DualSPHysics: from fluid dynamics to multiphysics problems

(Article begins on next page)

DualSPHysics: from fluid dynamics to multiphysics problems

J.M. Domínguez[1*], G. Fourtakas[2], C. Altomare[3,4], R.B. Canelas[5], A. Tafuni[6], O. García-Feal[1], I. Martínez-Estévez[1], A. Mokos[7], R. Vacondio[8], A.J.C. Crespo[1], B.D. Rogers[2], P.K. Stansby[2], M. Gómez-Gesteira[1]

[1]*EPHYSLAB Environmental Physics Laboratory, CIM-UVIGO, Universidade de Vigo, Spain*

[2]*Department of Mechanical, Aerospace and Civil Engineering (MACE), University of Manchester, United Kingdom*

[3]*Maritime Engineering Laboratory, Universitat Politècnica de Catalunya – BarcelonaTech, Barcelona, Spain*

[4]*Ghent University, Department of Civil Engineering, Technologiepark 60, 9052 Gent, Belgium*

[5]*Bentley Systems, Lisbon, Portugal*

[6]*School of Applied Engineering and Technology, New Jersey Institute of Technology, Newark, US*

[7]*Saint-Venant Hydraulics Laboratory, Ecole des Ponts ParisTech, Paris, France*

[8]*Department of Engineering and Architecture, University of Parma, Parma, Italy*

*Corresponding author. E-mail address: jmdominguez@uvigo.es, ORCID: 0000-0002-2586-5081

Abstract

DualSPHysics is a weakly compressible smoothed particle hydrodynamics (SPH) Navier-Stokes solver initially conceived to deal with coastal engineering problems, especially those related to wave impact with coastal structures. Since the first release back in 2011, DualSPHysics has shown to be robust and accurate for simulating extreme wave events along with a continuous improvement in efficiency thanks to the exploitation of hardware such as graphics processing units (GPUs) for scientific computing or the coupling with wave propagating models such as SWASH and OceanWave3D. Numerous additional functionalities have also been included in the DualSPHysics package over the last few years which allow the simulation of fluid-driven objects. The use of the discrete element method (DEM) has allowed the solver to simulate the interaction among different bodies (sliding rocks, for example), which provides a unique tool to analyse debris flows. In addition, the recent coupling with other solvers like Project Chrono or MoorDyn has been a milestone in the development of the solver. Project Chrono allows the simulation of articulated structures with joints, hinges, sliders and springs and MoorDyn allows simulating moored structures. Both functionalities make DualSPHysics especially suited for the simulation of offshore energy harvesting devices. Lately, the present state of maturity of the solver goes beyond single phase simulations, allowing multi-phase simulations with gas-liquid and a combination of Newtonian and non-Newtonian models expanding further the capabilities and range of applications for the DualSPHysics solver. These advances and functionalities make DualSPHysics an advanced meshless solver with emphasis on free-surface flow modelling.

Keywords: Meshfree, Lagrangian, SPH, Particles, DualSPHysics, Navier-Stokes equations, Free-surface flows

1. Introduction

The meshless method smoothed particle hydrodynamics (SPH) has made progress over the past two decades with contributions to the method continually appearing. Modern SPH schemes enable the simulation of violent free-surface flows with improved accuracy, while new physical processes can be included with relative ease accompanied by a significant reduction of the computational time achieved through hardware acceleration.

The development of SPH enables the simulation of different physical phenomena, including violent hydrodynamics of coastal/offshore waves impacting structures, galaxy and planetary formation and evolution, flow in process industries characterised by multiple phases and mixing, large deformations of solids and structures, modelling damage and failure, including opening and propagation of cracks, fluid-structure interaction, and exploitation in computer graphics and computer games (see [1–4] for recent examples). The most relevant advantages over Eulerian methods [5] are the exact conservation of mass and momentum and the meshless properties. Despite these advances, some key challenges still remain [6], and are encapsulated in the SPHERIC Grand Challenges (https://spheric-sph.org/grand-challenges).

With the large numbers of neighbouring particle interactions and the time step restricted by the Courant condition for weakly-compressible flow, SPH is computationally demanding and there have been multiple efforts within the broad SPH community to accelerate the simulations. Massive parallelisation is the traditional approach where machines with tens of thousands of processing cores are used in a combination of distributed and shared memory frameworks [7–9]. In the past 10 years, the exploitation of hardware such as graphics processing units (GPUs) for scientific computing has gained traction. SPH is particularly suitable to graphics processing unit (GPU) acceleration thanks to its vector lists of particles and their interactions. Multiple GPU-based SPH codes now exist [10–12].

The open-source code DualSPHysics is being developed precisely to address these issues. In a collaborative approach, the Universidade de Vigo, University of Manchester, University of Lisbon, Università di Parma, Flanders Hydraulics Research, Universitat Politécnica de Catalunya, and New Jersey Institute of Technology have developed DualSPHysics, which aims to exploit the computational power of modern GPUs and enables simulation of real engineering problems using a standard desktop PC. The open-source code with pre- and post-processing tools and examples is freely available at https://dual.sphysics.org/ and is released under GNU Lesser General Public License (LGPL). The source code of DualSPHysics is also available in the public collaborative development platform GitHub (https://github.com/DualSPHysics/DualSPHysics).

DualSPHysics' name derives from being able to run on both central processing unit (CPU) using the shared memory OpenMP approach, or on a GPU. The main thrust of the DualSPHysics development has been to exploit the computational acceleration using the compute unified device architecture (CUDA) programming framework on Nvidia GPUs. One of the guiding philosophies for this dual functionality is that new features are developed first in C++ language for the CPU version, and once validated they are more easily ported to a GPU using CUDA. This allows new developers to implement their own customized solutions for CPU or GPU depending on their programming skills and performance needs. This approach also enables engineers to focus on developing new functionalities, while computer scientists can work on ensuring computational efficiency of the GPU version. The code has optimised CUDA kernels for a single GPU, it uses hierarchical templates and optimised cell-linked neighbour lists [13] to achieve maximum code flexibility and processing speed [14]. A multi-GPU implementation of DualSPHysics has also been developed [15], but this feature is not included in the public version of DualSPHysics 5.0 since it is not yet supported by the latest functionalities.

DualSPHysics comes with dedicated pre- and post-processing tools. In particular, the GenCase application is used to generate all the necessary inputs. In the XML input files constants, geometries, and parameters are specified. Recently, a graphical user interface (GUI) plugin for the open-source

software FreeCAD, namely DesignSPHysics (https://design.sphysics.org/), has made the workflow for the generation of new test cases more straightforward and accessible. Different post-processing tools are available inside the DualSPHysics package, either to visualise the simulation using any scientific visualisation software such as Paraview or to extract computed quantities of interest (vorticity, surface elevation, forces, etc.).

The DualSPHysics code has evolved continuously since its first release in 2011. The most recent release (version 5.0, July 2020) includes the following novelties: (i) Coupling with the discrete element method and Project Chrono (https://projectchrono.org/); (ii) Coupling with MoorDyn (http://www.matt-hall.ca/moordyn.html); (iii) a multi-phase solver (soil-water and gas-liquid); (iv) specialised functionality for second-order wave generation and active wave absorption, plus relaxation zones; (v) coupling with wave propagation models; and (vi) inlet/outlet flow conditions. Some limitations remain related to the SPHERIC Grand Challenges (GC3: Adaptivity, GC4: Coupling, GC5: Applicability to industry), which are the focus of ongoing research.

The aim of this paper is to present the novel developments of the DualSPHysics software, emphasising the process of validation that is undertaken before software releases. The paper is organised as follows: Section 2 describes the core SPH formulation. Section 3 is dedicated to the description of different features: wave generation, fluid-structure interaction, inflow-outflow, and multi-phase formulations. Section 4 presents one of the key developments of coupling with other models, including wave propagation models, DEM, the physics engine Project Chrono and the library MoorDyn. Section 5 presents the new code implementation, which has changed significantly over the lifespan of the code. In Section 6 validations and applications are presented. Finally, the current challenges are briefly discussed in Section 7 in the context of ever-changing hardware, user demands and ongoing research.

2. SPH formulation

2.1. Interpolants and kernel functions

The integral (continuous) representation of SPH in a domain Ω for a smooth function $f(\mathbf{r}), \mathbf{r} \in \mathbb{R}^d$ can be written as the convolution a kernel function W and f

$$\langle f(\mathbf{r}) \rangle := \int_{\Omega} f(\mathbf{r}')W(\mathbf{r}-\mathbf{r}',h)d\mathbf{r}', \tag{1}$$

where the $\langle \cdot \rangle$ brackets denote an approximation, $\mathbf{r}' \in \Omega$ is a temporary position variable and $W(\mathbf{r}, h)$ is a positive function with the general form of,

$$W(\mathbf{r},h) := \frac{1}{h^d}\omega(q), \tag{2}$$

with $h > 0$ and $q = |\mathbf{r}|/h$. The function $\omega: \mathbb{R} \to \mathbb{R}$ is a smooth non-negative function such that,

$$\int_{\Omega} \omega(q)d\mathbf{r} = \frac{1}{h^d}, \tag{3}$$

with compact support (see Figure 1),

$$\omega(q) = 0 \text{ for } |\mathbf{r}| \geq kh, \ k \in \square^{+}. \tag{4}$$
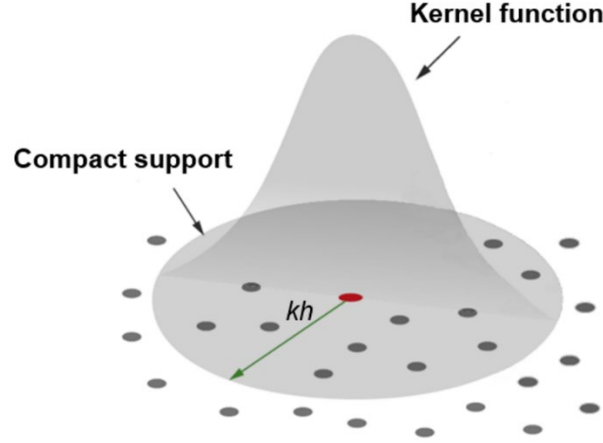
Figure 1. Shape and compact support of the kernel function.

Several smoothing kernels have been proposed in the literature as summarised in [16]. The smoothing kernel or simply kernel takes the form of Eq. (2) and satisfies Eqs. (3) and (4), with $W: \mathbb{R}^d \to \mathbb{R}$. Moreover, it is positively defined within the support domain $\{\forall \mathbf{r}' \in \Omega; W(\mathbf{r} - \mathbf{r}', h) > 0\}$, it is a monotonic decreasing sufficiently smooth function defined over the interval kh and,

$$\lim_{h \to 0} W(\mathbf{r}, h) = \delta(\mathbf{r}), \tag{5}$$

where δ is the Dirac delta function [17, 18]. Several kernel functions have been implemented in DualSPHysics.

For example the third order B-splines kernel (cubic spline) has been widely used [18]

$$\omega(q) = a_D \begin{cases} 1 - \dfrac{3}{2}q^2 + \dfrac{3}{4}q^3 & 0 \le q \le 1 \\ \dfrac{1}{4}(2-q)^3 & 1 < q \le 2 \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

where α_D^d is $10/7\pi$ and $1/\pi$ in 2-D and 3-D, respectively.

The fifth order Wendland function is now popular, characterised by reduced pairing instability [19]. This C^2 kernel exhibits a positive kernel Fourier transform [16, 20] , and reads

$$\omega(q) = a_D \left(1 - \frac{q}{2}\right)^4 (2q + 1) \qquad 0 \le q \le 2, \tag{7}$$

where α_D^d is $7/4\pi$ and $21/16\pi$ in 2-D and 3-D, respectively.

The discrete convolution of W and f, defined herein as the SPH discrete approximation, is obtained by

$$\langle f(\mathbf{r}_a) \rangle := \sum_{b \in P} f(\mathbf{r}_b) W(\mathbf{r}_a - \mathbf{r}_b, h) \Delta r_b^d, \quad a = 1, \cdots, N, \tag{8}$$

where the subscripts $a, b \in P = \{1, \cdots, N\}$ denote the interpolating and neighbouring discrete particles, while Δr_b^d is the associated volume V of the b-th particle in P. The $\langle \cdot \rangle$ brackets will be dropped henceforth for simplicity. The set P contains all particles within the domain i.e. fluid F, boundary B and floating objects K as shown in Figure 2 with $P = F \cup B$ and $K \subset B$.
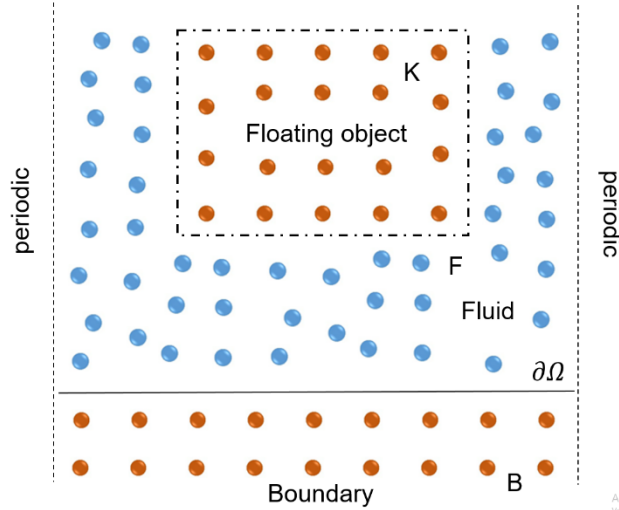
4

Figure 2. Sketch of different sets of particles $a \in \mathrm{P}$, where the set P contains all particles within the domain.

For a more complete description of SPH, the reader is referred to [21, 22] and a recent review in [5]. Similar integral and discrete expressions can be formulated for a gradient operator for a given function f by,

$$\frac{\partial f(\mathbf{r})}{\partial \mathbf{r}} = \int_{\Omega} f(\mathbf{r}') \frac{\partial W}{\partial \mathbf{r}} (\mathbf{r} - \mathbf{r}', h) d\mathbf{r}', \tag{9}$$

by employing the convolution identity,

$$\frac{\partial f}{\partial \mathbf{r}} * W = f * \frac{\partial W}{\partial \mathbf{r}}. \tag{10}$$

The discrete gradient takes the following form,

$$\frac{\partial}{\partial \mathbf{r}} f(\mathbf{r}_a) := \sum_{b \in \mathrm{P}} f(\mathbf{r}_b) \frac{\partial}{\partial \mathbf{r}} W(\mathbf{r}_a - \mathbf{r}_b, h) V_b, \quad a = 1, \cdots, N. \tag{11}$$

2.2. Governing equations

The governing equations are described by the Navier-Stokes equations for a compressible fluid. The continuity and momentum equations in Lagrangian form can be written as,

$$\frac{d\rho}{dt} = -\rho \nabla \cdot \mathbf{v}, \tag{12}$$

and

$$\frac{d\mathbf{v}}{dt} = -\frac{1}{\rho} \nabla P + \Gamma + \mathbf{f}, \tag{13}$$

respectively, where d denotes the total or material derivative, $\mathbf{v}$ is the velocity vector, ρ is density, P is pressure, Γ denotes the dissipation terms and $\mathbf{f}$ represents accelerations due to external forces, such as the gravity.

2.3. SPH discretisation of the governing equations

The mass conservation property of SPH,

$$\frac{dm}{dt} = 0 \,, \tag{14}$$

where the mass is conserved exactly within a Lagrangian particle, results in a density change due to the volumetric change of the term at the right-hand side of Eq. (12). In the SPH formalism, the discrete form of the continuity equation at a point a with position $\mathbf{r}_a$ reads [21],

$$\left. \frac{d\rho}{dt} \right|_{a \in \mathrm{P}} = \rho_a \sum_{b \in \mathrm{P}} \frac{m_b}{\rho_b} \mathbf{v}_{ab} \cdot \nabla_a W_{ab} + D_a \,, \tag{15}$$

where $W_{ab} = W(\mathbf{r}_a - \mathbf{r}_b, h)$ and $(\cdot)_{ab} = (\cdot)_a - (\cdot)_b$. The second term in the right-hand side represents a numerical density diffusion term [23].

The discrete form of Eq. (13) in a SPH formalism reads

$$\left. \frac{d\mathbf{v}}{dt} \right|_{a \in \mathrm{F}} = -\sum_{b \in \mathrm{P}} m_b \left(\frac{P_a + P_b}{\rho_a \rho_b} \right) \nabla W_{ab} + \left\langle \Gamma \right\rangle_a + \mathbf{f} \,, \tag{16}$$

where a symmetric SPH operator has been used that guarantees conservation of momentum [24] for the pressure term and the discrete form of the dissipation terms is discussed in Section 2.3.2.

2.3.1. Density diffusion terms

Two density diffusion terms (DDT) formulations are implemented in DualSPHysics which act as a high frequency numerical noise filter improving the stability of the scheme by smoothing the density and consequently the pressure. The latter is known to be spurious due to the collocated (in velocity and density) and explicit (time integration) nature of weakly compressible SPH. These terms take the general form of

$$D_a = \delta h c_a \sum_{b \in \mathrm{F}} \psi_{ab} \cdot \nabla W_{ab} V_b \,, \tag{17}$$

where δ controls the magnitude of the diffusion term. The term Ψ_{ab} is based on the Neumann–Richtmeyer artificial dissipation. Molteni and Colagrossi [25] introduced this artificial dissipation term as

$$\psi_{ab} = 2 \left(\rho_b - \rho_a \right) \frac{\mathbf{r}_{ab}}{\left\| r_{ab} \right\|^2} \,. \tag{18}$$

Lately, Fourtakas et al. [26] modified Eq. (18) by accounting for the dynamic component of the density by,

$$\psi_{ab} = 2 \left(\rho_{ba}^{\mathrm{T}} - \rho_{ab}^{\mathrm{H}} \right) \frac{\mathbf{r}_{ab}}{\left\| r_{ab} \right\|^2} \,, \tag{19}$$

where superscripts $(\cdot)^T$ and $(\cdot)^H$ denote the total and hydrostatic component of the density of a weakly compressible and barotropic fluid by locally constructing the hydrostatic pressure as,

$$P_{ab}^{\mathrm{H}} = \rho_0 g z_{ab} \,. \tag{20}$$

Although both terms are not consistent near a truncated kernel support [23] (such as free-surface or a wall boundary), thus resulting in an outwards vector net contribution, the latter term improves the behaviour of pressure near the wall boundaries by operating on the dynamic pressure.

2.3.2. Dissipation terms

2.3.2.1. Artificial viscosity

An artificial diffusive term can be used in the momentum equation [27] based on the Neumann–Richtmeyer artificial viscosity, aiming to reduce oscillations and stabilise the SPH scheme. The artificial viscosity term is added in the SPH gradient operator in right hand side of Eq. (16) and reads,

$$
\Pi_{ab} = \begin{cases} \left(\dfrac{-\alpha \overline{c_{ab}}}{\overline{\rho_{ab}}} \right) \left(\dfrac{h \mathbf{v}_{ab} \cdot \mathbf{r}_{ab}}{r_{ab}^2 + \eta^2} \right) & \mathbf{v}_{ab} \cdot \mathbf{r}_{ab} < 0 \\ 0 & \mathbf{v}_{ab} \cdot \mathbf{r}_{ab} \geq 0 \end{cases}, \tag{21}
$$

where $\overline{(\cdot)_{ab}} = [(\cdot)_a - (\cdot)_b]/2$. Herein, c denotes the numerical speed of sound (see Section 2.4) and α is the artificial viscosity coefficient. The parameter $\eta = 0.001 h^2$, with $\{\eta \in \mathbb{R}; r_{ab} > \eta\}$ guarantees a non-singular operator. It can be shown that $\Pi_{ab} \propto \upsilon_0 \nabla^2 \mathbf{v}$, where υ_0 is the kinematic viscosity [21, 28]. The final form of the momentum equation in the presence of artificial viscosity reads,

$$
\left. \frac{d\mathbf{v}}{dt} \right|_{a \in F} = -\sum_{b \in P} m_b \left(\frac{P_a + P_b}{\rho_a \rho_b} + \Pi_{ab} \right) \nabla W_{ab} + \mathbf{f}, \tag{22}
$$

Due to its simplicity, the artificial viscosity formulation is commonly used in SPH as a viscous dissipation term.

2.3.2.2. Laminar viscosity

Viscous dissipation of momentum in the laminar regime in DualSPHysics is approximated by Lo and Shao [29],

$$
\upsilon_0 \nabla^2 \mathbf{v}_a = \sum_{b \in P} m_b \frac{4 \upsilon_0 \mathbf{r}_{ab} \cdot \nabla W_{ab}}{(\rho_a + \rho_b)(r_{ab}^2 + \eta^2)} \mathbf{v}_{ab}, \tag{23}
$$

where υ_0 denotes the kinematic viscosity of the fluid. Note that the laminar viscous term diverges close the free-surface as discussed in [30].

2.3.2.3. Sub-particle scale model

The large eddy simulation sub-particle scale model (SPS) [31] as described by Dalrymple and Rogers [32] using Favre averaging in a weakly compressible approach is implemented in DualSPHysics. The SPS stress tensor τ is defined in Einstein notation over superscripts i, j according to

$$
\tau^{ij} = \overline{v^i v^j} - \overline{v^i} \, \overline{v^j}, \tag{24}
$$

modelled by an eddy viscosity closure as,

$$
\frac{\tau^{ij}}{\rho} = 2 \upsilon_{SPS} \left(S^{ij} - \frac{1}{3} S^{ii} \delta^{ij} \right) - \frac{2}{3} C_l \Delta^2 \delta_{ij} \left| S^{ij} \right|^2. \tag{25}
$$

Here, $\upsilon_{SPS} = [C_S \Delta]^2 \left| S^{ij} \right|^2$, where $C_S = 0.12$ is the Smagorinsky constant $C_l = 0.0066$, Δ is the initial particle spacing, and $\left| S^{ij} \right| = 1/2 \left(2 S^{ij} S^{ij} \right)^{1/2}$, where S^{ij} is an element of the SPS strain tensor. Using

the same variationally-consistent form of the pressure gradient (Eq. 16), the discrete form of the term reads [32],

$$\frac{1}{\rho}\nabla\cdot\tau_a^{ij}=\sum_{b\in P}m_b\left(\frac{\tau_a^{ij}+\tau_b^{ij}}{\rho_a\rho_b}\right)\nabla^i W_{ab}\ .$$

(26)

Finally, the momentum dissipation term in DualSPHysics takes the following form

$$\Gamma_a=\sum_{b\in P}m_b\frac{4\upsilon_0\mathbf{r}_{ab}\cdot\nabla W_{ab}}{\left(\rho_a+\rho_b\right)\left(r_{ab}^2+\eta^2\right)}\mathbf{v}_{ab}+\sum_{b\in P}m_b\left(\frac{\tau_a^{ij}+\tau_b^{ij}}{\rho_a\rho_b}\right)\nabla^i W_{ab}\ .$$

(27)

2.4.   Equation of state and compressibility

In the SPH formulation implemented in DualSPHysics, density and pressure are coupled by means of an equation of state (EOS) allowing for weak compressibility of the fluid based on the numerical speed of sound [33],

$$P=\frac{c_s^2\rho_0}{\gamma}\left[\left(\frac{\rho}{\rho_0}\right)^\gamma-1\right],$$

(28)

where γ is the polytropic index (usually 7 for water), ρ_0 is the reference density, and the numerical speed of sound is defined as $c_s=\sqrt{\partial P/\partial\rho}$ [34].

A numerical speed of sound c_s is chosen based on a typical length scale and time scale of the domain which allows for much larger time steps within the explicit time integration than would be possible with a physical speed of sound [33]. With $c_s=10\|\mathbf{v}\|_{max}$ where $\|\mathbf{v}\|_{max}=\sqrt{gh_o}$ with h_o the initial fluid height in the domain a variation in density of up to about 1% generally occurs. This should however be monitored as there are special cases where 1% may be exceeded and compressibility may cease to be 'weak'. In such cases the speed of sound should be increased with an inevitable decrease in time step and increase in computational time.

2.5.   Time integrators and time step

In DualSPHysics, two explicit time integration schemes are implemented. For brevity, the governing equations are written as

$$\frac{d\mathbf{v}_a}{dt}=F_a;\ \frac{d\rho_a}{dt}=R_a;\ \frac{d\mathbf{r}_a}{dt}=\mathbf{v}_a\ .$$

(29)

The time integrators are briefly introduced below.

2.5.1. Verlet time integration scheme

Verlet schemes [35] are commonly used in molecular dynamics providing low computational cost with a second order accurate space integrator. We use a velocity Verlet version that does not require multiple calculation steps within an iteration interval, the WCSPH variables are calculated according to,

$$\mathbf{v}_a^{n+1}=\mathbf{v}_a^{n-1}+2\Delta t\mathbf{F}_a^n;\quad\mathbf{r}_a^{n+1}=\mathbf{r}_a^n+\Delta t\mathbf{v}_a^n+\frac{1}{2}\Delta t^2\mathbf{F}_a^n;$$

$$\rho_a^{n+1}=\rho_a^{n-1}+2\Delta tR_a^n$$

(30)

Due to the integration over a staggered time interval, the equations of density and velocity are decoupled, which may lead to divergence of the integrated values. Therefore, an intermediate step is required every N_S steps ($N_S \approx 40$ is recommended) according to

$$\mathbf{v}_a^{n+1} = \mathbf{v}_a^n + \Delta t \mathbf{F}_a^n, \qquad \mathbf{r}_a^{n+1} = \mathbf{r}_a^n + \Delta t \mathbf{v}_a^n + \frac{1}{2}\Delta t^2 \mathbf{F}_a^n,$$
$$\rho_a^{n+1} = \rho_a^n + \Delta t R_a^n \tag{31}$$

where the superscript $n \in \mathbb{N}$ and denotes the time step and $t = n\Delta t$.

2.5.2. Symplectic time integration scheme

The symplectic position Verlet time integrator scheme [36] is second order accurate in time. It is ideal for Lagrangian schemes as it is time reversible and symmetric in the absence of diffusive terms that preserve geometric futures. The position Verlet scheme in the absence of dissipation forces reads,

$$\mathbf{r}_a^{n+\frac{1}{2}} = \mathbf{r}_a^n + \frac{\Delta t}{2}\mathbf{v}_a^n;$$
$$\mathbf{v}_a^{n+1} = \mathbf{v}_a^n + \Delta t \mathbf{F}_a^{n+\frac{1}{2}};$$
$$\mathbf{r}_a^{n+1} = \mathbf{r}_a^{n+\frac{1}{2}} + \frac{\Delta t}{2}\mathbf{v}_a^{n+1}. \tag{32}$$

However, in the presence of viscous forces and density evolution in DualSPHysics, the velocity is required at the $(n + 1/2)$ step thus, a velocity Verlet half step is used to compute the required velocity for the acceleration and density evolution for $\mathbf{F}(\mathbf{r}_{n+1/2})$ and $R(\mathbf{r}_{n+1/2})$, respectively. The scheme implemented in DualSPHysics reads,

$$\mathbf{r}_a^{n+\frac{1}{2}} = \mathbf{r}_a^n + \frac{\Delta t}{2}\mathbf{v}_a^n$$
$$\mathbf{v}_a^{n+\frac{1}{2}} = \mathbf{v}_a^n + \frac{\Delta t}{2}\mathbf{F}_a^n$$
$$\mathbf{v}_a^{n+1} = \mathbf{v}_a^n + \Delta t \mathbf{F}_a^{n+\frac{1}{2}} \tag{33}$$
$$\mathbf{r}_a^{n+1} = \mathbf{r}_a^n + \Delta t \frac{\left(\mathbf{v}_a^{n+1} + \mathbf{v}_a^n\right)}{2}$$

where $\mathbf{r}^{n+1/2}$ is substituted to $\mathbf{r}^{n+1}$ in Eq. (32) to eliminate dependence from $\mathbf{u}^{n+1/2}$. Finally, the density evolution follows the half time steps of the symplectic position Verlet scheme as follows [37],

$$\rho_a^{n+\frac{1}{2}} = \rho_a^n + \frac{\Delta t}{2} R_a^n$$
$$\rho_a^{n+1} = \rho_a^n \frac{2 - \varepsilon_a^{n+\frac{1}{2}}}{2 + \varepsilon_a^{n+\frac{1}{2}}}, \tag{34}$$

where $\varepsilon_a^{n+1/2} = -\left(R_a^{n+\frac{1}{2}}/\rho_a^{n+1/2}\right)\Delta t$.

2.5.3. Variable time step

The time integration is bounded by the Courant–Friedrich–Levy (CFL) condition necessary in explicit time integration schemes to limit the numerical domain to the physical domain of dependence [38],

$$\Delta t_f = \min_a \left(\sqrt{h / \left| \frac{d\mathbf{v}_a}{dt} \right|} \right) ; \qquad \Delta t_{cv} = \min_a \frac{h}{c_s + \max_b \frac{\left| h \mathbf{v}_a \cdot \mathbf{r}_a \right|}{r_{ab}^2 + \eta^2}} ;$$

$$\Delta t = C_{CFL} \min \left(\Delta t_f, \Delta t_{cv} \right) \tag{35}$$

where $|d\mathbf{v}_a/dt|$ is the magnitude of particle acceleration. The variable time step is chosen as the minimum of Δt_f and Δt_{cv}, and is bounded by the Courant number C_{CFL}, usually in the range of 0.1 to 0.2.

2.6. Boundary conditions

Several boundary conditions are implemented in DualSPHysics. Solid boundary conditions are discretised by a set of boundary particles $a \in B$ that differ from the fluid particles F. Due to the Lagrangian nature of SPH, solid boundaries can also be moved straightforwardly according to user-defined motion. The two solid boundary conditions currently implemented in DualSPHysics are briefly described below.

Dynamic boundary conditions (DBC) are simply represented by fixed particles with density computed with the continuity equation and pressure obtained from the equation of state. Boundaries are easy to set up and computations are relatively stable and efficient, providing a robust option for complex geometries. Validations with dam-break flows and wave flumes have been published and real engineering applications have been simulated successfully [39, 40]. However, an unphysical gap between the fluid and these solid boundaries appear when fluid approaches to the boundary, decreasing the accuracy of pressures measured on the boundary. Therefore, a second approach is included now in DualSPHysics where the density of solid particles is obtained from ghost positions within the fluid domain by linear extrapolation, described in [41]. With this second approach, the gap is avoided and pressures of the boundary particles in still water converge to hydrostatic.

2.6.1. Dynamic boundary conditions

The dynamic boundary condition (DBC) discretises a solid wall with a set of boundary particles B for which the continuity equation is evaluated as [42]

$$\left. \frac{d\rho}{dt} \right|_{a \in B} = \rho_a \sum_{b \in F} \frac{m_b}{\rho_b} \mathbf{v}_{ab} \cdot \nabla W , \tag{36}$$

while the position of these particles is updated following the equation

$$\left. \frac{d\mathbf{v}}{dt} \right|_{\forall a \in B} = F_{(imposed)} , \tag{37}$$

where $F_{(imposed)}$ is the force imposed on moving boundary particles by the fluid, an external force or another solid object as described in Sections 3.2 and 4. A motion can be imposed by predefined user function or accelerations integrated in time. The increase in density that arises from Eq. (36) results in a pressure increase in the momentum equation of particle $a \in F$, yielding a repulsive force between the fluid and boundary particles. This simplified boundary condition is ideal for GPU implementation since

it enables code optimisation using vector lists. More details on this formulation can be found in Crespo et al. [42].

2.6.2. Modified dynamic boundary conditions

Starting from version 5.0 of DualSPHysics, an alternative way to discretize walls and other impermeable boundaries has been introduced to overcome some of the limitations of the DBC approach. This formulation has been named Modified-DBC (mDBC). Boundary particles used in mDBC are arranged in the same way as DBC particles. However, for each mDBC boundary particle, a ghost node is projected into the fluid across a boundary interface, similarly as the procedure employed by Marrone et al. [43]. Fluid properties are then computed at the ghost node through a corrected SPH approximation proposed by Liu and Liu [44] and finally mirrored back to the boundary particles. More details on this boundary condition can be found in English et al. [41].

The fluid density and its gradient are computed at the ghost node, and then the density of the boundary particle ρ_B is obtained by means of a linear approximation in the form:

$$\rho_B = \rho_G + (\mathbf{r}_B - \mathbf{r}_G) \cdot \left\langle \nabla \rho_G \right\rangle, \tag{38}$$

where $\boldsymbol{r}_B$ and $\boldsymbol{r}_G$ are the position of the boundary particle and associated ghost node, respectively, with $B \cap G$, and $\langle \nabla \rho_g \rangle$ is the corrected SPH gradient at the ghost node. The velocity at the ghost node is found using a Shepard filter sum

$$\mathbf{v}\Big|_{a \in G} = \frac{\sum_{b \in F} V_b \mathbf{v}_b W}{\sum_{b \in F} V_b W}, \tag{39}$$

The boundary particle then receives this velocity with the direction reversed to create a no-slip condition at the boundary interface. With this method, it is also possible to create a free-slip boundary by assigning the exact tangential velocity found at the ghost node to the boundary particles.

2.6.3. Periodic boundary conditions

Periodic boundary conditions (PBC), which can be used to describe an infinitely long domain by a finite cell domain, are implemented in DualSPHysics. This is achieved by allowing particles located within kh (with $k = 2$ for the smoothing kernels implemented in DualSPHysics) near an open lateral boundary to interact with the fluid particles of the opposite lateral boundary completing the truncated support in a cyclic manner in 2-D and 3-D.

2.7.   Particle shifting algorithm

The anisotropic distribution of particles introduces additional discretisation error through the zeroth and higher order kernel moments (i.e. the discrete version of Eq. (3)). This is especially true in negligible or large dynamics [33] and violent flows where particles may not maintain an isotropic distribution. In DualSPHysics, the Fickian based particle shifting algorithm of Lind et al. [45] is used to maintain a near isotropic particle distribution,

$$\delta \mathbf{r}_a \Big|_{a \in F} = -D_s \nabla C_a, \tag{40}$$

where $\delta \mathbf{r}_a$ is the shifting distance, ∇C_a is the kernel gradient and D_s

$$D_s = A_s h \left\| \mathbf{v}_a \right\| \Delta t, \tag{41}$$

where A_s is a parameter and $\|\mathbf{v}_a\|$ is the velocity magnitude of particle a [46]. This imposes a restriction based on the particle velocity magnitude to prevent excessive movement and loss of information from particles moving to different domain cells. The parameter value A_s is in the range $[1-6]$, with a value of 2 recommended by Skillen et al. [46]. Expressions for ∇C_a are given in Section 3.4.

3. Functionalities

3.1. Wave generation and absorption in DualSPHysics

DualSPHysics employs moving boundary particles to generate regular and random long-crested waves. Moving boundaries mimic wavemakers typical of experimental facilities, namely piston-type and flap with variable draft. Moreover, a piston wavemaker allows solitary wave generation. Active and passive wave absorption techniques are implemented to allow the generation of long time wave series in relatively short domains with negligible reflection. Wave absorption techniques can be categorized as passive and active, respectively. Passive wave absorbers are usually required to damp the wave energy and to reduce the reflection exerted by the boundary of the model domain. Further details on passive absorption in DualSPHysics can be found in Altomare et al. [47]. In active absorption systems, the wavemaker real-time displacement is corrected to cancel out the reflected waves and to damp the re-reflection phenomenon altogether. For a complete description of the implementation, the reader can refer to Altomare et al. [47] and Domínguez et al. [48].

3.1.1. Generation of regular waves

The Biesel transfer functions [49] relate the wavemaker displacement to the water surface elevation. Based on Hughes [50], the general solution for the second-order wavemaker motion to suppress second harmonics can be expressed as follows:

$$X_0(t) = \frac{H}{2m_{cr}}\sin\left(\frac{2\pi t}{T}+\phi\right)+\left[\left(\frac{H^2}{32\left(1-\dfrac{d}{2(d+d_0)}\right)}\right)\cdot\left(\frac{3\cosh\left(\dfrac{2\pi d}{L}\right)}{\sinh^3\left(\dfrac{2\pi d}{L}\right)}-\frac{2}{m_{cr}}\right)\right]\sin\left(\frac{4\pi t}{T}+2\phi\right), \quad (42)$$

where m_{cr} is given by

$$m_{cr} = \frac{4\sinh\left(\dfrac{2\pi d}{L}\right)}{\sinh\left(\dfrac{4\pi d}{L}\right)+\dfrac{2\pi d}{L}}\left[\sinh\left(\frac{2\pi d}{L}\right)+\frac{1-\cosh\left(\dfrac{2\pi d}{L}\right)}{\dfrac{2\pi}{L}(d+d_0)}\right], \quad (43)$$

where H is the wave height, d is the initial water depth, ϕ $[0, 2\pi]$ is the initial phase and d_0 is the location of the wavemaker hinge (for flap: $d_0 < 0$ if hinge is above the sea bottom, $d_0 > 0$ if below; for piston $d_0 \rightarrow \infty$). The quantity $2\pi/T$ is the angular frequency and $2\pi/L$ corresponds to the wave number. T is the wave period and L the wave length. Eqs. (42) and (43) are valid for regular waves and for flap- and piston-type wavemakers. $X_0(t)$ is the wavemaker displacement at the free surface: for a piston, this corresponds to the horizontal movement of the entire board, $e(t)$; for a flap, the amplitude of the motion is expressed by the angle swept in time around the hinge, and calculated as $\Theta(t) = \tan^{-1}(X_0(t) / (d + d_0))$. The application of Eq. (42) is limited to values of the Ursell number $HL^2/d^3 < 8\pi^2/3$.

3.1.2. Generation of random waves

Irregular wave generation is performed in DualSPHysics based on Liu and Frigaard [51]. Starting from an assigned wave spectrum, the transfer function is applied to each component in which the spectrum is discretised. Composing all the *i*-th components of the wavemaker displacement will provide the first-order solution for irregular waves. Two standard wave spectra are implemented in DualSPHysics: JONSWAP and Pierson-Moskowitz spectra. The characteristic parameters of each spectrum can be assigned by the user together with the value of N_s (number of parts into which the spectrum is divided). The spectral frequency discretization can be done in four different ways: equidistant discretization, unevenly distributed, and by applying a stretched or cosine stretched function. The use of the last two methods has shown the most accurate results in terms of wave height distribution and groupness, even for low values of *N*.

Second-order wave generation is implemented in order to cancel out the parasitic long waves [52] and the spurious displacement long waves, the latter ones caused by finite wavemaker displacements away from the mean position. The method implemented in DualSPHysics is based on the solution for the control signal of the wavemaker that is described in Barthel et al. [53] and is fully described in Altomare et al. [47].

3.1.3. Solitary waves

The displacement of a piston-type wavemaker to generate a solitary wave is prescribed by the following equation [54]:

$$x_s(t) = \frac{2H}{Kd} \tanh \left[K \left(ct - x_s(t) \right) \right], \tag{44}$$

in which the parameter *K* is the outskirt coefficient describing the way the free surface elevation tends towards the mean level at infinity. Three different theories for solitary wave generation have been implemented in DualSPHysics. The three different approaches result in different wave propagation properties, i.e. different loss of amplitude and differences in generated trailing waves [48]. One solution is derived from the Rayleigh solitary wave solution; the second one uses a wavemaker law of motion derived from the KdV solution; the third one consists of solving Eq. (44) iteratively using the Newton-Raphson method.

Additionally, DualSPHysics allows the generation of a series of multiple solitary waves where the motion of the wavemaker is calculated for each *i*-th solitary wave. Any of the three aforementioned generation theories can be used. Then, the time lag between the (*i*+1)-th and *i*-th solitary waves has to be specified as a fraction or multiple of the generation time used for the *i*-th solitary wave. The generation of multiple solitary waves is a simple but efficient way to model a wave train of several tsunamis that might have been triggered by the same tectonic event. For further details on solitary wave generation in DualSPHysics, please see Domínguez et al. [48].

3.1.4. Active wave absorption

The implemented active wave absorption system is based on the time-domain filtering technique that uses the free-surface elevation at the wavemaker position as feedback for the control of the wavemaker displacement. The wavemaker velocity is corrected to match the velocity induced by the wave that will be absorbed. For a piston-type wavemaker, the wave absorption is performed using the linear long wave theory [55]. The corrected velocity, U_c, is expressed as follows:

$$U_C(t) = U_i(t) - \left(\eta_{WG,SPH}(t) - \eta_{AWAS,i}(t) \right) \sqrt{g/d} , \tag{45}$$

where $\eta_{AWAS,i}$ is the target incident free-surface elevation and $\eta_{WG,PH}$ is the measured one in front of the wavemaker, U_i is the theoretical wave maker velocity, equal to the derivative in time of the piston-type

step using the following equation:

$$e(t + \Delta t) = e(t) + \left(U_C(t + \Delta t) + U_C(t) \right) \frac{\Delta t}{2}. \tag{46}$$

3.2. Fluid-driven objects

One of the useful capabilities of SPH models is the simulation of fluid-driven objects. The implementation of this functionality in DualSPHysics derives the motion of a floating structure by considering its interaction with fluid particles and using the resulting forces to drive its motion.

First, the net force on each individual particle that form the same floating object is computed as a summation of the contributions of all surrounding fluid particles. In this way, each floating particle $a \, \epsilon \, K$ experiences an acceleration ($d\mathbf{v}/dt$) given by

$$\left. \frac{d\mathbf{v}}{dt} \right|_{a \in K} = \sum_{b \in F} m_b \frac{d\mathbf{v}_{ab}}{dt}. \tag{47}$$

As the object is assumed rigid, the basic equations of rigid body dynamics are solved to obtain its motion:

$$M \frac{d\mathbf{v}}{dt} = \sum_{b \in K} m_b \frac{d\mathbf{v}_b}{dt}, \tag{48}$$

$$I \frac{d\mathbf{\Omega}}{dt} = \sum_{b \in K} m_b \left(\mathbf{r}_b - \mathbf{R}_0 \right) \frac{d\mathbf{v}_b}{dt}, \tag{49}$$

where M is the total mass of the object, I the moment of inertia, $\mathbf{v}$ the velocity, $\mathbf{\Omega}$ the angular velocity and $\mathbf{R}_0$ the centre of mass. Eqs (48) and (49) are integrated in time to obtain the values of $\mathbf{v}$ and $\mathbf{\Omega}$ at the beginning of the subsequent time step. Each particle belonging to the object will move with a velocity $\mathbf{v}_k$ given by

$$\mathbf{v}_k = \mathbf{v} + \mathbf{\Omega} \times \left(\mathbf{r}_k - \mathbf{R}_0 \right). \tag{50}$$

This technique has been shown to preserve both linear and angular momenta [56].

3.3. Open boundaries (inlet & outlet)

Several types of open boundary approaches are available in SPH, among which the unified semi-analytical model [57–59], the mirror particles method [60], and the buffer region strategy, originally introduced by Lastiwka et al. [61], then used in other works [62–65]. The latter approach has been chosen for implementation in DualSPHysics, mainly for its simplicity but also for its suitability to a variety of engineering applications.

The rationale behind using buffer regions is to attach several layers of fictitious SPH particles to a region of the computational domain where an open boundary is needed, for example an inlet or an outlet. Buffer particles are usually deployed to follow the geometry of the open boundary. Figure 3 depicts a buffer region near a permeable boundary, where an inflow (or outflow) of SPH particles is desired. Buffer particles are extruded starting from the permeable boundary, i.e. the fluid-buffer interface, with a finite number of layers chosen to equal or exceed the kernel radius, ensuring full kernel support for the closest fluid particles to the inlet/outlet threshold. Once the buffer particles are created, it is necessary to enforce boundary conditions appropriately. This can be done by assigning physical quantities to the buffer

particles in two different ways: the first is to simply pre-assign the quantities to these particles, for example a certain velocity or pressure field defined by the SPH user. The second option is to extrapolate the physical quantities from the fluid domain, using ghost nodes properly placed within the fluid particles neighbouring the open boundary. The placement of ghost nodes is shown in Figure 3 for three different buffer particles, although it is noted that every buffer particle must be assigned a corresponding ghost node in the fluid. Ghost nodes are created by mirroring the position of the buffer particles into the fluid domain, following the normal direction to the permeable boundary.
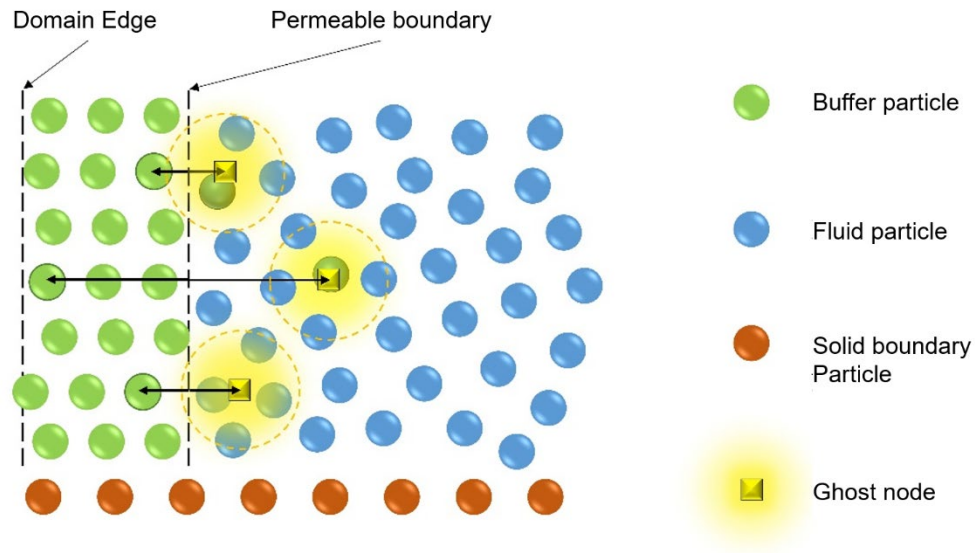

Figure 3. A sketch of the placement of ghost nodes for the open boundary algorithm implemented in DualSPHysics.

When extrapolating fluid properties from the ghost nodes, the idea for retrieving quantities of the buffer particles is to perform an SPH interpolation at the ghost nodes, and then transport that information back to the buffer particles. However, a standard SPH particle interpolation would not be consistent because these nodes are situated near an open boundary, and thus not fully surrounded by fluid particles. To obviate this loss of consistency, the algorithm proposed in Liu and Liu [44] is implemented, restoring first order kernel and particle consistency. More details about the fluid extrapolation via ghost nodes in DualSPHysics, including the linear system solved at each time step, can be found in Tafuni et al. [66].

The implementation of open boundaries in DualSPHysics is done in a versatile way, with no distinction between an inlet and an outlet region: a buffer area can behave simultaneously as an inlet and outlet, enabling the use of DualSPHysics to problems with backflow and flow reversion, see for example [67]. Other attractive features for use in real engineering problems include: (a) the capability of enforcing unsteady velocity and pressure profiles and/or pressure and velocity gradients along a given direction; (b) the simulation of variable free-surface elevation, an essential aspect in flow with a free surface because of the presence of waves entering and exiting the computational domain; (c) the ability to generate several buffer geometries, including square, circular and other common engineering shapes.

3.4. Multi-phase: Liquid and gas

In its original release, the DualSPHysics code could only deal with single-phase flows. This limits the range of applications, as a diverse range of problems such as liquid-gas flows with mixing and violent free-surface hydrodynamic interaction cannot be accurately modelled. To address these and other problems typical of industrial and research fields, such as coastal and nuclear engineering, DualSPHysics includes the multi-phase model from Mokos et al. [68].

The multi-phase model uses a modified version of Tait's equation of state [34] (the same equation used in single-phase DualSPHysics) for both liquids and gases, with an additional term representing the cohesion forces between particles of the same phase as proposed by Nugent and Posch [69].

$$P = \frac{c_s^2 \rho_0}{\gamma}\left[\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right] + X - \alpha\rho^2, \text{with } \alpha = 1.5g\left(\frac{\rho_l}{\rho_g^2}\right)L_c \,, \tag{51}$$

where X represents a constant background pressure. The last term is only used for the gas phase and the cohesion coefficient α depends on the properties of the different phases (ρ_l and ρ_g are the initial densities of the liquid and the gas) and a characteristic length scale L_c of the problem [70].

For the continuity and momentum equations, the multi-phase model uses the variationally consistent forms of the velocity divergence and pressure gradient as proposed by Colagrossi and Landrini [71]. An additional term, originating from the cohesion term in the equation of state, is included in the momentum equation for gases [69].

$$\left.\frac{d\mathbf{v}}{dt}\right|_{a\in F} = -\sum_{b\in P} m_b\left(\frac{P_a + P_b}{\rho_a\rho_b} + \Pi_{ab}\right)\nabla W_{ab} + 2\alpha\rho_a^2\sum_{b\in P}\frac{m_b}{\rho_a\rho_b}\nabla W_{ab} + \mathbf{g} \,. \tag{52}$$

The use of high resolution, made possible by GPUs, allowed the identification of a numerical inconsistency for phases with a high density ratio, particularly in violent flows. Unphysical voids and phase separation occur, ultimately leading to numerical instability. The Fickian-based particle shifting algorithm of Section 2.7, with a selectively activated free-surface correction, has been adapted for multi-phase to prevent the creation of unnatural voids and maintain numerical stability through nearly uniform distributions [68]. The Fickian shifting algorithm is a modified version of the one developed by Lind et al. [45], adapted for multi-phase flows. The algorithm uses Fickian diffusion to shift particles towards areas with lower concentration, while limiting the movement on the normal direction to the free surface. The particle concentration gradient is computed through an SPH gradient approximation with the particle concentration C computed through the sum of the smoothing kernel.

$$\nabla C|_{a\in F} = \sum_{b\in P}\frac{m_b}{\rho_b}(C_b - C_a)\nabla W_{ab} \text{ with } C_a = \sum_{b\in P}\frac{m_b}{\rho_b}W_{ab} \,. \tag{53}$$

The concentration gradient allows the computation of the particle shifting distance, δr_s, with the Fickian diffusion coefficient computed through the method of Skillen et al. [46].

To prevent the unphysical movement at the free surface Lind et al. [45] proposed that the concentration gradient near the surface was controlled using the local tangent (**s**) and normal (**n**) vectors at the free surface. The free surface equation is expanded to 3-D so that the bitangent vector (**b**, orthogonal to **s** and **n**) is also considered [68].

$$\delta\mathbf{r}_s = -D_s\left(\frac{\partial C_a}{\partial s}\mathbf{s} + \frac{\partial C_a}{\partial b}\mathbf{b} + a_n\left(\frac{\partial C_a}{\partial n} - \beta_n\right)\mathbf{n}\right), \tag{54}$$

where β_n is the initial concentration gradient and α_n is a numerical parameter limiting diffusion in the normal direction. This is set to 0 for violent flows, and 0.1 for long slow flows, such as standing gravity waves [45]. For the multi-phase flow, the free surface term is applied to the liquid-gas interface, but only on the liquid particles. Shifting for the gas phase should not be restricted in the interface, to allow for free expansion to areas of low concentration, similar to the behaviour of an actual gas [68].

For flows in which the Weber or the Eötvös number is sufficiently small, surface tension forces need to be taken into account. In order to model these flows, DualSPHysics includes a surface tension formulation. The method uses the Continuum Surface Force (CSF) model [72] and the SPH implementation by Hu and Adams [73]. This implementation uses a tensor formulation to avoid

computing the surface curvature as proposed by Lafaurie et al. [74]. A colour function is used to distinguish between the different phases and compute the surface normal [75].

3.5. Multi-phase: Non-Newtonian flows

The rheology of materials such as polymers, slurries, pastes and suspensions is a frequent occurrence in industrial applications and academic research. These materials are usually characterized by a yield stress below which the material has not yielded and does not flow. Furthermore, the shear strain to shear stress relationship may not be linear as with Newtonian fluids. In general, fluids that do not obey the Newton law of viscosity may be classified as non-Newtonian fluids with viscoelastic, time dependant viscosity and non-Newtonian viscosity.

In DualSPHysics the generalized Herschel-Bulkley-Papanastasiou [76, 77] (HBP) model has been implemented following the work of Fourtakas and Rogers [78] on liquid-sediment two-phase flows. The HBP approach allows the modelling of a wide range of viscoplastic materials, such as Bingham plastic, Bingham pseudoplastic and dilatant fluids. Furthermore, the HBP generalised non-Newtonian model can be used to simulate shear thinning or thickening materials in the absence of yield strength. The constitutive equation for the HBP model is written as,

$$\tau^{ij} = \eta_{app}\dot{\gamma}^{ij}, \tag{55}$$

where τ^{ij} is the shear stress tensor and ij denote coordinate directions using the Einstein notation. By dropping the Einstein notation for brevity, the symmetric shear rate tensor $\dot{\gamma}$ given by

$$\dot{\gamma} = \left[\nabla\mathbf{v} + \nabla\mathbf{v}^T\right], \tag{56}$$

with $\nabla\mathbf{v}$ and $\nabla\mathbf{v}^T$ representing the velocity gradient and its transpose, respectively, while η_{app} is the apparent viscosity

$$\eta_{app} = \mu\left|\dot{\gamma}\right|^{n_{HB}-1} + \frac{\tau_y}{\left|\dot{\gamma}\right|}\left[1 - e^{-m_P|\dot{\gamma}|}\right]. \tag{57}$$

In the above, τ_y is the yield stress of the fluid, μ is the viscosity (or consistency index), n_{HB} is the power law index (commonly referred to as the Herschel-Bulkley parameter), m_p is the exponential shear rate growth parameter (or Papanastasiou parameter), and $|\dot{\gamma}|$ is the magnitude of the symmetric strain rate tensor defined as

$$\left|\dot{\gamma}\right| = \sqrt{\frac{1}{2}\left((tr(\dot{\gamma}))^2 - tr(\dot{\gamma}^2)\right)}. \tag{58}$$

The Papanastasiou parameter controls the growth of the stress with a finite value for small shear rates in the unyielded region, and the linear behaviour in the yielded region. Note that, as $m_p \to \infty$, the HBP model reduces to the original Herschel-Bulkley model, and with $n_{HB} =1$ it further reduces to a simple Bingham model. When $n_{HB} = 1$ and $m_p = 0$, the model reduces to a Newtonian fluid. Moreover, the HBP model does not exhibit a discontinuity at zero shear rates, in contrast with a pure Bingham model, which is usually discretized using a bi-viscosity model.

The implementation of the HBP model in DualSPHysics is outlined next. The momentum Eq. (13) is written as

$$\left.\frac{d\mathbf{v}}{dt}\right|_{a\in F} = -\sum_{b\in P} m_b\left(\frac{P_b + P_a}{\rho_b\rho_a}\right)\nabla_a W_{ab} + \left\langle\frac{1}{\rho}\nabla\cdot\tau\right\rangle + \boldsymbol{g}, \tag{59}$$

where the brackets denote an SPH approximation. Two formulations are available in the multi-phase non-Newtonian formulation for the discretization of the viscous forces. The first uses the formulation of Morris et al. [79] that reads

$$\left.\frac{1}{\rho}\nabla\cdot\tau\right|_{a\in\mathrm{F}} \approx \upsilon_{app}\nabla^2\mathbf{v}_a = -\sum_{b\in\mathrm{P}}\frac{m_b(\upsilon_a+\upsilon_b)\mathbf{r}_{ab}\cdot\nabla_a W_{ab}}{\rho_b(\left|r_{ab}\right|^2+0.001h^2)}\mathbf{v}_{ab},$$ (60)

where υ_{app} is the apparent kinematic viscosity. The second formulation takes advantage of the conservative gradient operator in SPH form and the viscous forces are discretised according to Eq. (26).

The continuity Eq. (12) has the additional density ratio terms for the interpolating and neighbouring particles, allowing different densities to be used in each phase of the system. It is thus written as

$$\left.\frac{d\rho}{dt}\right|_{a\in\mathrm{F}} = \rho_a\sum_{b\in\mathrm{P}}\frac{m_b}{\rho_b}\mathbf{v}_{ab}\cdot\nabla_a W_{ab}+D_{a\in\mathrm{M}},$$ (61)

with $\mathrm{M} =\sqcup_{n\in\mathbb{N}}\mathrm{M}_n$, and n is the number of phases. For the discretisation of the shear stress tensor and velocity gradients of Eq. (56), a finite-difference approach or an SPH gradient summation can be used. The former reads

$$\nabla\mathbf{v}_{ab} = \frac{\mathbf{v}_{ab}\cdot\mathbf{r}_{ab}}{\left|r_{ab}\right|^2},$$ (62)

whereas the latter takes the following form

$$\left.\nabla\mathbf{v}\right|_{a\in\mathrm{P}} = \sum_{b\in\mathrm{P}}\frac{m_b}{\rho_b}(\mathbf{v}_a-\mathbf{v}_b)\nabla_a W_{ab}.$$ (63)

In this implementation, any combination of viscous operators and gradients is allowed in the multi-phase non-Newtonian DualSPHysics formulation. More information about the multi-phase non-Newtonian formulation can be found in Fourtakas and Rogers [78]. Future implementations are planned, and include a sediment phase with yielded and unyielded sediment regions, as described in the aforementioned manuscript.

4. Coupling with other models

Following the guidelines in the "SPHERIC Grand Challenge #4: Coupling to other models", it has been concluded that, depending on the problem under scope, it can be much more efficient to couple SPH with a different numerical method rather than sticking to a single method of choice. The main reason for coupling is to enhance the capabilities of both models beyond their specific application fields, thus enabling the study of a wider range of problems. This section describes the successful coupling strategies of DualSPHysics with other models. For brevity, the governing equations of the coupled models are not included below, but details and reference literature are provided.

4.1. Coupling with wave propagation models

Multi-scale complex simulations, often characterized by three-dimensional domains and long durations, require extensive computational resources if only one stand-alone model is employed. In the special case of coastal engineering applications, the proper simulation of all processes from wave generation to wave-structure/beach interaction is required. SPH-based solvers allow a detailed model of local phenomena, such as interaction between sea waves and coastal defences [39, 80]. The mesh-free nature

of SPH is successful at mimicking the wave generation system of experimental facilities, employing moving boundaries such as piston- or flap-type wavemakers. However, when using these schemes, waves need to be generated and propagate over relatively long domains to simulate wave transformation properly and thus guarantee the correct wave sea state at the toe of the structure, similarly to what happen in physical model tests. To reduce the computational cost of such simulations, less computationally demanding wave propagation models can be coupled with SPH-based solvers.

a) Multi-layer piston wavemaker

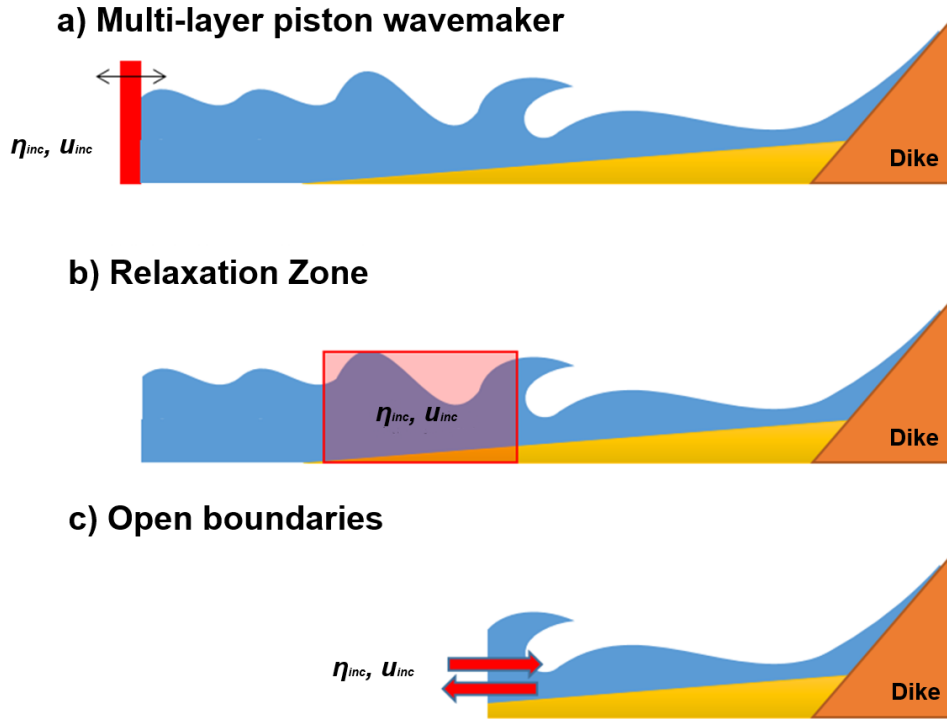### b) Relaxation Zone

c) Open boundaries

Figure 4. Representation of the coupling techniques between DualSPHysics and wave propagation models: u_{inc} and η_{inc} are the velocity and water surface elevation from the wave propagation model used as steering signal in DualSPHysics.

The three coupling schemes between the DualSPHysics and SWASH models shown in Figure 4 are hereby listed and briefly described:

- Multi-layered piston: moving boundaries are employed to generate waves. This scheme has been coupled with the SWASH model [81] by Altomare et al. [82]. The real time displacement of the multi-layered piston wavemaker is reconstructed on the basis of the velocity time series calculated by the SWASH model and interpolated along the water depth at a specific x-coordinate (here the x-axis corresponds to the direction of wave propagation). There is no reflection compensation, therefore the use of this technique is recommended for low-reflective cases or very short time series. It is a one-way offline coupling and at least 8 layers are recommended to be used in SWASH. The same strategy was applied to develop a two-way coupling with a fully nonlinear potential flow solver OceanWave3D [83] by Verbrugghe et al. [84]. In this case, both models (DualSPHysics and OceanWave3D) exchange sea-state information throughout the simulation.

- Relaxation zone [85]: the coupling zone is extended from one single location to an area. In this one-way offline coupling, the orbital velocity calculated by the wave propagation model is imposed to the fluid particles in DualSPHysics. The particle velocity is controlled by an analytical solution with a weighting function C to balance the reflected waves inside the relaxation zone. A correction for the velocity of fluid particles is implemented to avoid unrealistic increase of the water level outside the generation zone, caused by Stokes drift velocity.

- Open boundaries: an improved two-way coupling with OceanWave3D is presented in Verbrugghe et al. [86]. The open boundary algorithm in Section 3.3 is employed at the coupling interfaces. At the inlet, horizontal orbital velocities and surface elevations are imposed on the buffer particles. At the outlet, horizontal orbital velocities are imposed, while surface elevation is extrapolated from the SPH fluid domain. A velocity correction algorithm based on linear shallow water theory is applied for absorbing wave reflection in the SPH fluid domain, similar to the one described in Altomare et al. [47]. The SPH surface elevation is coupled back to OceanWave3D to replace the original signal.

All coupling methodologies described above used 2-D tests for validation, proving a model accuracy that is generally in the order of the smoothing length. Inherently, open boundaries allow coupling in shallow waters, where wave non-linearity and mass transport are dominant (e.g. in wave breaking zone).

4.2. Coupling with Discrete Element Method (DEM)

Simple (i.e. granular-type) simultaneous solid–fluid and solid–solid interactions represent a set of problems common to several engineering disciplines such as coastal, offshore, maritime and fluvial engineering. In order to characterize such flows, DualSPHysics includes a Distributed Contact Discrete Element Method (DCDEM) [87] within its SPH formulations for fluid-solid modelling. The general concept is to compute the forces acting on a fluid-solid particle pair using the SPH formulation and the solid-solid interactions via DEM, retaining the same explicit integrator and also the DualSPHysics meshless framework. The DCDEM concept introduced by Cummins and Cleary [88] is used, allowing arbitrary solid shapes to be reproduced by a collection of particles that are then subjected to a rigid body restriction. Given adequate non-linear contact models [89], a wide range of material behaviours can be modelled using three basic parameters: Poisson ratio, Young's modulus and dynamic friction coefficient. The drawbacks of the model are common to all general DEM formulations: the explicit nature may impose very small stability regions, extended frictional contacts are either computationally expensive or inaccurately reproduced and low resolution simulations may lead to artificial geometry effects due to the underlying spherical shape of the particles.

Accounting for these limitations, given its simplicity and speed, applications include assessment of the severity of hydrodynamic actions on structures, risk of natural hazards, design of floating bodies or design of exposed structures.

4.3. Coupling with Project Chrono

Given the shortcomings of the DCDEM approach for general simulations, the Differential Variational Inequality (DVI) implementation of Project Chrono is leveraged [90]. The implementation resorts to linearising and efficiently solving a set of Cone Complementary Problems describing the restricted dynamics of a multi-body system. The notion of restriction is loose, so besides non-penetration contacts, mechanical-type restrictions can be imposed, such as joints, hinges, sliders, springs and combinations of these. Given the underlying mesh representing the bodies (that DualSPHysics generates and maps to an underlying SPH particle distribution), frictional contacts are fully described, including sliding and rolling friction. Thanks to the efficiency of the DVI formulation, the impact on the computational time is residual, allowing the simulation of large and complex multiphysics systems. In addition, the DVI formulation presents several advantages comparing with explicit DEM models since it has a much larger stability region leading to larger time-steps.

The work of Canelas et al. [91] included details of the implementation and validation with fluid–structure–structure interaction cases. Moreover, several hypothetical cases were also presented to demonstrate the capabilities of DualSPHysics coupled with Project Chrono.

Figure 5 shows the flow chart of the DualSPHysics coupling with Project Chrono. It can be seen how DualSPHysics computes the linear and angular acceleration of floating objects starting from interaction with fluid particles. These values and the *dt* are passed to Project Chrono, who calculates final position

and velocities according to the constraints defined for Chrono model. Finally, DualSPHysics updates the data of the floating particles according to the information received from Project Chrono.
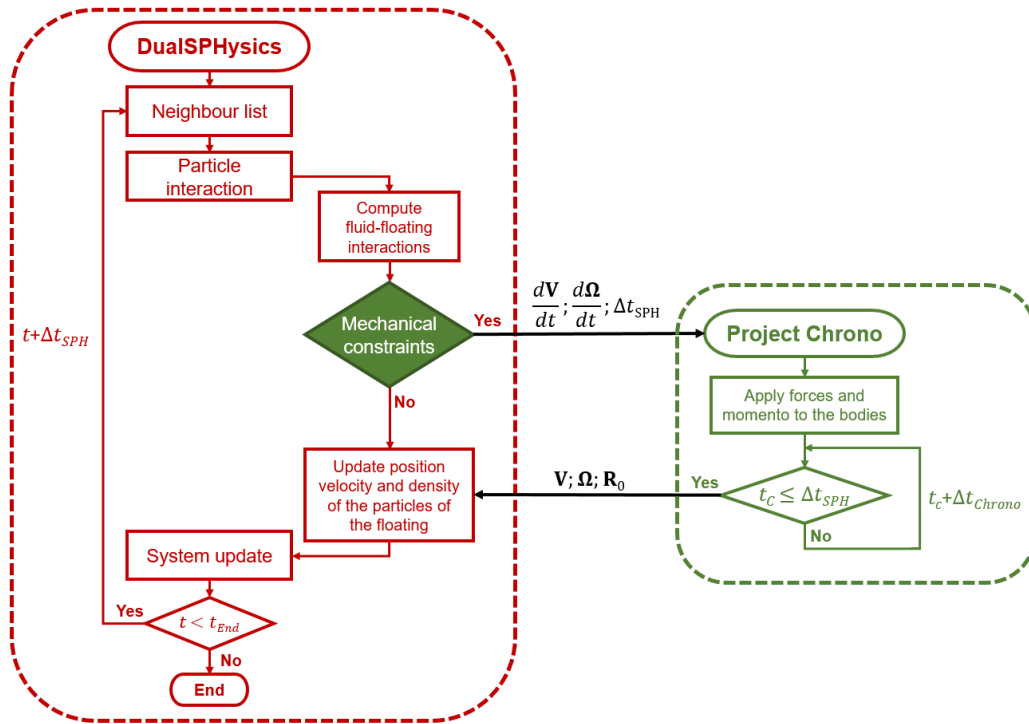

Figure 5. Flow chart of the coupling between DualSPHysics and Project Chrono.

4.4. Coupling with MoorDyn

In Section 3.2, the capability of DualSPHysics to simulate fluid-driven objects has been described. The code was shown to be accurate in simulating the interaction of sea waves with floating objects. In real-life problems, floating structures are usually moored to the seabed. Hence, mooring tensions in the floating objects should be included in the SPH solver. The solution is to couple DualSPHysics with a library designed for coupling with other numerical models that already solves mooring dynamics. Therefore, DualSPHysics was coupled with MoorDyn (http://www.matt-hall.ca/moordyn.html), an open-source dynamic mooring line model [92]. MoorDyn discretizes mooring lines as point masses connected by linear spring-damper segments to provide elasticity in the axial direction. It is accurate and efficient, and it includes the contact forces with the seabed.

A complete description of the coupling procedure can be found in Domínguez et al. [93], where the coupling between DualSPHysics and MoorDyn was applied to simulate the motion of a moored floating structure under the action of waves.

Figure 6 shows the flow chart of this coupling approach. It consists in passing the fairlead kinematics to MoorDyn, which then solves the mooring system dynamics. The resulting fairlead tensions are transferred back to DualSPHysics. In this way, MoorDyn calculates the tensions of the mooring lines, which are added as external forces to derive the final resulting motion of the moored floating in the SPH solver.
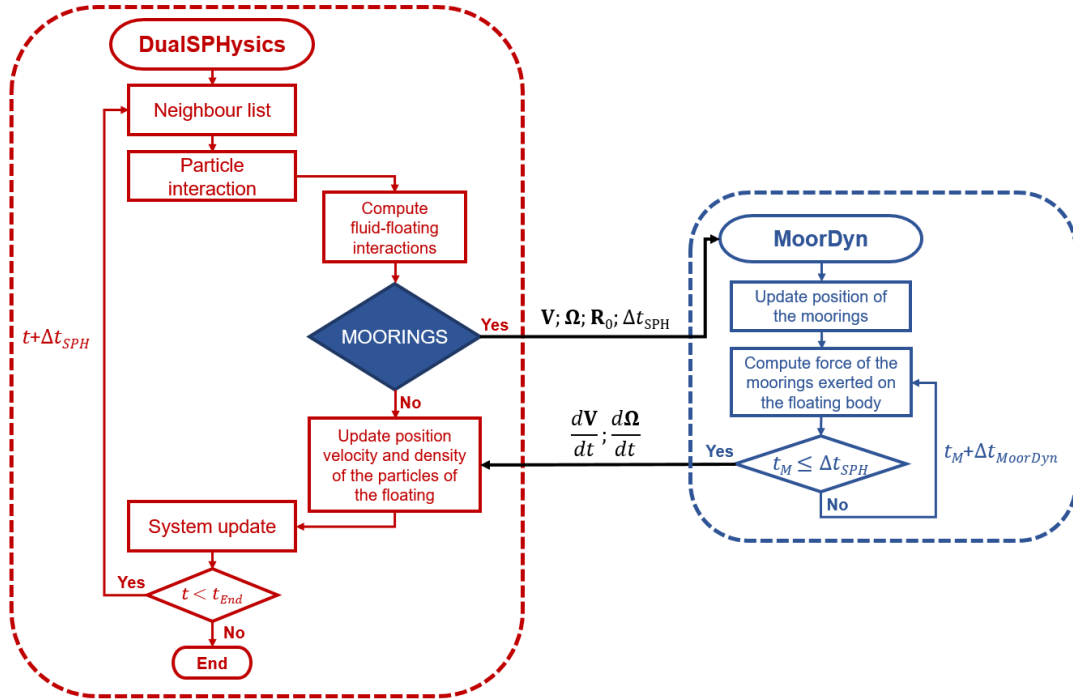
Figure 6. Flow chart of the coupling between DualSPHysics and MoorDyn.

5. Code and implementation

DualSPHysics is developed to run on two different hardware architectures: the traditional Central Processing Unit (CPU) and the more recent Graphics Processing Unit (GPU). Therefore, all algorithms and features included in this SPH solver are implemented and optimised for both architectures. The programing language C++ is used to code the SPH formulation presented in Section 3 for CPU execution, but also implements the main program logic and all tasks not related to SPH calculations, such as the initial configuration and input/output data. The GPU execution is performed using the NVIDIA's programming framework Compute Unified Device Architecture (CUDA). Thus, the SPH formulation is also implemented by using the CUDA programming language in kernels or functions executed on GPU rather than CPU.

This dual capability of the DualSPHysics execution provides significant advantages but also some drawbacks. Among the advantages, DualSPHysics can perform large simulations by using the computational power of the latest GPUs by NVIDIA, but can also run simulations on workstations without GPUs by using the current multi-core CPU version of the code. This solver enables fair comparisons of the SPH performance between CPU and GPU, since the same formulation is implemented in an optimised way for both hardware architectures. With the advent of more mature compilers and development environments, the CPU code has become more accessible than the GPU one, especially when modifying and debugging, so it represents an attractive option for users approaching DualSPHysics for the first time. This is paramount since DualSPHysics was developed as an open-source framework for the SPH method and researchers use it to develop and test new formulations. The main drawback is the higher maintenance cost of the solver and optimisation of new functionalities, since two different implementations must be maintained and different optimisation strategies are needed for each architecture.

5.1. Implementation

The SPH implementation in DualSPHysics is centred on three main consecutive tasks: (1) Neighbour List (NL), (2) Particle Interaction (PI), and (3) System Update (SU). These allow the simulation of an interval of physical time (i.e., the simulation step) and their execution is repeated until the total physical time to be simulated is complete, as also shown in Figure 7:

(1) The Neighbour List (NL) prepares the data needed to find the neighbouring particles of each SPH particle efficiently in the next task. Different NL approaches have been considered in the literature [13, 94, 95]. DualSPHysics implements a cell-linked list approach, since this appears to be the most efficient for the simulation of large numbers of particles, as shown in Domínguez et al. [13] for CPU simulations and later in Winkler et al. [95] for GPU simulations. During the NL task, the simulation domain is split up in cubic cells of size $2h$, i.e. the maximum interaction distance between particles. The particles with their data are sorted according to the cell to which they belong and an index array with the first and last particle of each cell is created. The runtime of this task usually represents a small percentage of the total execution time. However, the result of the Neighbour List is critical to maximize the performance on the subsequent task.

(2) The Particle Interaction (PI) is the most time-consuming task. It represents over 90% of the total execution time. During the PI process, the momentum and continuity equations are solved by the interaction of each particle with its neighbouring particles. The neighbour search is performed efficiently since, for each particle, only the neighbouring particles in the adjacent cells will contribute to the particle interaction.

(3) The last task is the System Update (SU), during which the duration of the simulation step is calculated, together with the new values of the properties of SPH particles (position, density, velocity). These properties are computed for the next simulation step according to the values of these properties at the current time step, the results of particle interactions, and the duration of the simulation step.

Other less important but necessary tasks are the initial configuration and data loading, which are executed once at the beginning of the calculation and also during the periodic data output that is usually performed every hundreds or thousands of simulation steps. More details of the work carried out during each of the above tasks can be found in Crespo et al. [96].
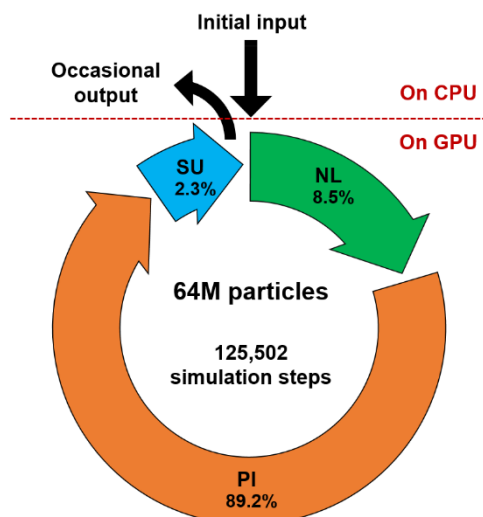

Figure 7. Scheme of the main SPH tasks (NL, PI and SU) during a DualSPHysics execution. The time percentage corresponding to each task is shown during the simulation of a 3-D dam break test case, with 64M particles on a Tesla V100 GPU.

Programming paradigms: DualSPHysics is written in C++ following the programming paradigm of Object-oriented programming (OOP), in which the particle data laid out in memory as Structure-of-Arrays (SoA). However, several properties are grouped in 16-bytes structures to maximise the performance in parallel executions as described in [14]. The CPU and GPU implementations are available under two different classes (JSphCpu and JSphGpu), both of which inherit from the same class (JSph) with common parts. In addition, the JSphGpu class calls the kernels written in CUDA language. The CPU and GPU codes are very similar to ease the expansion and maintenance of DualSPHysics. However, some differences are inevitable due to the characteristics of each architecture and the different optimisation strategies employed to maximize the performance in each hardware architecture [14]. The CPU implementation uses the application programming interface Open Multi-Processing (OpenMP) to perform parallel execution of SPH method on one or several multi-core CPUs hosted on the same workstation or computation node. Each CPU thread executes in parallel the interactions with the neighbours of a given set of particles, while a CUDA thread is created to compute the interactions of each particle on GPU.

Memory access: The CPUs access the main memory of the host directly through different levels of cache memory that minimise the latency of this access. However, the efficient use of the GPU memory's hierarchy is the responsibility of the developer and it is critical to performance. The GPU computation is executed in the device's (GPU) memory and the communication between the main memory and the GPU memory (typically via PCI Express bus) represents the main bottleneck in GPU computing. The GPU implementation of DualSPHysics copies all particle data to the GPU memory at the beginning of the simulation and all calculations are performed on GPU, so that data transfers between the main memory and the device memory are minimized. The particles' data on the GPU memory is only transferred to the CPU memory when this data must be stored in long-term memory, which happens occasionally during the simulation, and thus with negligible impact on performance.

Precision: Most of the real variables in DualSPHysics are coded in double precision, but some particle data uses single precision since double precision is generally not necessary and the use of single precision improves the performance, especially during GPU execution. However, the use of double precision is mandatory on position variables for high-resolution simulations with large domains [97]. The precision of particles' position is critical in two phases: the calculation of interaction distance between particles in PI task and the update of particle magnitudes for next simulation step in the SU task. The position variables are implemented in DualSPHysics using double precision and the SU task is performed by using double precision for both architectures (CPU and GPU). However, the relative position of the particle within its cell (using single precision) is computed before the particle interaction and is used to compute the distance between particles on GPU implementation. This optimisation provides a significant performance improvement without loss of accuracy on GPU models where the performance ratio between double and single precision computing is 1/32. This optimisation is not applied to the CPU version of the code since the speedup on CPU is negligible.

5.2. Performance

With each particle having a constantly changing stencil of neighbouring particles, typically 40 in 2D and around 250 in 3D, the SPH method is computationally demanding. It is therefore essential to apply optimisations such as those discussed above to maximize the performance. This section shows the performance of DualSPHysics according to runtime and Particle Interactions Per Second (PIPS) using the latest sets of optimisation. The runtime of SPH simulations increases rapidly with the number of particles, since each particle interacts with more than 268 neighbouring particles in 3-D (when $2h$ is set to be four times the initial interparticle distance, for example). However, other factors influence the total execution time. The number of simulation steps per physical seconds grows when the resolution is increased or when there are rapid time-varying events requiring a small timestep during the simulation.

The distribution of the fluid in the simulation domain can increase or reduce the total number of interactions between particles. Other important factors affecting the runtime are the physical time to be simulated, the complexity of the chosen formulations and some hardware features. The use of PIPS allows measuring the performance by focusing on the actual number of particle interactions regardless of the number of simulation steps, the amount of particles or their distribution during the simulation. Note that the particle interactions are counted twice per simulation step when the predictor-corrector scheme is used.

The DualSPHysics performance was evaluated by simulating the second validation test of SPHERIC (https://spheric-sph.org/tests/test-2). This test case is a dam break flow impacting an obstacle. It was described in Kleefsman et al. [98] and validated with DualSPHysics in Crespo et al. [99]. The first two physical seconds are simulated for different resolutions, where the number of particles ranges from 116,658 to 80,353,962. Simulations were performed using different CPU and GPU models (see Tables 1 and 2 respectively). Figure 8 (a) shows the runtime of these simulations after limiting the maximum runtime to 48 hours. Runtimes are compared against the number of particles using different devices, but also the maximum number of particles that each hardware device can simulate in less than 2 days. The fastest GPU is the NVIDIA Tesla V100, which can simulate test cases with 80 million particles in less than 48 hours. The second fastest GPU is the NVIDIA GeForce RTX 2080 Ti, capable of simulating 56 million particles in approximately 36 hours. The fastest CPU can simulate 2.15 million particles in less than 48h.

Figure 8 (b) shows the GPIPS (10^9 PIPS) achieved by each device according to the number of particles. This value is about 0.14 using CPU2 (12 execution threads), close to 3 using Kepler GPUs, 12 using the GTX 1080 Ti, 23 using the RTX 2080 Ti and 29 using the Tesla V100. The performance of the computing devices increases according to the size of the problem, since a minimum number of particles is needed to make an efficient and full use of the hardware. This becomes clearer the higher the parallel computing power of the device. We can say that the device is saturated when the performance does not increase as the number of particles increases. The GPIPS value is almost constant when the number of particles is large enough to saturate the calculation device. This saturation value is reached with 100,000 particles on CPU2, 2 million particles on the Kepler GPUs and 12 million particles on the Tesla V100. These results show that the speedup of the Tesla V100 against CPU2 is more than 200x from 4 million particles onwards. The speedup of Tesla V100 against the GeForce RTX 2080 Ti is about 1.3x since the relative position of particles to the cell is used to minimise the use of double precision. This speedup is more than 5x when the particles position in double precision is used, and it is due to a performance ratio double/single precision of 1/32 for the RTX 2080 Ti.

Table 1. Main characteristics of CPU devices.

	CPU1: Intel Core i7-6700K	CPU2: Intel Core i7-8700K
Code name	Skylake	Coffe Lake
Technology [nm]	14	14
Frequency [GHz]	4.0	3.7
Physical cores	4	6
Execution threads	8	12

Table 2. Main characteristics of GPU devices.

	Tesla K20	Tesla K40	GeForce GTX Titan Black	GeForce GTX 1080 Ti	GeForce RTX 2080 Ti	Tesla V100
GPU microarchitecture	Kepler	Kepler	Kepler	Pascal	Turing	Volta
Global memory [MB]	5,063	12,207	6,084	11,264	10,989	16,128
CUDA Cores	2,496	2,880	2,880	3,584	4,352	5,120
Multiprocessors	13	15	15	28	68	80

CUDA Cores/MP	192	192	192	128	64	64
Ratio double/single	1/3	1/3	1/3	1/32	1/32	1/2
GPU Max Clock rate [MHz]	706	745	980	1,645	1,545	1,530
Memory Clock rate [MHz]	2,600	3,004	3,500	5,505	7,000	877
Memory Bus Width [bits]	320	384	384	352	352	4,096

(a)   (b)
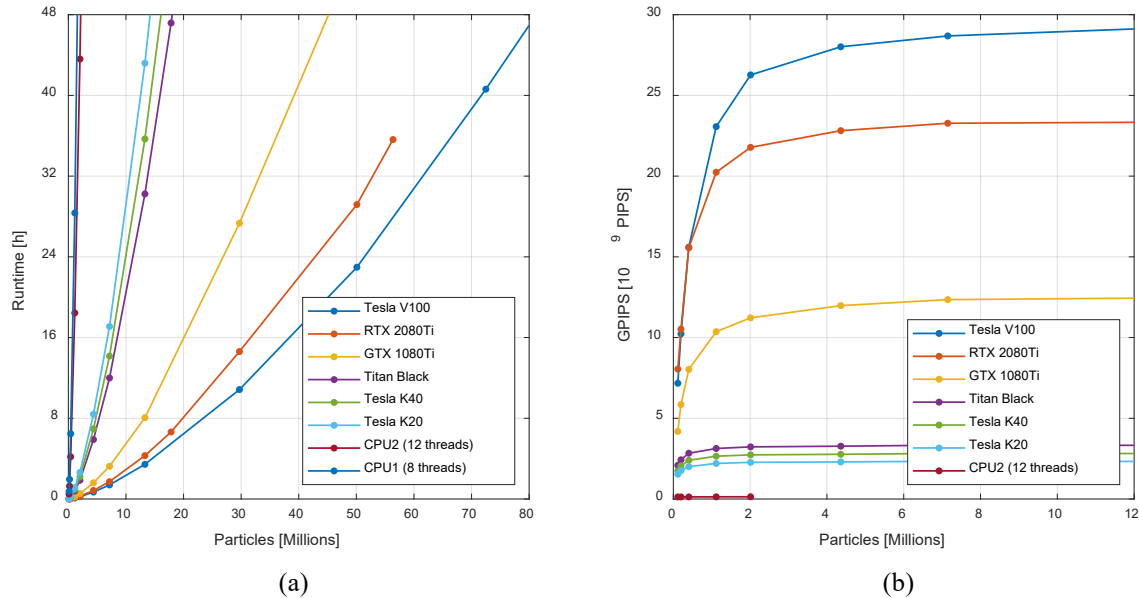
Figure 8. Runtime and Giga Particle Interactions Per Second (10^9 PIPS) of the dam break simulation according to number of particles using different hardware devices at (a) and (b) panel respectively.

The performance tests were carried out using cells of size h instead of $2h$ to improve the search for neighbouring particles. This optimisation was introduced in Domínguez et al. [14] and enables CPU and GPU speedups of 1.14x and 1.3x, respectively. However, the memory consumption increases and the maximum number of particles that a GPU can simulate is reduced. The use of cells of size of $2h$ allows simulating 6.05 million particles per Gigabyte of device memory, whereas h allows simulating 5.8 million particles per Gigabyte. Figure 9 shows the maximum number of particles according to memory size of each GPU model. The memory size of GPU devices is a major limitation for simulations requiring large numbers of particles. This limitation is removed in DualSPHysics by using a multi-GPU approach [15, 100]. This approach use Message Passing Interface (MPI) to combine the computation power of different GPUs hosted in the same or different computation nodes. This Multi-GPU version based on DualSPHysics v3.0 allows performing simulations with more than 1,000 million particles with relatively short runtimes [15].
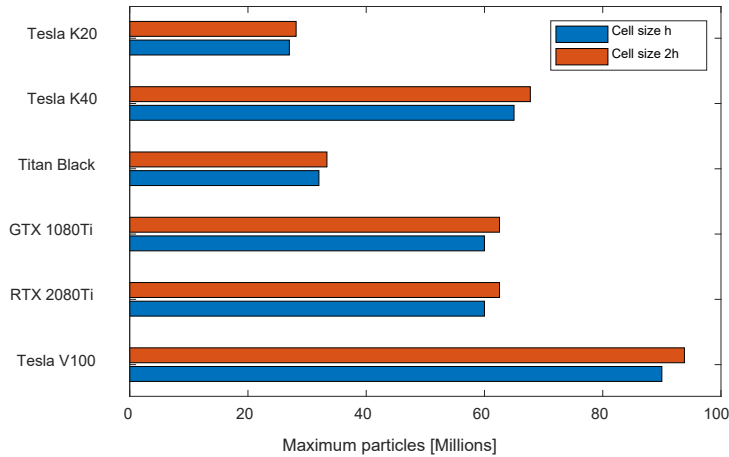
26

Figure 9. Maximum number of particles according to the memory size of each GPU model.

6. Applications

The features developed in the different versions of DualSPHysics have been validated with analytical solutions, numerical benchmarks and experimental results demonstrating accuracy and convergence as it will be described in the following examples retrieved from the literature. Table 3 includes some relevant works focused on DualSPHysics published over the last few years, with special emphasis on validated applications.

Table 3. Examples in the literature over the last years using DualSPHysics.

Manuscript	Main topic	Main achievements
Altomare et al., 2015a [82]	Coastal structures	Regular and random waves colliding with coastal structures Comparison with experimental data and analytical solutions
Altomare et al., 2015b [101]	Coastal structures	Coupling with SWASH (multi-layer piston) Comparison with experimental data
Pringgana et al., 2016 [102]	Coastal structures	Tsunami-induce bore impact on onshore structure Comparison with semi-empirical approaches
Altomare et al., 2017 [47]	Coastal structures	Automatic wave generation Passive and active wave absorption Comparison with experimental data and analytical solutions
Altomare et al., 2018 [85]	Coastal structures	Coupling with SWASH (relaxation zone) Comparison with experimental data and theoretical solutions
González-Cao et al., 2018a [103]	Coastal structures	Comparison with experimental data and mesh-based models
Zhang et al., 2018 [40]	Coastal structures	Regular waves interacting with porous breakwater Time series of wave run-up Comparison with experimental data
Rota-Roselli et al., 2018 [104]	Coastal structures	Optimization of model setup for wave propagation through genetic algorithms Comparison with theoretical solutions
Verbrugghe et al., 2019b [86]	Coastal structures	Coupling with OceanWave3D Open boundary conditions

Domínguez et al., 2019a [48]	Coastal structures	Solitary waves (different theories) Comparison with experimental data and analytical solutions
Lowe et al., 2019 [105]	Coastal structures	Spilling and plunging wave breaking Comparison with experimental data
Subramaniam et al., 2019 [106]	Coastal structures	3D modeling of wave-run on convex and concave dikes Comparison with experimental results and mesh-based methods
Altomare et al., 2020 [107]	Coastal structures	Large-Scale structure under extreme wave conditions Wave generation by Open boundaries Pier failure analysis
Canelas et al., 2015 [108]	Fluid-driven objects	Buoyancy of floating bodies Comparison with experimental data and mesh-based models
Canelas et al., 2016 [87]	Fluid-driven objects	DEM technique Interaction between floating bodies Comparison with experimental data
Canelas et al., 2017 [109]	Fluid-driven objects	DEM technique for Debris flow Interaction between floating bodies Comparison with experimental data
Domínguez et al., 2019b [93]	Fluid-driven objects	Simulation of freely and moored floating box Coupling with MoorDyn Comparison with experimental data
Canelas et al., 2018 [91]	Multiphysics problems	Coupling with Chrono Comparison with experimental data
Ropero-Giralda et al., 2020 [110]	Multiphysics problems	Simulation of Point-absorber wave energy with Power Take-Off system Coupling with Chrono Comparison with experimental data Efficiency and survivability analysis
Crespo et al., 2017 [111]	Wave Energy Converters (WECs)	Simulation of fixed and floating oscillating water column with open chamber Comparison with experimental data
Verbrugghe et al., 2018 [84]	Wave Energy Converters (WECs)	Coupling with OceanWave3D Simulation of oscillating water column and point absorber
Brito et al., 2020 [112]	Wave Energy Converters (WECs)	Simulation of oscillating wave surge converter Coupling with Chrono Comparison with experimental data
González-Cao et al., 2018b [113]	Hydraulic problems	Safety of spillways Open boundary conditions
Novak et al., 2019 [114]	Hydraulic problems	Vertical slot fishway Comparison with experimental data
Nóbrega et al., 2020 [115]	Hydraulic problems	Smooth and stepped spillways Comparison with experimental data
Mokos et al., 2015 [70]	Multi-phase air-water	GPU implementation of gas-liquid solver Comparison with experimental data
Mokos et al., 2017 [68]	Multi-phase air-water	Gas-liquid solver with shifting algorithms Comparison with experimental data
Fourtakas and Rogers, 2016 [78]	Multi-phase soil-water	State-of-the-art non-Newtonian and multi-phase solver Comparison with experimental data
Zubeldia et al., 2018 [116]	Multi-phase soil-water	Improvements over the erosion characteristics of the multi-phase soil-water solver Comparison with experimental data

One of the major fields of application of DualSPHysics has been coastal engineering, where the model has been employed as a complementary tool to experimental measurements. A fundamental requirement for proper capturing of wave-structure interaction phenomena is the accurate modelling of wave generation and propagation. Altomare et al. [47] implemented detailed and accurate wave generation

and absorption schemes for long-crested second-order waves (Section 3.1). The validation of the results from the numerical model was carried out against analytical solutions and experimental results for both regular and random waves. Zhang et al. [40] studied wave run-up for the design of coastal defences. First, run-up time series were validated against experiments using a porous breakwater for which the exact geometry was used in the numerical tank (Figure 10). Then, the model was applied to analyse a real sea dike from the coast of Chongwu in China.
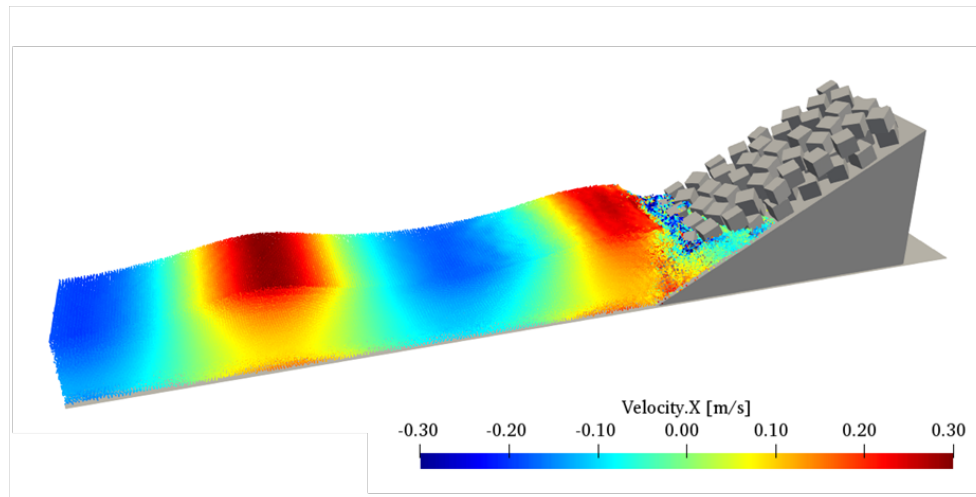
Figure 10. Simulation snapshot of an armour breakwater presented in Zhang et al. [40]. Colour of the particles represent the longitudinal velocity. Complete simulation here: https://youtu.be/W9uIsGomTvE

As previously described in Section 4.1, several efficient wave propagation models can be coupled with SPH-based models to reduce the computational cost in order to propagate the incoming wave far enough from the coastal structure. The two-way coupling with OceanWave3D and based on open boundary conditions was described in Verbrugghe et al. [86], where the experimental data from Ren et al. [117] was used to validate the method and to study the response of a floating box in terms of its three degrees of freedom (heave, surge and pitch).

The accuracy of DualSPHysics to simulate fluid-driven objects, including collisions between solids, was analysed in Canelas et al. [87]. The solid-solid interactions solved with Discrete Element Method (DEM) in conjunction with DualSPHysics was first introduced in that work. Free surface flows and rigid body dynamics were solved in the same meshless framework. An experimental campaign was used to validate this coupling approach. The experiments represented a dam break impacting a set of blocks disposed in different configurations. Blocks were tracked and experimental positions were compared with the numerical ones. The model showed to be accurate and capable of handling highly complex interactions. Later, this SPH-DCDEM approach was tested by comparing numerical results with an experiment designed for stony debris flows in a slit check dam in Canelas et al. [109]. The simulations (Figure 11) aim to mimic the experiment where the solid material is introduced through a hopper, assuring a constant solid discharge. DCDEM was used to solve thousands of time-evolving interactions due to the many simultaneous contacts between the particles that form the different stones. Validation included the comparison of the sediment trapping efficiency, which is obtained by measuring the solid discharges at a position sufficiently upstream and immediately downstream of the dam. Numerical results showed a good agreement with experimental data.
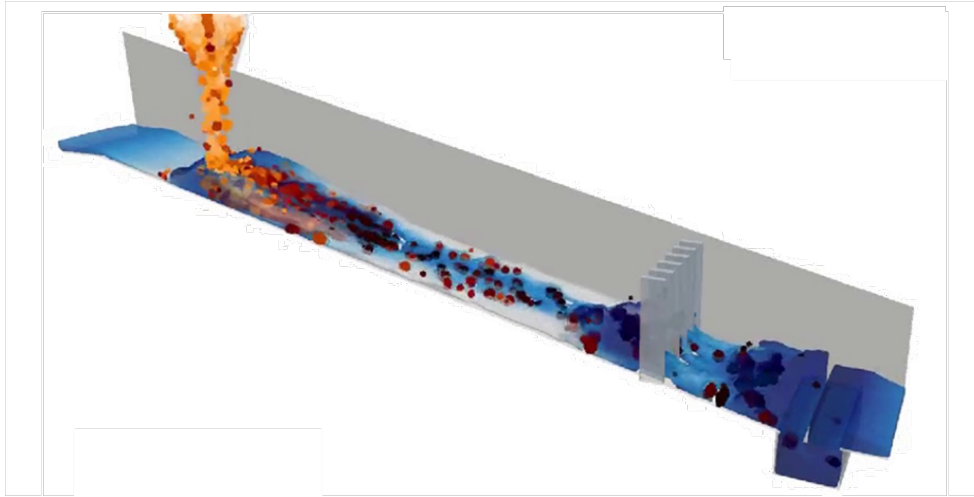
Figure 11. Snapshot of the Simulation of debris flow simulation performed in Canelas et al. [109]. Complete simulation here: https://youtu.be/trbkKUTuB9I.

The validation of moored floating objects under the action of waves using DualSPHysics can be found in Domínguez et al. [93], where the numerical results of nonlinear waves interacting with a freely floating box were compared with the experimental data of Ren et al. [117]. Good agreement was obtained for the motions of the box (heave, surge and pitch). The coupling with the MoorDyn library was validated against data from scale model tests generated during the experimental campaigns for the European MaRINET2 EsflOWC project. The physical tests consisted of a floating box moored to the bottom through four chains. Figure 12 includes an instant of the simulation of the floating box moored with four lines. Numerical and experimental motions of the box and tensions in the lines were in good agreement for different regular waves. However, the model still needs to be validated under several sea states (also irregular waves) including those where an incoming wave train approaches the natural frequency of the structure.
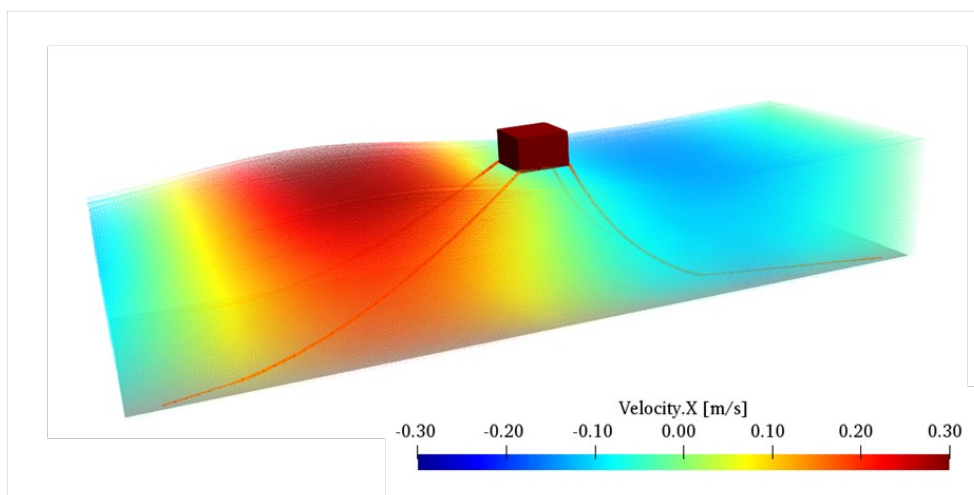
Figure 12. Simulation snapshot of the moored box under regular waves studied in Domínguez et al. [93]. Colour of the particles represent the horizontal velocity. Complete simulation here: https://youtu.be/UaRQII8UgQs.

A detailed description of the coupling between DualSPHysics and Project Chrono was included in Canelas et al. [91]. A dam break flow impacting a supported platform was simulated in order to validate

frictional contacts. The numerical times for movement initiation and the angle inversion times were consistent with the experimental ones. Secondly, different pendulum systems were tested, including a system set-up with a hinge (gravity pendulum) and another one with a spring (spring pendulum). The numerical angular rotation was in agreement with the experimental results. Canelas et al. [91] also included some working examples and interesting applications of this coupling with Project Chrono. Figure 13 includes a snapshot of the simulation of the WaveStar machine, a wave energy converter (WEC) that consists of a row of half-submerged buoys. The oscillatory motion of the buoys interacting with waves is converted to electricity with a Power Take-Off system (PTO) that uses the rotational speed of the arm connected to the buoy. DualSPHysics coupled with Project Chrono can solve both the interaction of the incoming waves with the buoys and the hinge restrictions in the connections between the arms and the buoys.
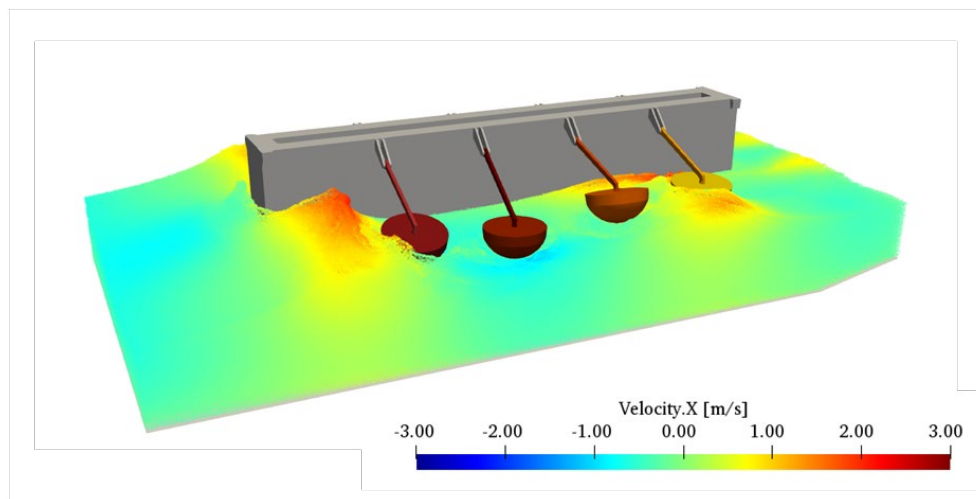
Figure 13. A snapshot of the simulation of the WaveStar machine presented in Canelas et al. [91]. Colour of the particles represent the horizontal velocity. Complete simulation here: https://youtu.be/JeD89PMiLLQ.

The latest version of DualSPHysics that include coupling with MoorDyn and Project Chrono allow studying the hydrodynamic response of different wave energy converters (WEC), including the numerical behaviour of their PTO. One of the examples that include an exhaustive comparison with experimental results can be found in Brito et al. [112], where an oscillating wave surge converter (OWSC) was studied. The version of DualSPHysics employed in Brito et al. [112] resolved the wave-flap interaction with SPH and the flap-mechanical constraints interaction with Project Chrono. The code was validated using the experimental data of an OWSC with mechanical constraints, including the effects of nonlinear constraints of the hydraulic PTO system and the frictional contacts between flap and bearings. The comparison between numerical and experimental results (including free-surface elevation and angular velocity of the flap) proved that the numerical simulation can predict the hydrodynamic response of the OWSC accurately for both unidirectional regular and irregular waves. Figure 14 depicts a frame of the simulation of OWSC using DualSPHysics. Once validated, the numerical simulation tool was applied to study the influence of several mechanical constraints, PTO damping characteristics and flap inertia on the efficiency of the device.
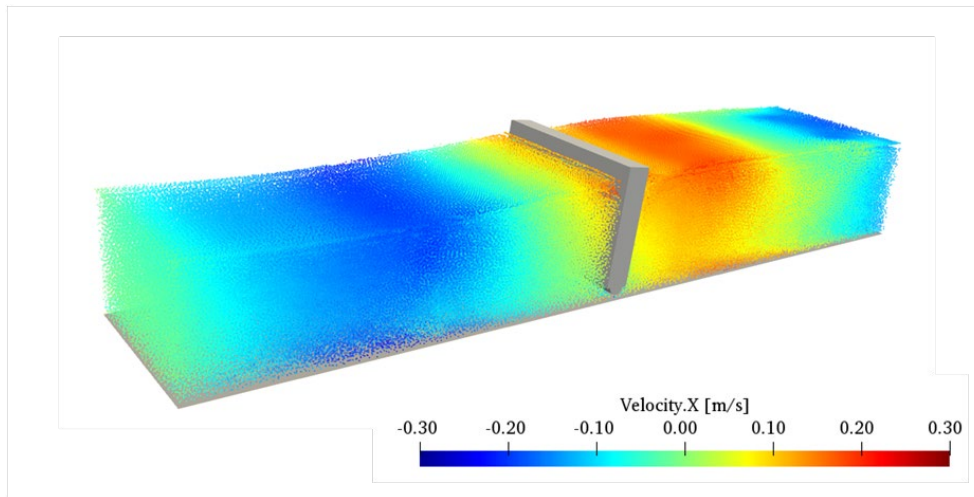
Figure 14. Simulation snapshot of the OWSC analysed in detail in Brito et al. [112]. Colour of the particles represent the horizontal velocity. Complete simulation here: https://youtu.be/-syzUy8qVTE.

Hydraulic engineering is another interesting and common field of application of DualSPHysics. A numerical study of a vertical slot fishway (VSF) in 3-D was performed in Novak et al. [114]. This work used the novel open boundary conditions based on buffer regions to reproduce the free surface turbulent subcritical flow presents in this VSF application. The results show a good agreement with experimental data in discharges, velocity profiles and water elevation at different locations. The Figure 15 shows a snapshot of the VSF simulation where the colour of particles represents the longitudinal velocity.
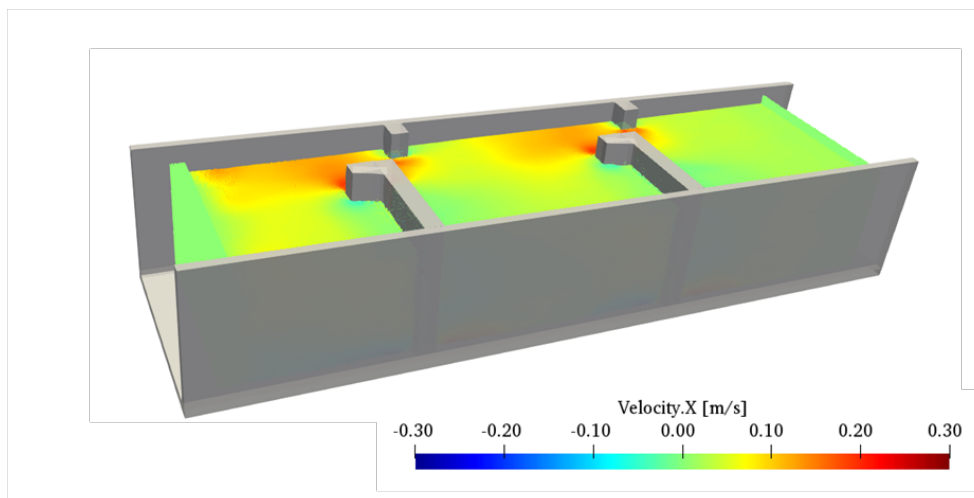
Figure 15. Snapshot of the simulation of the Vertical Slot Fishway analysed in Novak et al. [114]. Colour of the particles represent the horizontal velocity.

Some applications cannot be adequately simulated by a single-phase representation. Since version 3.4 of DualSPHysics, the multi-phase capability has been added to the code to respond to a growing number of requests for this functionality from users, but also to extend the applicability of the code to liquid-gas and liquid-sediment interactions problems. The code has been used for the simulation of 3-D dam breaks impacting structures and fuel-tank sloshing [68, 70], where the presence of the gas phase is necessary to obtain a closer agreement with experimental data. During this process, the hardware acceleration and increased resolution led to the formation of unphysical voids in the flow, which required the development of a novel multi-phase shifting algorithm [68]. Figure 16 shows the results with the

32

application of multi-phase shifting. The non-Newtonian formulation presented in this paper in Section 3.5 is the result of multiple developments for liquid-sediment interaction where water flows over erodible sediment [78]. This requires a combination of (1) unyielded sediment, (2) a yield surface predicted using the Drucker-Prager model, (3) yielded sediment predicted using the Herschel-Bulckley-Papanastiou (HBP) non-Newtonian flow model, (4) a bed load predicted by the Sheilds' criterion, (5) fluid with entrained sediment [116]. Figure 17 shows the results from Fourtakas and Rogers [78], simulating a dam break simulation capturing 3-D experimental erodible sediment profiles with consistent repeatability. The results showed encouraging agreement with experiments for the time history of the water level and the final sediment profiles along the tank.
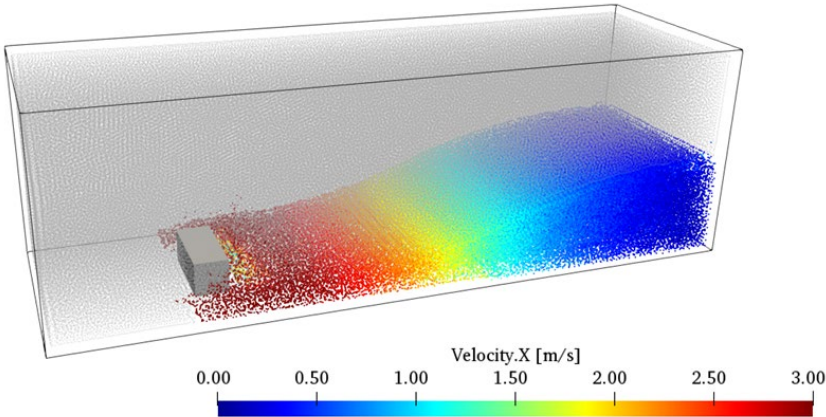
Figure 16. Snapshot of the simulation of the dam break impacting an obstacle using the multi-phase air-water code presented in Mokos et al. [68]. Colour of the particles represent the horizontal velocity. Complete simulation here: https://youtu.be/Q2RZWo-APC8.
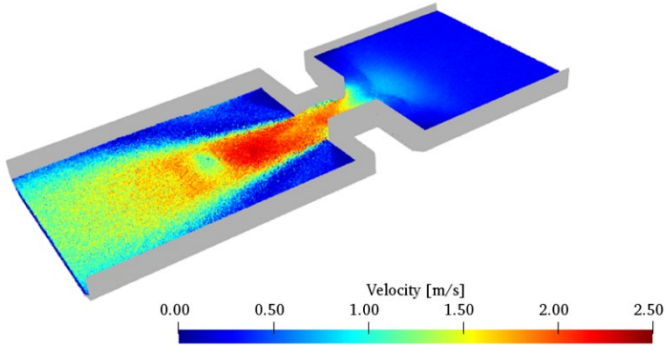

Figure 17. Simulation snapshot of a 3-D dam break over an erodible bed presented in Fourtakas and Rogers [78]. Colour of the particles represent the velocity.

As mentioned above, DualSPHysics is an open source code that can be adapted by users to solve particular engineering problems that were not in the initial scope of the solver. Table 4 shows some examples of these tailored solutions that were implemented with or without the collaboration of the developers. There is a wide range of applications that goes from fundamental problems that cover the development of an incompressible version of the code [118], new boundary conditions [119] or the use

of adaptive spatial resolution [120]; to applications like heat transfer [121] or the decay of radionuclides [122]. Due to the different nature of the phenomena under study these new developments do not necessarily appear in further releases.

Table 4. Tailored solutions based on DualSPHysics.

Manuscript	Main topic	Main achievements
Mayoral-Villa et al., 2016 [122]	Radionuclides in aqueous dissolution	SPH technique to model advection, diffusion, and radioactive decay Benchmark test cases
Chow et al., 2018 [118]	Incompressible SPH	Implementation of a strictly incompressible SPH solver Comparison with experimental data
Green et al., 2018 [119]	Boundary conditions	Ghost particles with correction of Adami et al [123] Long duration sloshing simulations Comparison with experimental data
Hosain et al., 2019 [121]	Heat Transfer modelling	Benchmarking of thermal solutions, comparison with Finite Volume Method Industrial processes
Leonardi et al., 2019 [120]	Variable resolution	Variable particle resolution approach with explicit kernel correction

7. Outlook

This paper has presented the latest novel developments of the community open-source SPH code DualSPHysics. The paper has outlined the SPH formulation, code structure and validations of the latest version 5.0. Table 5 summarizes the features available in the three approaches included in the DualSPHysics package: Single-phase, Multi-phase Non-Newtonian and Multi-phase Liquid and gas. The sketch presented in Figure 18 summarizes the main functionalities of the DualSPHysics package and the present range of applications. Several of the novel developments presented are motivated towards solving the SPHERIC Grand Challenges: GC2-Boundary conditions, GC4-Coupling, GC5-Applicability to industry. Indeed, GC5 perhaps receives less attention than it deserves, and therefore it needs special efforts to encourage the uptake and use of SPH as a simulation tool in industry.

Table 5. Main features and formulations of each approach in the DualSPHysics package v5.0.

| Feature	Single-phase	Multi-phase: Non-Newtonian	Multi-phase: Liquid and gas
Hardware	CPU multi-core; single GPU		
Dimension	2-D simulations; 3-D simulations		
Kernel functions	Cubic spline; Wendland		
Density diffusion terms	Molteni and Colagrossi term; Fourtakas term		Molteni and Colagrossi term
Dissipation terms	Artificial viscosity; Laminar viscosity with SPS		
Time integrators	Verlet scheme; Symplectic scheme; Variable timestep		
Boundary conditions	DBC; mDBC	DBC	DBC
Boundary limit treatments	Periodic conditions; Open boundaries (only fluid phase)		Periodic conditions
Particle shifting algorithm	Fickian diffusion algorithm	Phase-dependent Fickian diffusion algorithm	
Fluid-driven objects	Floating bodies		
Wave generation and absorption	Regular waves, Irregular waves; Solitary waves; AWAS		
Coupling with wave propagation models	Multi-layered piston; Relaxation zone; Open boundaries		Multi-layered piston
Coupling with other models	DEM; Project Chrono; MoorDyn		
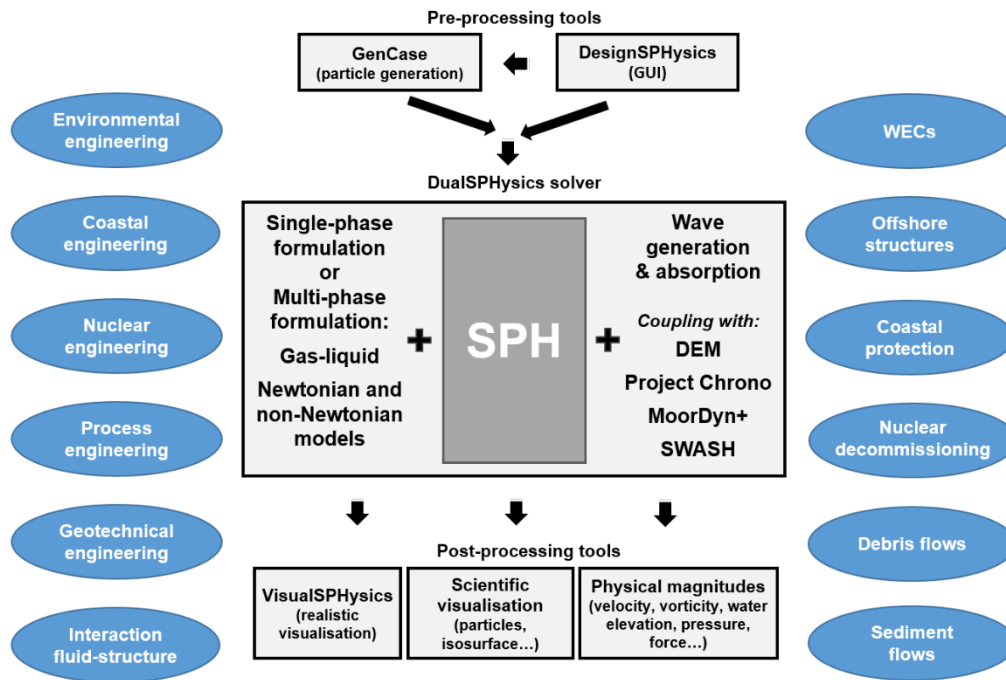
Figure 18. Sketch of DualSPHysics package with components, functionalities and applications.

The resolution required to capture the correct physics of engineering simulations is often beyond the computational capabilities of GPU-accelerated or massively parallel CFD software packages. It is potentially more challenging for SPH since its Lagrangian nature means that the physical processes that have the potential to be implemented offer a tantalising prospect but come with resolution and hence computational demands that are beyond most software packages. Examples include multi-phase applications where the physics can be extremely challenging over a large range of time and length scales. The development of DualSPHysics is specifically aimed at meeting these challenges and will continue to develop in this direction.

Within the context of the rapidly advances and diversification in the field of computer hardware, DualSPHysics is highly optimised to exploit the Nvidia GPU hardware with optimised CUDA kernels, templates, and other optimisations. When general-purpose computing on graphics processing units (GPGPU) first gained traction in the scientific computing community, GPUs were entirely separate devices from the CPU, with CUDA kernels being launched solely from the CPU. Now, the general trend in the development of hardware for computing is for increased integration of the streaming multiprocessors within the motherboard. The most recent multi-GPU computer released by Nvidia (A100) features 16 GPUs on a single board connected with a NVLink (https://www.nvidia.com/en-us/data-center/nvlink). Similar developments are taking place with the other major hardware vendors. It is clear that increasing integration will be the future of scientific computing with GPGPUs. In order to exploit this, DualSPHysics will evolve to meet the emerging challenges of new hardware. Ultimately, it is possible that SPH will offer a viable method for the biggest challenge that the scientific computing community is now facing: developing algorithms to exploit the ever-emerging hardware for the newborn Exascale computing revolution.

Acknowledgements

Conflict of Interest: The authors declare that they have no conflict of interest.

References

1. Shadloo MS, Oger G, Le Touzé D (2016) Smoothed particle hydrodynamics method for fluid flows, towards industrial applications: Motivations, current state, and challenges. Comput Fluids 136:11–34. https://doi.org/10.1016/j.compfluid.2016.05.029

2. Gotoh H, Khayyer A (2018) On the state-of-the-art of particle methods for coastal and ocean engineering. Coast Eng J 60:79–103. https://doi.org/10.1080/21664250.2018.1436243

3. Mogan SRC, Chen D, Hartwig JW, et al (2018) Hydrodynamic analysis and optimization of the Titan submarine via the SPH and Finite–Volume methods. Comput Fluids. https://doi.org/10.1016/j.compfluid.2018.08.014

4. Manenti, Wang, Domínguez, et al (2019) SPH Modeling of Water-Related Natural Hazards. Water 11:1875. https://doi.org/10.3390/w11091875

5. Violeau D, Rogers BD (2016) Smoothed particle hydrodynamics (SPH) for free-surface flows: past, present and future. J Hydraul Res 54:1–26. https://doi.org/10.1080/00221686.2015.1119209

6. Vacondio R, Altomare C, De Leffe M, et al (2020) Grand challenges for Smoothed Particle Hydrodynamics numerical schemes. Comput. Part. Mech.

7. Oger G, Le Touzé D, Guibert D, et al (2016) On distributed memory MPI-based parallelization of SPH codes in massive HPC context. Comput Phys Commun 200:1–14. https://doi.org/10.1016/j.cpc.2015.08.021

8. Yeylaghi S, Moa B, Oshkai P, et al (2017) ISPH modelling for hydrodynamic applications using a new MPI-based parallel approach. J Ocean Eng Mar Energy 3:35–50. https://doi.org/10.1007/s40722-016-0070-6

9. Guo X, Rogers BD, Lind S, Stansby PK (2018) New massively parallel scheme for Incompressible Smoothed Particle Hydrodynamics (ISPH) for highly nonlinear and distorted flow. Comput Phys Commun 233:16–28. https://doi.org/10.1016/j.cpc.2018.06.006

10. Hérault A, Bilotta G, Dalrymple RA (2010) SPH on GPU with CUDA. J Hydraul Res 48:74–79. https://doi.org/10.1080/00221686.2010.9641247

11. Ramachandran P (2016) PySPH: a reproducible and high-performance framework for smoothed particle hydrodynamics. In: Proceedings of the 3rd International Conference on Particle-Based Methods (Particles 2013). Stuttgart, pp 122–129

12. Cercos-Pita JL (2015) AQUAgpusph, a new free 3D SPH solver accelerated with OpenCL. Comput Phys Commun. https://doi.org/10.1016/j.cpc.2015.01.026

13. Domínguez JM, Crespo AJC, Gómez-Gesteira M, Marongiu JC (2011) Neighbour lists in smoothed particle hydrodynamics. Int J Numer Methods Fluids 67:2026–2042. https://doi.org/10.1002/fld.2481

14. Domínguez JM, Crespo AJC, Gómez-Gesteira M (2013) Optimization strategies for CPU and GPU implementations of a smoothed particle hydrodynamics method. Comput Phys Commun 184:617–627. https://doi.org/10.1016/j.cpc.2012.10.015

15. Domínguez JM, Crespo AJC, Valdez-Balderas D, et al (2013) New multi-GPU implementation for smoothed particle hydrodynamics on heterogeneous clusters. Comput Phys Commun 184:1848–1860.

https://doi.org/10.1016/j.cpc.2013.03.008

16. Dehnen W, Aly H (2012) Improving convergence in smoothed particle hydrodynamics simulations without pairing instability. Mon Not R Astron Soc. https://doi.org/10.1111/j.1365-2966.2012.21439.x

17. Liu MB, Liu GR, Lam KY (2003) Constructing smoothing functions in smoothed particle hydrodynamics with applications. J Comput Appl Math 155:263–284. https://doi.org/10.1016/S0377-0427(02)00869-5

18. Monaghan JJ (1992) Smoothed Particle Hydrodynamics. Annu Rev Astron Astrophys 30:543–574. https://doi.org/10.1146/annurev.aa.30.090192.002551

19. Wendland H (1995) Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. Adv Comput Math 4:389–396. https://doi.org/10.1007/BF02123482

20. Robinson MJ (2010) Turbulence and viscous mixing using smoothed particle hydrodynamics. Monash University

21. Monaghan JJ (2005) Smoothed particle hydrodynamics. Reports Prog Phys 68:1703–1759. https://doi.org/10.1088/0034-4885/68/8/R01

22. Violeau D (2012) Fluid Mechanics and the SPH Method. Oxford University Press

23. Antuono M, Colagrossi A, Marrone S (2012) Numerical diffusive terms in weakly-compressible SPH schemes. Comput Phys Commun 183:2570–2580. https://doi.org/10.1016/j.cpc.2012.07.006

24. Bonet J, Lok T-SL (1999) Variational and momentum preservation aspects of Smooth Particle Hydrodynamic formulations. Comput Methods Appl Mech Eng 180:97–115. https://doi.org/10.1016/S0045-7825(99)00051-1

25. Molteni D, Colagrossi A (2009) A simple procedure to improve the pressure evaluation in hydrodynamic context using the SPH. Comput Phys Commun 180:861–872. https://doi.org/10.1016/j.cpc.2008.12.004

26. Fourtakas G, Dominguez JM, Vacondio R, Rogers BD (2019) Local uniform stencil (LUST) boundary condition for arbitrary 3-D boundaries in parallel smoothed particle hydrodynamics (SPH) models. Comput Fluids 190:346–361. https://doi.org/10.1016/j.compfluid.2019.06.009

27. Monaghan J., Gingold R. (1983) Shock simulation by the particle method SPH. J Comput Phys 52:374–389. https://doi.org/10.1016/0021-9991(83)90036-0

28. Español P, Revenga M (2003) Smoothed dissipative particle dynamics. Phys Rev E 67:026705. https://doi.org/10.1103/PhysRevE.67.026705

29. Y.M. Lo E, Shao S (2002) Simulation of near-shore solitary wave mechanics by an incompressible SPH method. Appl Ocean Res 24:275–286. https://doi.org/10.1016/S0141-1187(03)00002-6

30. Colagrossi A, Antuono M, Souto-Iglesias A, Le Touzé D (2011) Theoretical analysis and numerical verification of the consistency of viscous smoothed-particle-hydrodynamics formulations in simulating free-surface flows. Phys Rev E - Stat Nonlinear, Soft Matter Phys. https://doi.org/10.1103/PhysRevE.84.026705

31. Gotoh H, Shibahara T, Sakai T (2001) Sub-particle-scale Turbulence Model for the MPS Method - Lagrangian Flow Model for Hydraulic Engineering. Adv Methods Comput Fluid Dyn 339–347

32. Dalrymple RA, Rogers BD (2006) Numerical modeling of water waves with the SPH method. Coast Eng 53:141–147. https://doi.org/10.1016/j.coastaleng.2005.10.004

33. Monaghan JJ (1994) Simulating Free Surface Flows with SPH. J Comput Phys 110:399–406. https://doi.org/10.1006/jcph.1994.1034

34. Batchelor GK (2000) An Introduction to Fluid Dynamics. Cambridge University Press

35. Verlet L (1967) Computer 'Experiments' on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. Phys Rev 159:98–103. https://doi.org/10.1103/PhysRev.159.98

36. Leimkuhler B, Matthews C (2015) Introduction. In: Molecular Dynamics: With Deterministic and

Stochastic Numerical Methods. pp 1–51

37. Parshikov AN, Medin SA, Loukashenko II, Milekhin VA (2000) Improvements in SPH method by means of interparticle contact algorithm and analysis of perforation tests at moderate projectile velocities. Int J Impact Eng 24:779–796. https://doi.org/10.1016/S0734-743X(99)00168-2

38. Monaghan JJ, Kos A (1999) Solitary Waves on a Cretan Beach. J Waterw Port, Coastal, Ocean Eng 125:145–155. https://doi.org/10.1061/(ASCE)0733-950X(1999)125:3(145)

39. Altomare C, Crespo AJC, Rogers BD, et al (2014) Numerical modelling of armour block sea breakwater with smoothed particle hydrodynamics. Comput Struct 130:34–45. https://doi.org/10.1016/j.compstruc.2013.10.011

40. Zhang F, Crespo A, Altomare C, et al (2018) DualSPHysics: A numerical tool to simulate real breakwaters. J Hydrodyn 30:95–105. https://doi.org/10.1007/s42241-018-0010-0

41. English A, Domínguez JM, Vacondio R, et al Modified dynamic boundary conditions (mDBC) for general purpose smoothed particle hydrodynamics (SPH): application to tank sloshing, dam break and fish pass problems. Comput Part Mech submitted:

42. Crespo AJC, Gómez-Gesteira M, Dalrymple RA (2007) Boundary conditions generated by dynamic particles in SPH methods. Comput Mater Contin 5:173–184. https://doi.org/10.3970/cmc.2007.005.173

43. Marrone S, Antuono M, Colagrossi A, et al (2011) δ-SPH model for simulating violent impact flows. Comput Methods Appl Mech Eng 200:1526–1542. https://doi.org/10.1016/j.cma.2010.12.016

44. Liu MB, Liu GR (2006) Restoring particle consistency in smoothed particle hydrodynamics. Appl Numer Math 56:19–36. https://doi.org/10.1016/j.apnum.2005.02.012

45. Lind SJ, Xu R, Stansby PK, Rogers BD (2012) Incompressible smoothed particle hydrodynamics for free-surface flows: A generalised diffusion-based algorithm for stability and validations for impulsive flows and propagating waves. J Comput Phys 231:1499–1523. https://doi.org/10.1016/j.jcp.2011.10.027

46. Skillen A, Lind S, Stansby PK, Rogers BD (2013) Incompressible smoothed particle hydrodynamics (SPH) with reduced temporal noise and generalised Fickian smoothing applied to body–water slam and efficient wave–body interaction. Comput Methods Appl Mech Eng 265:163–173. https://doi.org/10.1016/j.cma.2013.05.017

47. Altomare C, Domínguez JM, Crespo AJC, et al (2017) Long-crested wave generation and absorption for SPH-based DualSPHysics model. Coast Eng 127:37–54. https://doi.org/10.1016/j.coastaleng.2017.06.004

48. Domínguez JM, Altomare C, Gonzalez-Cao J, Lomonaco P (2019) Towards a more complete tool for coastal engineering: solitary wave generation, propagation and breaking in an SPH-based model. Coast Eng J 61:15–40. https://doi.org/10.1080/21664250.2018.1560682

49. Biesel F, Suquet F (1951) Etude theorique d'un type d'appareil a la houle. La Houille Blanche 2:157–160

50. Hughes SA (1993) Physical Models and Laboratory Techniques in Coastal Engineering, Advanced S. WORLD SCIENTIFIC

51. Liu Z, Frigaards P (2001) Generation and Analysis of Random Waves. Aalborg Universitet, Denmark

52. Hansen EON, Sorensen T, Gravesen H, et al (1981) Correct reproduction of group-induced long waves. PROC SEVENTEENTH Coast ENGNG CONF (SYDNEY, Aust MAR 23-28, 1980)

53. Barthel V, Mansard EPD, Sand SE, Vis FC (1983) Group bounded long waves in physical models. Ocean Eng 10:261–294. https://doi.org/10.1016/0029-8018(83)90012-4

54. Goring DG (1978) TSUNAMIS - THE PROPAGATION OF LONG WAVES ONTO A SHELF. Calif Inst Technol W M Keck Lab Hydraul Water Resour Rep KH-R

55. Schäffer HA, Klopman G (2000) Review of multidirectional active wave absorption methods. J Waterw Port, Coast Ocean Eng. https://doi.org/10.1061/(ASCE)0733-950X(2000)126:2(88)

56. Monaghan JJ, Kos A, Issa N (2003) Fluid motion generated by impact. J Waterw Port, Coast Ocean Eng. https://doi.org/10.1061/(ASCE)0733-950X(2003)129:6(250)

57. Ferrand M, Joly A, Kassiotis C, et al (2017) Unsteady open boundaries for SPH using semi-analytical conditions and Riemann solver in 2D. Comput Phys Commun 210:29–44. https://doi.org/10.1016/j.cpc.2016.09.009

58. Leroy A, Violeau D, Ferrand M, Kassiotis C (2014) Unified semi-analytical wall boundary conditions applied to 2-D incompressible SPH. J Comput Phys. https://doi.org/10.1016/j.jcp.2013.12.035

59. Leroy A, Violeau D, Ferrand M, et al (2016) A new open boundary formulation for incompressible SPH. Comput Math with Appl 72:2417–2432. https://doi.org/10.1016/j.camwa.2016.09.008

60. Hirschler M, Kunz P, Huber M, et al (2016) Open boundary conditions for ISPH and their application to micro-flow. J Comput Phys 307:614–633. https://doi.org/10.1016/j.jcp.2015.12.024

61. Lastiwka M, Basa M, Quinlan NJ (2009) Permeable and non-reflecting boundary conditions in SPH. Int J Numer Methods Fluids 61:709–724. https://doi.org/10.1002/fld.1971

62. Federico I, Marrone S, Colagrossi A, et al (2012) Simulating 2D open-channel flows through an SPH model. Eur J Mech - B/Fluids 34:35–46. https://doi.org/10.1016/j.euromechflu.2012.02.002

63. Vacondio R, Rogers BD, Stansby PK, Mignosa P (2012) SPH Modeling of Shallow Flow with Open Boundaries for Practical Flood Simulation. J Hydraul Eng 138:530–541. https://doi.org/10.1061/(ASCE)HY.1943-7900.0000543

64. Khorasanizade S, Sousa JMM (2016) An innovative open boundary treatment for incompressible SPH. Int J Numer Methods Fluids 80:161–180. https://doi.org/10.1002/fld.4074

65. Tafuni A, Domínguez JM, Vacondio R, Crespo A (2017) Accurate and efficient SPH open boundary conditions for real 3-D engineering problems. In: Proceedings of SPHERIC 2017. pp 346–354

66. Tafuni A, Domínguez JM, Vacondio R, Crespo AJC (2018) A versatile algorithm for the treatment of open boundary conditions in Smoothed particle hydrodynamics GPU models. Comput Methods Appl Mech Eng 342:604–624. https://doi.org/10.1016/j.cma.2018.08.004

67. Verbrugghe T, Domínguez JM, Altomare C, et al (2019) Non-linear wave generation and absorption using open boundaries within DualSPHysics. Comput Phys Commun 240:46–59. https://doi.org/10.1016/j.cpc.2019.02.003

68. Mokos A, Rogers BD, Stansby PK (2017) A multi-phase particle shifting algorithm for SPH simulations of violent hydrodynamics with a large number of particles. J Hydraul Res 55:143–162. https://doi.org/10.1080/00221686.2016.1212944

69. Nugent S, Posch HA (2000) Liquid drops and surface tension with smoothed particle applied mechanics. Phys Rev E 62:4968–4975. https://doi.org/10.1103/PhysRevE.62.4968

70. Mokos A, Rogers BD, Stansby PK, Domínguez JM (2015) Multi-phase SPH modelling of violent hydrodynamics on GPUs. Comput Phys Commun 196:304–316. https://doi.org/10.1016/j.cpc.2015.06.020

71. Colagrossi A, Landrini M (2003) Numerical simulation of interfacial flows by smoothed particle hydrodynamics. J Comput Phys 191:448–475. https://doi.org/10.1016/S0021-9991(03)00324-3

72. Brackbill J., Kothe D., Zemach C (1992) A continuum method for modeling surface tension. J Comput Phys 100:335–354. https://doi.org/10.1016/0021-9991(92)90240-Y

73. Hu XY, Adams NA (2006) A multi-phase SPH method for macroscopic and mesoscopic flows. J Comput Phys 213:844–861. https://doi.org/10.1016/j.jcp.2005.09.001

74. Lafaurie B, Nardone C, Scardovelli R, et al (1994) Modelling Merging and Fragmentation in Multiphase Flows with SURFER. J Comput Phys 113:134–147. https://doi.org/10.1006/jcph.1994.1123

75. Wu J, Yu S-T, Jiang B-N (1998) Simulation of two-fluid flows by the least-squares finite element method

using a continuum surface tension model. Int J Numer Methods Eng 42:583–600. https://doi.org/10.1002/(SICI)1097-0207(19980630)42:4<583::AID-NME341>3.0.CO;2-M

76. Papanastasiou TC (1987) Flows of Materials with Yield. J Rheol (N Y N Y) 31:385–404. https://doi.org/10.1122/1.549926

77. Mitsoulis E (2007) Flows of viscoplastic materials: Models and computations. Rheol Rev 135–178

78. Fourtakas G, Rogers BD (2016) Modelling multi-phase liquid-sediment scour and resuspension induced by rapid flows using Smoothed Particle Hydrodynamics (SPH) accelerated with a Graphics Processing Unit (GPU). Adv Water Resour 92:186–199. https://doi.org/10.1016/j.advwatres.2016.04.009

79. Morris JP, Fox PJ, Zhu Y (1997) Modeling Low Reynolds Number Incompressible Flows Using SPH. J Comput Phys 136:214–226. https://doi.org/10.1006/jcph.1997.5776

80. St-Germain P, Nistor I, Townsend R, Shibayama T (2014) Smoothed-Particle Hydrodynamics Numerical Modeling of Structures Impacted by Tsunami Bores. J Waterw Port, Coastal, Ocean Eng 140:66–81. https://doi.org/10.1061/(ASCE)WW.1943-5460.0000225

81. Zijlema M, Stelling G, Smit P (2011) SWASH: An operational public domain code for simulating wave fields and rapidly varied flows in coastal waters. Coast Eng 58:992–1012. https://doi.org/10.1016/j.coastaleng.2011.05.015

82. Altomare C, Domínguez JM, Crespo AJC, et al (2015) Hybridization of the Wave Propagation Model SWASH and the Meshfree Particle Method SPH for Real Coastal Applications. Coast Eng J 57:1550024-1-1550024–34. https://doi.org/10.1142/S0578563415500242

83. Engsig-Karup AP, Bingham HB, Lindberg O (2009) An efficient flexible-order model for 3D nonlinear water waves. J Comput Phys 228:2100–2118. https://doi.org/10.1016/j.jcp.2008.11.028

84. Verbrugghe T, Domínguez JM, Crespo AJC, et al (2018) Coupling methodology for smoothed particle hydrodynamics modelling of non-linear wave-structure interactions. Coast Eng 138:184–198. https://doi.org/10.1016/j.coastaleng.2018.04.021

85. Altomare C, Tagliafierro B, Dominguez JM, et al (2018) Improved relaxation zone method in SPH-based model for coastal engineering applications. Appl Ocean Res 81:15–33. https://doi.org/10.1016/j.apor.2018.09.013

86. Verbrugghe T, Stratigaki V, Altomare C, et al (2019) Implementation of Open Boundaries within a Two-Way Coupled SPH Model to Simulate Nonlinear Wave–Structure Interactions. Energies 12:697. https://doi.org/10.3390/en12040697

87. Canelas RB, Crespo AJC, Domínguez JM, et al (2016) SPH–DCDEM model for arbitrary geometries in free surface solid–fluid flows. Comput Phys Commun 202:131–140. https://doi.org/10.1016/j.cpc.2016.01.006

88. Cummins SJ, Cleary PW (2011) Using distributed contacts in DEM. Appl Math Model 35:1904–1914. https://doi.org/10.1016/j.apm.2010.10.019

89. Brilliantov N V., Pöschel T (2001) Granular Gases with Impact-Velocity-Dependent Restitution Coefficient. In: Granular Gases. Springer, pp 100–124

90. Tasora A, Anitescu M (2011) A matrix-free cone complementarity approach for solving large-scale, nonsmooth, rigid body dynamics. Comput Methods Appl Mech Eng 200:439–453. https://doi.org/10.1016/j.cma.2010.06.030

91. Canelas RB, Brito M, Feal OG, et al (2018) Extending DualSPHysics with a Differential Variational Inequality: modeling fluid-mechanism interaction. Appl Ocean Res 76:88–97. https://doi.org/10.1016/j.apor.2018.04.015

92. Hall M (2018) MoorDyn User's Guide

93. Domínguez JM, Crespo AJC, Hall M, et al (2019) SPH simulation of floating structures with moorings. Coast Eng 153:103560. https://doi.org/10.1016/j.coastaleng.2019.103560

94. Viccione G, Bovolin V, Carratelli EP (2008) Defining and optimizing algorithms for neighbouring particle identification in SPH fluid simulations. Int J Numer Methods Fluids 58:625–638. https://doi.org/10.1002/fld.1761

95. Winkler D, Rezavand M, Rauch W (2018) Neighbour lists for smoothed particle hydrodynamics on GPUs. Comput Phys Commun 225:140–148. https://doi.org/10.1016/j.cpc.2017.12.014

96. Crespo AJC, Domínguez JM, Rogers BD, et al (2015) DualSPHysics: Open-source parallel CFD solver based on Smoothed Particle Hydrodynamics (SPH). Comput Phys Commun 187:204–216. https://doi.org/10.1016/j.cpc.2014.10.004

97. Domínguez JM, Crespo AJC, Barreiro A, et al (2014) Efficient implementation of double precision in GPU computing to simulate realistic cases with high resolution. In: Proceedings of the 9th SPHERIC International Workshop. pp 140–145

98. Kleefsman KMT, Fekken G, Veldman AEP, et al (2005) A Volume-of-Fluid based simulation method for wave impact problems. J Comput Phys 206:363–393. https://doi.org/10.1016/j.jcp.2004.12.007

99. Crespo AC, Dominguez JM, Barreiro A, et al (2011) GPUs, a New Tool of Acceleration in CFD: Efficiency and Reliability on Smoothed Particle Hydrodynamics Methods. PLoS One 6:e20685. https://doi.org/10.1371/journal.pone.0020685

100. Valdez-Balderas D, Domínguez JM, Rogers BD, Crespo AJC (2013) Towards accelerating smoothed particle hydrodynamics simulations for free-surface flows on multi-GPU clusters. J Parallel Distrib Comput 73:1483–1493. https://doi.org/10.1016/j.jpdc.2012.07.010

101. Altomare C, Crespo AJC, Domínguez JM, et al (2015) Applicability of Smoothed Particle Hydrodynamics for estimation of sea wave impact on coastal structures. Coast Eng 96:1–12. https://doi.org/10.1016/j.coastaleng.2014.11.001

102. Pringgana G, Cunningham LS, Rogers BD (2016) Modelling of tsunami-induced bore and structure interaction. Proc Inst Civ Eng - Eng Comput Mech 169:109–125. https://doi.org/10.1680/jencm.15.00020

103. González-Cao J, Altomare C, Crespo AJC, et al (2019) On the accuracy of DualSPHysics to assess violent collisions with coastal structures. Comput Fluids 179:604–612. https://doi.org/10.1016/j.compfluid.2018.11.021

104. Rota Roselli RA, Vernengo G, Altomare C, et al (2018) Ensuring numerical stability of wave propagation by tuning model parameters using genetic algorithms and response surface methods. Environ Model Softw 103:62–73. https://doi.org/10.1016/j.envsoft.2018.02.003

105. Lowe RJ, Buckley ML, Altomare C, et al (2019) Numerical simulations of surf zone wave dynamics using Smoothed Particle Hydrodynamics. Ocean Model 144:101481. https://doi.org/10.1016/j.ocemod.2019.101481

106. Subramaniam SP, Scheres B, Schilling M, et al (2019) Influence of Convex and Concave Curvatures in a Coastal Dike Line on Wave Run-up. Water 11:1333. https://doi.org/10.3390/w11071333

107. Altomare C, Tafuni A, Domínguez JM, et al (2020) SPH simulations of real sea waves impacting a large-scale structure. J Mar Sci Eng. https://doi.org/10.3390/jmse8100826

108. Canelas RB, Domínguez JM, Crespo AJC, et al (2015) A Smooth Particle Hydrodynamics discretization for the modelling of free surface flows and rigid body dynamics. Int J Numer Methods Fluids 78:581–593. https://doi.org/10.1002/fld.4031

109. Canelas RB, Domínguez JM, Crespo AJC, et al (2017) Resolved Simulation of a Granular-Fluid Flow with a Coupled SPH-DCDEM Model. J Hydraul Eng 143:06017012. https://doi.org/10.1061/(ASCE)HY.1943-7900.0001331

110. Ropero-Giralda P, Crespo AJC, Tagliafierro B, et al (2020) Efficiency and survivability analysis of a point-absorber wave energy converter using DualSPHysics. Renew Energy. https://doi.org/10.1016/j.renene.2020.10.012

111. Gómez-Gesteira M, González-Cao J, Altomare C, et al (2017) Towards simulating floating offshore oscillating water column converters with Smoothed Particle Hydrodynamics. Coast Eng 126:11–26. https://doi.org/10.1016/j.coastaleng.2017.05.001

112. Brito M, Canelas RB, García-Feal O, et al (2020) A numerical tool for modelling oscillating wave surge converter with nonlinear mechanical constraints. Renew Energy 146:2024–2043. https://doi.org/10.1016/j.renene.2019.08.034

113. González-Cao J, García-Feal O, Domínguez JM, et al (2018) Analysis of the hydrological safety of dams combining two numerical tools: Iber and DualSPHysics. J Hydrodyn 30:87–94. https://doi.org/10.1007/s42241-018-0009-6

114. Novak G, Tafuni A, Domínguez JM, et al (2019) A Numerical Study of Fluid Flow in a Vertical Slot Fishway with the Smoothed Particle Hydrodynamics Method. Water 11:1928. https://doi.org/10.3390/w11091928

115. Nóbrega JD, Matos J, Schulz HE, Canelas RB (2020) Smooth and Stepped Spillway Modeling Using the SPH Method. J Hydraul Eng. https://doi.org/10.1061/(asce)hy.1943-7900.0001776

116. Zubeldia EH, Fourtakas G, Rogers BD, Farias MM (2018) Multi-phase SPH model for simulation of erosion and scouring by means of the shields and Drucker–Prager criteria. Adv Water Resour 117:98–114. https://doi.org/10.1016/j.advwatres.2018.04.011

117. Ren B, He M, Dong P, Wen H (2015) Nonlinear simulations of wave-induced motions of a freely floating body using WCSPH method. Appl Ocean Res 50:1–12. https://doi.org/10.1016/j.apor.2014.12.003

118. Chow AD, Rogers BD, Lind SJ, Stansby PK (2018) Incompressible SPH (ISPH) with fast Poisson solver on a GPU. Comput Phys Commun 226:81–103. https://doi.org/10.1016/j.cpc.2018.01.005

119. Green MD, Peiró J (2018) Long duration SPH simulations of sloshing in tanks with a low fill ratio and high stretching. Comput Fluids 174:179–199. https://doi.org/10.1016/j.compfluid.2018.07.006

120. Leonardi M, Domínguez JM, Rung T (2019) An approximately consistent SPH simulation approach with variable particle resolution for engineering applications. Eng Anal Bound Elem 106:555–570. https://doi.org/10.1016/j.enganabound.2019.06.001

121. Hosain ML, Domínguez JM, Bel Fdhila R, Kyprianidis K (2019) Smoothed particle hydrodynamics modeling of industrial processes involving heat transfer. Appl Energy 252:113441. https://doi.org/10.1016/j.apenergy.2019.113441

122. Mayoral-Villa E, Alvarado-Rodríguez CE, Klapp J, et al (2016) Smoothed particle hydrodynamics: Applications to migration of radionuclides in confined aqueous systems. J Contam Hydrol 187:65–78. https://doi.org/10.1016/j.jconhyd.2016.01.008

123. Adami S, Hu XY, Adams NA (2012) A generalized wall boundary condition for smoothed particle hydrodynamics. J Comput Phys 231:7057–7075. https://doi.org/10.1016/j.jcp.2012.05.005