frac{1}{\gamma_a} \sum_{b \in \mathcal{P}} V_b(\boldsymbol{v_b} - \boldsymbol{v_a}) \cdot \boldsymbol{\nabla}W_{ab} - \frac{1}{\gamma_a} \sum_{s \in \mathcal{S}} (\boldsymbol{v_s} - \boldsymbol{v_a}) \cdot \boldsymbol{\nabla}\gamma_{as} \tag{45}$$

respectively. $\boldsymbol{\nabla}\gamma_{as}$ is the surface integral of the kernel on a triangle s which is solved using the same 5th-order quadrature rule. The derivation of these operators can be found in Ferrand et al. (2012). The basic idea is to not drop the surface integral term in Eq. (21) but instead replace it with a discrete approximation, *i.e.* the sum over the segments. The Laplacian operator, required to discretize the viscous term of the momentum equation, is given by

$$\mathbf{L}_a(\nu, \boldsymbol{v}) = \frac{1}{\gamma_a} \sum_{b \in \mathcal{P}} m_b \left(\frac{\nu_a}{\rho_b} + \frac{\nu_b}{\rho_a} \right) \frac{\boldsymbol{v}_{ab}}{|\boldsymbol{r}_{ab}|^2} \boldsymbol{r}_{ab} \cdot \boldsymbol{\nabla}W_{ab} - \frac{2\nu_a \boldsymbol{v}_a}{\gamma_a} \sum_{s \in \mathcal{S}} \frac{|\boldsymbol{\nabla}\gamma_{as}|}{\delta r_{as}} \tag{46}$$

where the laminar shear stress was used in the boundary term in Ferrand et al. (2012), $\boldsymbol{r}_{ab} = \boldsymbol{r}_a - \boldsymbol{r}_b$ and $\delta r_{as} = \max(\boldsymbol{r}_{as} \cdot \boldsymbol{n}_s, \Delta r)$. Finally the Dirichlet boundary condition $\boldsymbol{v} = 0$ is applied at no-slip boundaries by imposing $\boldsymbol{v}_a = 0 \ \forall a \in \mathcal{V} \cup \mathcal{S}$. The Neumann boundary condition $\partial\rho/\partial\boldsymbol{n} = 0$ is imposed using the approximation

$$\rho_a = \frac{1}{\alpha_a} \sum_{b \in \mathcal{F}} m_b W_{ab} \tag{47}$$

with

$$\alpha_a = \sum_{b \in \mathcal{F}} V_b W_{ab} \tag{48}$$

12

for all vertex particles and boundary elements (for details see Mayrhofer et al. (2013b)).

The alternative continuity equation can also be extended in order to take boundaries into account via

$$\rho_a^{n+1} = \frac{1}{\gamma_a^{n+1}} \left[\gamma_a^n \rho_a^n - \left(\sum_{b \in \mathcal{F}} m_b W_{ab} \right)^n + \left(\sum_{b \in \mathcal{F}} \right] m_b W_{ab} \right)^{n+1} \tag{49}$$

5 Corrections

5.1 Velocity smoothing: XSPH

Under construction

5.2 Sheppard filtering on density

Considering that

$$\sum_{b \in \mathcal{F}} \frac{m_b}{\rho_b} W_{ab} \neq 1$$

the standard SPH summation cannot achieve zeroth order consistency. We can restore zeroth order consistency by using

$$\widetilde{W_{ab}} = \frac{W_{ab}}{\sum\limits_{b \in \mathcal{F}} \frac{m_b}{\rho_b} W_{ab}}$$

instead of W.

Applied to density summation this lead to the Sheppard correction:

$$\rho_a^* = \frac{\sum\limits_{b \in \mathcal{F}} m_b W_{ab}}{\sum\limits_{b \in \mathcal{F}} \frac{m_b}{\rho_b} W_{ab}} \tag{50}$$

In GPUSPH this correction can be periodically enabling Sheppard filtering. **Note**: Sheppard correction leads to a volume increase. Thus this correction should not be used in for long term simulation. In the later it's preferable to use one of stabilizing method described in the next section.

5.3 MLS filtering on density

Under construction

13

6 Stabilizing methods

Due to the collocated nature of the SPH method it is unavoidable that spurious pressure modes are created. In order to prevent the numerical solution from becoming unstable several stabilizing methods can be employed.

6.1 Ferrari

The Ferrari correction is based on the work by Ferrari et al. (2009) and modifies the continuity equation with an additional term given by

$$\frac{\mathrm{d}\rho_a}{\mathrm{d}t} = -\rho_a D_a(\boldsymbol{v}) + \eta_F \sum_{b \in \mathcal{F}} V_b c_{a,b} \frac{\boldsymbol{r}_{ab}}{r_{ab}} \rho_{ab} \cdot \boldsymbol{\nabla} W_{ab} \tag{51}$$

where

$$c_{a,b} = \max(c_a, c_b) \tag{52}$$

and

$$c_a = c_0 \sqrt{\left(\frac{\rho_a}{\rho_0}\right)^{\xi-1}} \tag{53}$$

Note that in the case of the semi-analytical boundaries the formulation is the same and not boundary term needs to be added.

In certain flows (*e.g.* direct numerical simulation of turbulent flow, see Mayrhofer et al. (2013a)) the induced diffusivity can be too high. A more appropriate damping can be introduced by using a damping coefficient η_F different from one. Mayrhofer et al. (2013b) have shown that it should be chosen according to

$$\eta_F = \frac{L}{10^3 \Delta r} \tag{54}$$

where L is a typical length-scale of the flow. In GPUSPH L can be set directly using the `ferrariLengthScale` variable.

6.2 Rhie and Chow

Similar to the Ferrari correction is the correction in the spirit of Rhie and Chow. Again an additional term is added to the continuity equation. However, the density at time $n+1$ is computed first, denoted by $\widetilde{\rho}^{n+1}$, by solving the continuity equation and then a correction is applied to obatin ρ^{n+1}. This correction is given by

$$\rho_a^{n+1} = \eta_{RC} \widetilde{\rho}^{n+1} \left[\mathbf{L}_a \left(\frac{\Delta t}{\widetilde{\rho}}, \widetilde{\rho} \right) - \mathbf{L}_a (\Delta t, \boldsymbol{g} \cdot \boldsymbol{r}) \right] \tag{55}$$

C_μ	σ_k	σ_ϵ	C_{ϵ_1}	C_{ϵ_2}	κ
0.09	1.0	1.3	1.44	1.92	0.41

Table 1: Constants of the $k - \epsilon$ model

where η_{RC} governs the strength of this correction, but is usually equal to one.

7 Turbulence modelling

7.1 The $k - \epsilon$ model

The $k - \epsilon$ model is a Reynolds-Averaged Navier Stokes (RANS) model that uses two additional differential equations to close the equations Pope (2001). The RANS equations modify the momentum equations as follows:

$$\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} = -\frac{1}{\rho}\boldsymbol{\nabla}p + \boldsymbol{\nabla} \cdot ((\nu + \nu_T)\boldsymbol{\nabla} \otimes \boldsymbol{v}) + \boldsymbol{g} \tag{56}$$

where ν_T is the turbulent viscosity. This in turn is given by

$$\nu_T = C_\mu \frac{k^2}{\epsilon} \tag{57}$$

where k is the turbulent kinetic energy and ϵ the turbulent dissipation. The constants, such as C_μ are summarized in Table 1. Both k and ϵ are given by two differential equations:

$$\frac{\mathrm{d}k}{\mathrm{d}t} = P - \epsilon + \nabla \cdot \left[\left(\nu + \frac{\nu_T}{\sigma_k}\right)\boldsymbol{\nabla}k\right] \tag{58}$$

and

$$\frac{\mathrm{d}\epsilon}{\mathrm{d}t} = \frac{\epsilon}{k}\left[C_{\epsilon_1}P - C_{\epsilon_2}\epsilon\right] + \nabla \cdot \left[\left(\nu + \frac{\nu_T}{\sigma_\epsilon}\right)\boldsymbol{\nabla}k\right] \tag{59}$$

where P is the production, given by

$$P = \min(\sqrt{C_\mu}kS, \nu_T S^2), \tag{60}$$

with S is the scalar mean rate-of-strain $S = \sqrt{2\boldsymbol{S} : \boldsymbol{S}}$. Finally, the boundary conditions for k and ϵ at solid walls need to be prescribed. For k a von Neumann boundary condition is imposed

$$\frac{\partial k}{\partial n} = 0. \tag{61}$$

ϵ close to the wall can be approximated via the theoretical relation

$$\epsilon = \frac{u_k^3}{\kappa z} \tag{62}$$

where $u_k = C_\mu^{1/4}\sqrt{k}$ and z is the normal distance to the wall. Clearly, this formula is singular at the wall and ϵ varies strongly when close to the wall. The flow is rather sensitive to this value and thus proper boundary conditions are essential. In the following section about the implementation of the $k - \epsilon$ model with the semi-analytical boundary model this can be taken into account.

7.1.1 Semi-analytical boundaries

The discretization presented in the present section is based on the work by Leroy et al. (2014) and for the exact derivation the interested reader is referred to this paper.

To discretize the equations for k and ϵ the production term requires the computations of the strain rate, which can be achieved by using the gradient operator. However, not the one of Eq. (44) is used, but instead the zero-order accurate version given by

$$\mathbf{G}_a^-(\boldsymbol{v}) = -\frac{1}{\gamma_a}\sum_{b \in \mathcal{P}} V_b \boldsymbol{v}_{ab} \otimes \boldsymbol{\nabla} W_{ab} + \frac{1}{\gamma_a}\sum_{s \in \mathcal{S}} \boldsymbol{v}_{ab} \otimes \boldsymbol{\nabla}\gamma_{as} \tag{63}$$

One important change compared to the laminar simulation is that the vertex particles now also carry a velocity which is evolved according to the viscous term, *i.e.*

$$\frac{\mathrm{d}\boldsymbol{v}_v}{\mathrm{d}t} = \mathbf{L}(\nu + \nu_T, \boldsymbol{v}) \tag{64}$$

The normal component of this velocity is set to zero to avoid particles penetrating the wall. The laplacian of the viscous term presented in Eq. (46) assumes a laminar flow profile and thus needs to be modified for turbulent flows in order to properly take the log law into account. This is achieved by setting the Laplacian to

$$\mathbf{L}_a(\nu, \boldsymbol{v}) = \frac{1}{\gamma_a}\sum_{b \in \mathcal{P}} m_b \left(\frac{\nu_a}{\rho_b} + \frac{\nu_b}{\rho_a}\right) \frac{\boldsymbol{v}_{ab}}{|\boldsymbol{r}_{ab}|^2} \boldsymbol{r}_{ab} \cdot \boldsymbol{\nabla} W_{ab} - \frac{2}{\gamma_a}\sum_{s \in \mathcal{S}} u_{*,as}^2 \boldsymbol{t}_{as} |\boldsymbol{\nabla}\gamma_{as}| \tag{65}$$

where $u_{*,as}$ is the friction velocity at the wall seen by particle a computed iteratively from the implicit equation

$$\frac{\boldsymbol{v}_{as} \cdot \boldsymbol{t}_{as}}{u_{*,as}} = \frac{1}{\kappa}\ln\left(\frac{\delta r_{as} u_{*,as}}{\nu}\right) + 5.2 \tag{66}$$

16

where

$$\boldsymbol{t}_{as} = \frac{\boldsymbol{v}_{as} - (\boldsymbol{v}_{as} \cdot \boldsymbol{n}_s)\boldsymbol{n}_s}{|\boldsymbol{v}_{as} - (\boldsymbol{v}_{as} \cdot \boldsymbol{n}_s)\boldsymbol{n}_s|} \tag{67}$$

The other term in the governing equations for k and ϵ is the Laplacian which are discretized as follows:

$$\mathbf{L}_a\left(\nu + \frac{\nu_T}{\sigma_k}, k\right) = \frac{1}{\gamma_a}\sum_{b \in \mathcal{P}} V_b \left(2\nu + \frac{\nu_{T,a} + \nu_{T,b}}{\sigma_k}\right) \frac{k_{ab}}{r_{ab}^2} \boldsymbol{r}_{ab} \cdot \boldsymbol{\nabla} W_{ab} \tag{68}$$

and

$$\mathbf{L}_a\left(\nu + \frac{\nu_T}{\sigma_\epsilon}, \epsilon\right) = \frac{1}{\gamma_a}\sum_{b \in \mathcal{P}} V_b \left(2\nu + \frac{\nu_{T,a} + \nu_{T,b}}{\sigma_\epsilon}\right) \frac{\epsilon_{ab}}{r_{ab}^2} \boldsymbol{r}_{ab} \cdot \boldsymbol{\nabla} W_{ab} + \frac{4C_\mu}{\sigma_\epsilon \gamma_a}\sum_{s \in \mathcal{S}} \frac{k_a^2}{\delta r_{as}} |\boldsymbol{\nabla} \gamma_{as}| \tag{69}$$

The compatible values on the boundary are given by

$$k_v = \frac{1}{\alpha_v}\sum_{b \in \mathcal{F}} V_b k_b W_{vb} \tag{70}$$

and

$$\epsilon_v = \frac{1}{\alpha_v}\sum_{b \in \mathcal{F}} V_b \left(\epsilon_b + \frac{4C_\mu^{3/4} k_b^{3/2}}{\kappa \delta r_{bv}}\right) W_{vb} \tag{71}$$

where α_v is given according to Eq. (48).

8 Open boundaries

GPUSPH currently implements two different types of open boundaries. One of them is for the dynamic boundary conditions and uses a buffer zone approach. The other can only be used in conjunction with the semi-analytical boundary conditions and uses flat open boundaries, which allows for in- and outflow to be at the same wall (*e.g.* waves).

8.1 Semi-analytical open boundaries

The open boundaries are based around the idea of vertex particles with varying mass. The mass of these vertex particles can change due to three different reasons:

- Flux through the boundary

- Fluid particle crossing boundary

- Creation of new fluid particle

An open boundary has either a prescribed velocity or a prescribed pressure and in the following these two options will be denoted with velocity and pressure boundary, respectively. The quantity that is not prescribed needs to be computed using Riemann invariants which is detailed in the following.

8.1.1 Riemann invariants for velocity boundaries

The imposed normal velocity at the open boundary is denoted with u_{ext}. The normal velocity inside the flow is extrapolated to the wall by

$$u_{int,a} = \frac{1}{\alpha_a} \sum_{b \in \mathcal{F}} V_b \boldsymbol{v}_b \cdot \boldsymbol{n}_a W_{ab} \tag{72}$$

Similary, the pressure is extrapolated according to

$$p_{int,a} = \frac{1}{\alpha_a} \sum_{b \in \mathcal{F}} V_b p_b W_{ab} \tag{73}$$

ρ_{int} can easily be computed using the inverse Equation of State (EOS^{-1}). The aim of this section will be to compute $p_{ext,a}$ from the extrapolated and imposed quantities. The main idea is that there is an internal and an external state with the interface at the open boundary. Due to the discontinuity of the internal and external fields and the assumption of a 1-D problem the Riemann problem is recovered and a solution is known that can be divided into three different states.
Let

$$\psi(\rho) = \frac{2c_0}{\xi - 1} \left(\frac{\rho}{\rho_0} \right)^{\frac{\xi-1}{2}} \text{ if } \xi > 1 \quad \text{or} \quad \psi(\rho) = c_0 \ln \left(\frac{\rho}{\rho_0} \right) \text{ if } \xi = 1 \tag{74}$$

and ψ^{-1} the respective inverse function. The three states yield the following densities

- Expansion wave:
$$\rho_{ext,e} = \psi^{-1}(\psi(\rho_{int}) + u_{ext} - u_{int}) \tag{75}$$

- Shock wave:
$$\rho_{ext,s} = EOS^{-1}(p_{int} + \rho_{int} u_{int}(u_{int} - u_{ext})) \tag{76}$$

- Contact discontinuity
$$\rho_{ext,c} = \rho_{int} \tag{77}$$

18

To decide which state needs to be considered the speed of sound as function of density needs to be defined as

$$c(\rho) = c_0 \left(\frac{\rho}{\rho_0}\right)^{\frac{\xi-1}{2}} \tag{78}$$

to be able to compute the celerities λ. They are given as

$$\lambda = u_{int} + c(\rho_{int}) \tag{79}$$
$$\lambda_e = u_{ext} + c(\rho_{ext,e}) \tag{80}$$
$$\lambda_s = u_{ext} + c(\rho_{ext,s}) \tag{81}$$

Based on these celerities the states occur according to

- $\lambda_e \leq \lambda \Rightarrow$ expansion wave

- $\lambda_s > \lambda \Rightarrow$ shock wave

- $\lambda_e > \lambda \geq \lambda_s \Rightarrow$ contact discontinuity

Depending on the computed state the pressure p_{ext} is set according to the corresponding density in Eqs. (75), (76) or (77).

8.1.2 Riemann invariants for pressure boundaries

Compared to the velocity boundary, this time u_{ext} needs to be computed and p_{ext} is imposed. Similar to the previous section three different fluxes, u_{ext}, can be computed according to the different states

- Expansion wave:
$$u_{ext,e} = u_{int} + \psi(\rho_{ext}) - \psi(\rho_{int}) \tag{82}$$

- Shock wave:
$$u_{ext,s} = u_{int} + \frac{p_{int} - p_{ext}}{\rho_{int} \max(u_{int}, 10^{-5} c_0)} \tag{83}$$

- Contact discontinuity:
$$u_{ext,c} = u_{int} \tag{84}$$

The celerities $\lambda_{\{e,s\}} = u_{ext,\{e,s\}} + c(\rho_{ext})$ can then be used equivalently as above to determine the appropriate state and thus to set u_{ext}.

8.1.3   Mass update

Assuming that both velocity u_{ext} and pressure p_{ext} are known at both segments and vertices of an open boundary the mass update of a vertex particle can be performed. Each segment has three vertices associated with it and similarly, each vertex has a defined set of segments that it is associated with it. The latter set will be denoted with $\mathcal{S}_v$ for a specific vertex v. The principal mass change comes from the flux through segments and reads

$$\widetilde{m}_v = m_v^n + \dot{m}_v \tag{85}$$

where

$$\dot{m}_v = \Delta t \sum_{s \in \mathcal{S}_v} \rho_{ext,s} u_{ext,s} A_s \beta_v(\boldsymbol{r}_s) \tag{86}$$

where A_s represents the area of the segment s and $\beta_v(\boldsymbol{r}_s)$ is the mass repartition factor that will be described below.
Next some mass clippings occur in the sequence listed below

- If no fluid particle is in the support of v and $\dot{m}_v < 0$ then $\widetilde{m}_v = 0$.

- Ensure that $|\widetilde{m}_v| < 2m_{ref}$, where $m_{ref} = \rho_0 \Delta r^3$ is the reference mass.

- If $\dot{m}_v < 0$ ensure that $|\widetilde{m}_v| < m_v^0$, where m_v^0 is the initial mass of the vertex v.

After these clippings have been made, a new fluid particle is created if $\widetilde{m}_v > \frac{1}{2}m_{ref}$. This new fluid particle has exactly the same properties as the vertex particle and a mass equal to the reference mass. Note that this can only happen in the corrector step of the time-stepping scheme. Further conditions are that $u_{ext,v}$ and $p_{ext,v}$ are both greater than zero. If a fluid particle is created than $\widetilde{m}_v$ has m_{ref} subtracted.
If a fluid particle a has crossed a segment $s \in \mathcal{S}_v$ then the mass of the fluid particle is redistributed to the vertex particles associated to that segment. If v is one such segment then

$$\widetilde{m}_v = \widetilde{m}_v + \beta_v(\boldsymbol{r}_a)m_{ref} \tag{87}$$

Finally the new mass of the vertex particle $m_v^{n+1} = \widetilde{m}_v$.

8.1.4   The corners

Vertex particles that are part of the open boundary but have associated segments that are not part of a boundary are labeled as corner vertices. These vertices have several special properties that are detailed below.
First, they do not change their mass and due to that also never create new fluid particles. If a corner vertex is part of a velocity boundary, then both its velocity

and pressure is set to that of the solid wall. If instead the corner vertex is part of a pressure boundary, then the velocity is set to that of the solid wall and the imposed pressure of the open boundary is used.

8.1.5 Modified continuity equation

The open boundaries are restricted to use with the alternative continuity equation given by Eq. (49). In the following $\mathcal{S}^o$ and $\mathcal{V}^o$ denote the segments and vertices respectively that are associated with open boundaries.

$$
\begin{aligned}
\rho_a^{n+1} \quad = \quad & \frac{1}{\gamma_a^{n+1}} \left\{ \gamma_a^n \rho_a^n + \sum_{b \in \mathcal{P}} m_b^n (W_{ab}^{n+1} - W_{ab}^n) + \right. \\
& + \sum_{v \in \mathcal{V}^o} m_v^n \left[W_{av}^n - w \left(r_{av}^n + \boldsymbol{\delta r}_v^o(\Delta t) \right) \right] \\
& \left. + \frac{\rho_a^n}{2} \sum_{s \in \mathcal{S}^o} \left[\boldsymbol{\nabla} \gamma_{as} \left(r_{as}^n + \boldsymbol{\delta r}_s^o(\Delta t) \right) + \boldsymbol{\nabla} \gamma_{as}(r_{as}^n) \right] \cdot \boldsymbol{\delta r}_s^o(\Delta t) \right\}
\end{aligned}
\tag{88}
$$

where

$$
\boldsymbol{\delta r}_a^o(\Delta t) = \Delta t (\boldsymbol{u}_a^n + \boldsymbol{v}_a^n)
\tag{89}
$$

where $\boldsymbol{u}$ and $\boldsymbol{v}$ are the Eulerian and Lagrangian velocity, respectively.

21

8.1.6 Time integration

The full time-stepping scheme including for open boundaries reads

$$
\boldsymbol{v}_a^{n+1/2} = \boldsymbol{v}_a^n + \frac{\Delta t}{2}\left(-\frac{1}{\rho_a^n}\mathbf{G}_a(p^n) + \mathbf{L}_a(\nu^n, \boldsymbol{v}^n) + \boldsymbol{g}\right) \tag{90}
$$

$$
\boldsymbol{r}_a^{n+1/2} = \boldsymbol{r}_a^n + \frac{\Delta t}{2}\boldsymbol{v}_a^n
$$

$$
\gamma_a^{n+1/2} = \gamma_a^n + \frac{1}{2}(\boldsymbol{r}_a^{n+1/2} - \boldsymbol{r}_a^n)\cdot(\boldsymbol{\nabla}\gamma_a^n + \boldsymbol{\nabla}\gamma_a^{n+1/2})
$$

$$
\rho_a^{n+1/2} = \frac{1}{\gamma_a^{n+1/2}}\left\{\gamma_a^n\rho_a^n + \sum_{b\in\mathcal{P}}m_b^n(W_{ab}^{n+1/2} - W_{ab}^n) + \right.
$$

$$
+ \sum_{v\in\mathcal{V}^o}m_v^n\left[W_{av}^n - w\left(\boldsymbol{r}_{av}^n + \boldsymbol{\delta r}_v^o(\Delta t/2)\right)\right]
$$

$$
\left.+ \frac{\rho_a^n}{2}\sum_{s\in\mathcal{S}^o}\left[\boldsymbol{\nabla}\gamma_{as}\left(\boldsymbol{r}_{as}^n + \boldsymbol{\delta r}_s^o(\Delta t/2)\right) + \boldsymbol{\nabla}\gamma_{as}(\boldsymbol{r}_{as}^n)\right]\cdot\boldsymbol{\delta r}_s^o(\Delta t/2)\right\}
$$

$$
p_a^{n+1/2} = \frac{c_0\rho_0}{\xi}\left[\left(\frac{\rho_a^{n+1/2}}{\rho_0}\right)^\xi - 1\right]
$$

Boundary conditions & mass update

$$
\boldsymbol{v}_a^{n+1} = \boldsymbol{v}_a^n + \Delta t\left(-\frac{1}{\rho_a^{n+1/2}}\mathbf{G}_a(p^{n+1/2}) + \mathbf{L}_a(\nu^{n+1/2}, \boldsymbol{v}^{n+1/2}) + \boldsymbol{g}\right)
$$

$$
\boldsymbol{r}_a^{n+1} = \boldsymbol{r}_a^n + \Delta t\,\boldsymbol{v}_a^{n+1/2}
$$

$$
\gamma_a^{n+1} = \gamma_a^n + \frac{1}{2}(\boldsymbol{r}^{n+1} - \boldsymbol{r}_a^n)\cdot(\boldsymbol{\nabla}\gamma_a^n + \boldsymbol{\nabla}\gamma_a^{n+1})
$$

$$
\rho_a^{n+1} = \frac{1}{\gamma_a^{n+1}}\left\{\gamma_a^n\rho_a^n + \sum_{b\in\mathcal{P}}m_b^n(W_{ab}^{n+1} - W_{ab}^n) + \right.
$$

$$
+ \sum_{v\in\mathcal{V}^o}m_v^n\left[W_{av}^n - w\left(\boldsymbol{r}_{av}^n + \boldsymbol{\delta r}_v^o(\Delta t)\right)\right]
$$

$$
\left.+ \frac{\rho_a^n}{2}\sum_{s\in\mathcal{S}^o}\left[\boldsymbol{\nabla}\gamma_{as}\left(\boldsymbol{r}_{as}^n + \boldsymbol{\delta r}_s^o(\Delta t)\right) + \boldsymbol{\nabla}\gamma_{as}(\boldsymbol{r}_{as}^n)\right]\cdot\boldsymbol{\delta r}_s^o(\Delta t)\right\}
$$

$$
p_a^{n+1} = \frac{c_0\rho_0}{\xi}\left[\left(\frac{\rho_a^{n+1}}{\rho_0}\right)^\xi - 1\right]
$$

Boundary conditions & mass update

Create and delete particles if required

9 Solid bodies

GPUSPH fully integrates moving and floating bodies. A moving body as an user prescribed by user while the movement of a floating body is controlled by the forces acting on it.

9.1 Reference frame and orientation

Under construction.

9.2 Moving bodies

Under construction

9.3 Floating bodies

All the floating bodies dynamics is solved using Chrono engine. Basically GPUSPH computes force and torque on each body and feed them to Chrono engine that integrate the movement.
Details under construction.

References

Martin Ferrand, Dominique Laurence, Benedict D. Rogers, Damien Violeau, and Christophe Kassiotis. Unified semi-analytical wall boundary conditions for inviscid, laminar or turbulent flows in the meshless SPH method. *International Journal for Numerical Methods in Fluids*, 71(4):446–472, 2012. doi: 10.1002/fld.3666. DOI: 10.1002/fld.3666.

A. Ferrari, M. Dumbser, E. F. Toro, and A. Armanini. A new 3D parallel SPH scheme for free surface flows. *Computers & Fluids*, 36(6):1203–1217, 2009.

A. Leroy, D. Violeau, M. Ferrand, and C. Kassiotis. Unified semi-analytical wall boundary conditions applied to 2-D incompressible SPH. *Journal of Computational Physics*, 261:106–129, 2014. ISSN 0021-9991. doi: 10.1016/j.jcp.2013.12.035.

A. Mayrhofer, D. Laurence, B. D. Rogers, D. Violeau, and M. Ferrand. Direct numerical simulation of 3-D turbulent wall bounded flows with SPH. In *Proc. 8th international SPHERIC workshop*, pages 130–138, 2013a.

Arno Mayrhofer, Benedict D. Rogers, Damien Violeau, and Martin Ferrand. Investigation of wall bounded flows using SPH and the unified semi-analytical wall boundary conditions. *Computer Physics Communications*, 184(11):2515–2527, November 2013b. ISSN 0010-4655. doi: 10.1016/j.cpc.2013.07.004.

Arno Mayrhofer, Martin Ferrand, Christophe Kassiotis, Damien Violeau, and François-Xavier Morel. Unified semi-analytical wall boundary conditions in SPH: analytical extension to 3-{D}. *Numerical Algorithms*, pages 1–20, February 2014. ISSN 1017-1398, 1572-9265. doi: 10.1007/s11075-014-9835-y.

J. J. Monaghan. Smoothed particle hydrodynamics. *Reports on Progress in Physics*, 68:1703–1759, 2005.

Stephen B Pope. *Turbulent Flows*. Cambridge University Press, 7 edition, 2001. URL http://stacks.iop.org/0957-0233/12/i=11/a=705.

J. P. Vila. On particle weighted methods and smooth particle hydrodynamics. *Mathematical Models and Methods in Applied Sciences*, 9(2):161–209, 1999.

Damien Violeau, Agnès Leroy, and Arno Mayrhofer. Exact computation of SPH wall renormalising integrals in 3-D. In *Proceedings of the 9th International SPHERIC Workshop*, pages 95–102, Paris, May 2014.