SPHERA v.10.0.0 (RSE SpA): documentation file

1. Introduction ... 6
1.1. Description and references ... 6
1.2. Warranties and responsibilities ... 7
1.3. Citation of SPHERA v.10.0.0 (RSE SpA) ... 7
1.4. SPHERA developers/authors ... 7
1.5. SPHERA official users ... 7
1.6. SPHERA acknowledgments ... 8
1.7. SPHERA (RSE SpA) registration ... 9
2. SPH-ALE$_3$-LC$_{m,p}$C$_0$ scheme (0th-order consistency) ... 9
2.1. Smoothed Particle Hydrodynamics (SPH) ... 10
2.2. SPH-ALE$_3$-LC$_{m,p}$C$_0$ balance equations ... 11
2.2.1. Local-form ALE balance equations ... 11
2.2.2. SPH-ALE$_0$-LC$_m$C$_0$ intermediate formulation ... 12
2.2.3. SPH-ALE$_{1u}$-LC$_{m,p}$C$_0$ intermediate formulation ... 12
2.2.4. SPH-ALE$_1$-LC$_/$C$_0$ intermediate formulation ... 14
2.2.5. SPH-ALE$_2$-LC$_m$C$_0$ intermediate formulation ... 15
2.2.6. SPH-ALE$_3$-LC$_{m,p}$C$_0$ formulation ... 16
2.3. SPH-ALE$_3$-LCC boundary terms ... 17
2.3.1. Mass-flux boundary terms in the Continuity Equation ... 17
2.3.2. Boundary contributions to the velocity divergence ... 18
2.4. SPH-ALE$_3$-LCC conservative time integration ... 19
2.5. Volume conservation and mass-variable approach in SPH-ALE$_3$-LCC ... 20
2.6. Comparisons with some state-of-the-art formulations ... 21
2.7. Viscous inner term in the Momentum Equation ... 22
2.8. Equation of State and its inverse function ... 23
2.9. Sub-particle term of the Continuity Equation: partial pressure smoothing ... 23
2.10. Sub-particle term of the Momentum Equation ... 23
2.11. General relationships for the kernel gradient ... 24
2.12. Kernel functions and gradients ... 24
2.13. The boundary velocity ... 25
2.14. Raw, C$_0$ and C1 consistent SPH approximations of the pressure-gradient with SPHERA boundary terms ... 25
2.15. Raw, C$_0$ and C1 SPH approximations of the ALE velocity gradient and divergence with SPHERA boundary terms ... 26
2.16. Dirichlet’s boundary conditions for the water depth ... 27
2.17. Inlet sections for Bazin’s sharp-crested weirs ... 28
2.18. Stability criteria ... 29
2.19. SPH monitors ... 29
3. SPH-ALE$_3$-LC$_m$C$_1$ scheme (1st-order consistency) ... 30
3.1. A C$_1$ consistency procedure for linear convolution methods with any boundary scheme and application to SPH ... 30
3.2. A SPH renormalization matrix with boundary terms ... 32
3.3. SPH-ALE$_3$-LC$_m$C$_1$ balance equations ... 33
3.4. Free-surface boundary conditions and negative pressure ... 34
4. SPH non-ALE scheme for dense granular flows and mono-phase flows (no consistency) ... 35
4.1. Mixture model for dense granular flows ... 35
4.2. 2-interface 3D erosion criterion ... 40
4.3. A simplified approach for soil liquefaction ... 43
4.4. Mono-phase formulation ... 43

4.5. Time integration schemes for the non-consistent SPH formulation 43
5. Boundary treatment scheme for fixed surface walls: the Semi-Analytical approach (SASPH).... 44
5.1. The Semi-Analytical approach.. 44
5.2. SASPH boundary contribution to the pressure-gradient term in the Momentum Equation... 45
5.3. SASPH boundary contribution to the velocity-divergence term in the Continuity Equation. 47
5.4. SASPH elastic-reaction boundary term... 48
5.5. SASPH boundary contribution to the viscous term of the Momentum Equation................. 48
5.6. SASPH boundary contribution to the sub-particle term of the Momentum Equation 48
5.7. Wall drag and wall-function similarity theory: adaptation to flash floods........................... 48
5.7.1. Wall shear stress 48
5.7.2. Turbulent viscosity for the Neutral Surface Boundary Layer 49
5.7.3. Slip coefficient for the Neutral Surface Boundary Layer 49
5.7.4. Roughness characteristic length 50
5.7.5. Roughness length for flash floods: CORINE Land Cover, DEM and DTM 51
6. Boundary treatment scheme for the transport of solid bodies 53
6.1. SPH balance equations for the transport of rigid solid bodies... 53
6.2. Improving 3D rotations 55
6.3. Reaction forces... 57
6.4. Solid-solid normal reactions under impingement 58
6.5. Normal restitution coefficient.. 60
6.6. Sliding friction force.. 60
6.6.1. Aerial stage (non-negative value for the input friction angle and body-frontier interactions) 60
6.6.2. Aerial stage (negative value for the input friction angle or body-body interactions)......... 61
6.6.3. Submerged stage 61
6.7. Body-boundary normal reaction force under sliding (at null normal velocity) 61
6.7.1. Aerial stage.. 62
6.7.2. Submerged stage 62
6.8. Fluid-body coupling terms.. 62
6.9. Recent improvements to the scheme for body transport 65
6.10. Improving initial conditions in the presence of solid bodies.. 67
7. DB-SPH boundary treatment scheme for mobile surface walls 67
7.1. DB-SPH particle approximation and modifications of the balance equations 67
7.2. 1D Linearized Partial Riemann Solver.. 69
7.3. Semi-particle volume... 70
7.4. DB-SPH inlet and outlet sections.. 70
7.5. Shear stress boundary terms for mobile wall frontiers... 71
8. The substation-flooding damage scheme 71
8.1. Mathematical models... 72
8.1.1. Proxy damage and vulnerability to flood-induced out-of-service events 72
8.1.2. Flood-induced damage to the components of the electrical substations........................... 73
8.2. The numerical model... 73
9. Installation, input-file template and tutorials.. 74
9.1. Installation 74
9.2. Commented templates of the input files of SPHERA v.10.0.0... 75
9.3. Tutorials... 75
9.3.1. “body_body_impacts”... 75
9.3.2. “body_boundary_impacts”.. 75
9.3.3. “dam_breach_ICOLD_trunks” 75
9.3.4. “db_2bodies” 75
9.3.5. “db_2D_demo” 75

9.3.6. “db_Alpe_Gera” 75
9.3.7. “db_Alpe_Gera_Lanzada_substations” 76
9.3.8. “db_body_exp_UniBas” 76
9.3.9. “db_ICOLD” 76
9.3.10. “db_multi_body” 76
9.3.11. “db_squat_obstacle” 76
9.3.12. “db_tall_obstacle” 76
9.3.13. “dike_breach_2D_expSchHag12JHR_ID22” 76
9.3.14. “edb_2D_demo” 76
9.3.15. “edb_2D_FraCap02” 76
9.3.16. “edb_2D_FraCap02_Taipei” 76
9.3.17. “edb_2D_Spi05” 76
9.3.18. “edb_ICOLD” 76
9.3.19. “edb_KarlSand” 77
9.3.20. “edb_Pon10” 77
9.3.21. “floating_cube_stability” 77
9.3.22. “flushing_2D” 77
9.3.23. “flushing_3D” 77
9.3.24. “jet_plate” 77
9.3.25. “rectangular_side_weir_Fr_0_491” 77
9.3.26. “San_Fernando_Lower_van_Norman_dam_liquefaction” 77
9.3.27. “sloshing_tank_TbyTn_0_78” 77
9.3.28. “sloshing_tank_TbyTn_1_07” 77
9.3.29. “spherical_Couette_flows” 78
9.3.30. “SPH_udb_exp_Kim2015HYDROL” 78
9.3.31. “submerged_landslide” 78
9.3.32. “still_water_tank” 78
9.3.33. “wave_motion_for_WaveSAX” 78
9.3.34. “wedges_falls_on_still_water” 78
9.3.35. “2D_Vajont_experiment” 78
9.3.36. “spillway_withpiers” 78
9.3.37. “tsunami_benchmark1_ISEC” 78
9.3.38. “dyke_failure_Kagerplassen_2015” 78
9.3.39. “stream_flat_bottom” 78
9.3.40. “douf_Vajont” 78
9.3.41. “uf_es_Calenzano” 78
9.3.42. “dam_break_3D_demo” 78
9.3.43. “Couette_sf” 78
9.3.44. “water_shipping” 78
9.3.45. “WSI_WAVESAX” 79
10. Overview of the program units 79
10.1. Folders of the program units 79
10.2. Program units for Generic Boundary Conditions 80
10.3. Program units for Mass-related balance equations 81
10.4. Program units for Momentum Equation 81
10.5. Program units for the Solid Momentum Equation 82
10.6. Program units for the contributions of solid bodies to the Mass-related balance equations 82
10.7. Program units for the contributions of solid bodies to the Momentum Equation 82
10.8. Program units for Initial conditions for solid bodies 82
10.9. Program units for SPH neighbouring search for body particles 83

10.10. Program units for post-processing for solid bodies ... 83
10.11. Program units for Time integration for solid bodies ... 83
10.12. Program units for CLC land-use classes ... 83
10.13. Program units for the Scheme for mobile surface walls ... 83
10.14. Program units for the Equation of State ... 85
10.15. Program units for the Erosion criterion ... 85
10.16. Program units for the geometry procedures ... 85
10.17. Program units for the Initial Conditions ... 87
10.18. Program units for the Scheme for dense granular flows ... 87
10.19. Program units for the Main algorithm ... 87
10.20. Program units for the Memory Input/Output management ... 89
10.21. Program units for the Allocations/deallocations of derived types ... 89
10.22. Program units for the Allocations/deallocations of preset types ... 90
10.23. Program units for Fortran modules ... 90
10.24. Program units for the SPH neighbouring search ... 90
10.25. Program units for Post-processing ... 91
10.26. Program units for Post-processing the 2D synthetic fields ... 92
10.27. Program units for Post-processing the elapsed time ... 92
10.28. Program units for Post-processing at SPH monitors ... 92
10.29. Program units for Pre-processing ... 93
10.30. Program units for Reading the input files ... 93
10.31. Program units for the SASPH contributions to the Mass-related balance equations ... 94
10.32. Program units for the SASPH contributions to the Momentum Equation ... 94
10.33. Program units for the Initial Conditions for the fixed surface walls ... 95
10.34. Program units for the assessment of the SASPH integrals at the initial time ... 95
10.35. Program units for the interpolations of SASPH integrals during the time steps ... 96
10.36. Program units for the SPH neighbouring search for fixed surface walls ... 96
10.37. Program units for the contribution to the stability criteria from fixed surface walls ... 96
10.38. Program units for the wall function for fixed surface walls ... 97
10.39. Program units for the string management ... 97
10.40. Program units for Time integration ... 97
10.41. Style formatting ... 97
11. The numerical chain of SPHERA ... 98
11.1. Engauge Digitizer ... 99
11.2. Site overview ... 99
11.3. The SRTM3 dataset and further DEM-DTM datasets ... 99
11.4. GDAL ... 99
11.5. DEM2xyz: preliminary elaboration of the raw DEM and DTM files ... 100
11.6. Paraview: post-processing of the preliminary DEM and DTM files ... 100
11.7. Elaboration of a unique DEM-DTM integrated surface ... 101
11.8. DTM/DEM dataset merging ... 103
11.9. DEM2xyz: further modifications to the DEM-DTM improvements ... 103
11.10. Paraview: post-processing of the final output of DEM2xyz ... 103
11.10.1. Simple use procedure to design a weir ... 103
11.11. Modification of the spatial resolution of the DEM-DTM ... 104
11.12. SnappyHexMesh ... 104
11.13. ply2SPHERA_perimeter ... 104
11.14. Paraview: procedures to pre-process the CORINE Land Cover (CLC) maps ... 105
11.15. SPHERA ... 106
11.16. Gnuplot ... 106
11.17. GSView ... 106

11.18. Grid Interpolator: post-processing of the 2D output fields of SPHERA 106
11.19. Paraview: post-processing and visualization of the 2D and 3D output fields of SPHERA 107
11.20. Image Magick 107
11.21. Virtual Dub 107
12. GNU Free Documentation License 107
References 116

This documentation file is intended to provide only additional and updated material, beyond the other SPHERA GitHub repository files and the associated papers on International Journals (Sec.1).

1. INTRODUCTION

1.1. Description and references

SPHERA (RSE SpA, 2022, [1]) is a 3D research FOSS code based on the Computational Fluid Dynamics (CFD) - Smoothed Particle Hydrodynamics (SPH) method. SPHERA application fields cover: subcatchment flash floods and fast landslides for the safety of hydroelectric power plants, electrical substations and electricity pylons; water waves and marine energy. Its numerical schemes published on International peer-reviewed Journals (index by Scopus and Web of Science) are synthesized as follows: a scheme for dense granular flows (Amicarelli et al., 2017, [2]), which can be computationally accelerated by a numerical parameter called limiting viscosity (Manenti et al., 2018, [3]); a scheme for the transport of solid bodies (Amicarelli et al., 2015, [4]) with its recent updates (Amicarelli et al., 2020, [5]; RSE code improvements reported in Sec.5 of Paggi et al., 2021, [6]; Amicarelli et al., 2022, [7]); two alternative boundary treatment schemes for fixed surface walls (Di Monaco et al., 2011, [8]) and mobile surface walls (Amicarelli et al., 2013, [9]; Amicarelli et al., 2022 [7]); a scheme for the flooding damage to electrical substations (Amicarelli et al., 2021, [10]); a 2D erosion criterion (Manenti et al., 2012, [11]), which can speed up the scheme for dense granular flows when the Shields-like erosion mechanism is the only cause of mobilization of the solid grains.
The Continuity Equation and the Momentum Equation for the mixture of solid granular material and a generic fluid is derived in Amicarelli et al. (2017, [2]), with the introduction of the mean effective stress, the pore-pressure and several mixture quantities (velocity, density, viscosity and stresses). Approximate sub-particle terms are also adopted. Following the Weakly-Compressible (WC) approach (Monaghan, 2005, [12]), often used by CFD methods, the Equation of State is linear, barotropic, and dependent on the artificial bulk modulus. The simulated mixture degenerates into a Newtonian liquid in case of null volume fraction of the solid phase. The balance equations for the linear and angular momenta of the solid bodies are expressed by the Newton-Euler equations for the 3D dynamics of rigid bodies. They are coupled with the mixture/fluid balance equations mentioned above and involve the hydrodynamic thrust and the reaction forces for body-boundary and body-body interactions included impingements, sliding friction, anti-pitch and anti-roll reactions. The released version v.9.0.0 was presented in Amicarelli et al. (2020, [5]).
Other two numerical schemes available in SPHERA v.10.0.0 are to be submitted. It is two Local-form 3-stage Arbitrary-Lagrangian-Eulerian Conservative-Consistent SPH schemes: SPH-ALE$_3$-LC$_{m,p}$-C$_0$ (Sec.2) and SPH-ALE$_3$-LC$_m$C$_1$ (Sec.3). Some improvements to the boundary scheme for fixed surface walls are also at the submission stage: they concern an adaptation to flash floods for wall drag and wall-function similarity theory (Sec.5.7). Due to Copyright constraints with journals, the sections associated with preprints are explicitly declared at their beginning.
With Copyright 2005-2022 (RSE SpA - formerly ERSE SpA, formerly CESI RICERCA, formerly CESI-Ricerca di Sistema -), SPHERA has been developed for RSE SpA (hereafter RSE, unique owner of the patrimonial rights of SPHERA) by the following authors (SPHERA author list): Andrea Amicarelli, Antonio Di Monaco, Sauro Manenti, Elia Giuseppe Bon, Daria Gatti, Giordano Agate, Stefano Falappi, Barbara Flamini, Roberto Guandalini, David Zuccalà, Emanuela Abbate, Qiao Cheng. Since its SPHERA v.7.0 branches SPHERA has being developed under a Git repository (GitHub web site). Its current version contains the folders of Table 1.1.
SPHERA is free software released under the GNU General Public License (Free Software Foundation). The email address to contact the first author of SPHERA is: andrea.amicarelli@rse-web.it

Folder	Description
(main folder)	License file (GNU-GPL license)
doc	Present documentation file
src	SPHERA source code (with Makefile)
exe	Folder where the executables are generated
input	Commented templates for the input files; tutorials

Table 1.1 Folders in SPHERA v.10.0.0 repository.

1.2. Warranties and responsibilities

SPHERA v.10.0.0 is released “as is” with no warranty. NEITHER RSE SPA, NOR ANY OF ITS REPRESENTATIVES (OR ANY CODE AUTHOR) MAKE ANY WARRANTY, EXPRESS OR IMPLIED, OR ASSUMES ANY LEGAL LIABILITY OR RESPONSIBILITY FOR THE ACCURACY, COMPLETENESS, EFFECTIVENESS, INTEGRITY, AVAILABILITY, OR USEFULNESS OF THE SOFTWARE, ANY INFORMATION PERTAINING TO THE SOFTWARE, OR REPRESENTS THAT ITS USE WOULD NOT INFRINGE PRIVATELY OWNED RIGHTS. No support service (for the code installation, use, teaching activities, …) is implied by or included in the software license. ”

1.3. Citation of SPHERA v.10.0.0 (RSE SpA)

All the published and unpublished items/products/documents of every kind (i.e. results, publications, software, projects, web pages, press and digital documents, teaching or technological devices, reports, dissemination tools/devices,…) related to SPHERA v.10.0.0 need the following citation: “SPHERA v.10.0.0 (RSE SpA)”.
Further proper citations may refer to SPHERA-related papers on International Peer-Reviewed Journals indexed by Scopus and Web of Science (Sec.1).
SPHERA should also be cited in all the related publications, reports and dissemination tools and media (also included press and digital products), by means of the following citation:
“SPHERA v.10.0.0 is realised by RSE SpA thanks to the funding “Fondo di Ricerca per il Sistema Elettrico” within the frame of a Program Agreement between RSE SpA and the Italian Ministry of Economic Development (Ministero dello Sviluppo Economico).”

1.4. SPHERA developers/authors

This section reports few and non-exhaustive notes, which may help potential authors of SPHERA or its derived codes.
If one receives a code with the GNU-GPL license, then she/he has to transmit the license rights unchanged. In particular, it can be useful to remind that GNU-GPL is viral. This also implies that a code, which contains just very few lines of a GNU-GPL code, becomes necessarily a GNU-GPL code in its entirety, when integrating those lines of a GNU-GPL code.
Every code derived from a GNU-GPL code must explicitly report every modification with respect to the original GNU-GPL code.

1.5. SPHERA official users

The information reported in this section only has an educational aim and does not modify the terms and conditions of SPHERA license.
SPHERA is available at github.com. Potential developers or users may:

1) contribute to the development of SPHERA as code authors by means of a free github account (basic knowledge of Git is mandatory);
2) contribute to the validation of SPHERA as “official users” by means of a free github account (basic knowledge of Git is not mandatory);
3) use SPHERA independently with possible minor modifications to the code, respecting the code name, license, citations and Copyright;
4) independently introduce relevant modifications to SPHERA, thus obtaining a FOSS derived code (which has a different name from SPHERA but has to cite SPHERA as the original code, mentioning all the parts of the original code, its Copyright and license) and redistribute it (according to the GNU-GPL license and SPHERA citation terms and Copyright), or propose such modifications to RSE for their integration in SPHERA;
5) to propose to RSE some program units not belonging to SPHERA and developed with independent funds: they can be released with GNU-GPL license and integrated in SPHERA;

6) to propose to RSE some program units of a code developed with independent funds: they can be released under the GNU-LGPL license and integrated in SPHERA, without constraints for the authors to make their original code a FOSS in its entirety.

The modifications of the source code and the new input files produced with independent funds by non-RSE authors could be proposed to RSE (with a non-RSE Copyright) to be integrated in SPHERA as FOSS program units and input files. In case of acceptance, these contributions will be kept updated by RSE in the following code versions, until RSE will consider them useful for SPHERA development and validation (RSE can delete program units since a certain version).

SPHERA authors and "official users" need to activate a free account on github.com (with recognizable name, surname and affiliation) and "fork" SPHERA, by clicking on the icon "fork" in the official SPHERA (2022, [1]) public repository.

The basic knowledge of Git is mandatory only for SPHERA authors. In this context, the following links may be useful:

- https://git-scm.com/
- https://www.youtube.com/watch?v=U8GBXvdmHT4&index=3&list=PLg7s6cbtAD15Das5LK9mXt_g59DLWxKUe

Anyone could be informed on the real-time code upgrades by means of automatic emails sent by github.com. This free service is available by activating a free account at https://github.com/join and then clicking on the icon "watch" in the official public repository of SPHERA. When activating a GitHub account, it could be convenient to choose a username, which included name, surname and affiliation. This will permit to get recognized and attend to SPHERA development/validation (one notices that the symbol "." is not permitted within a github username).

If any activity dedicated to SPHERA is declared to RSE, that activity might be reserved by RSE for a specific user during a specific period to avoid redundancy and conflicts between users.

Finally, SPHERA is indexed in the list of SPH codes of SPHERIC (2021, [13]).

The translation of a code in another programming language does not imply any Copyright change.

Any code (included the Open-Source codes) has its own Copyright which is transmitted to any derived software.

1.6. SPHERA acknowledgments

SPHERA v.10.0.0 (RSE SpA) is realised thanks to the funding "Fondo di Ricerca per il Sistema Elettrico" within the frame of a Program Agreement between RSE SpA and the Italian Ministry of Economic Development (Ministero dello Sviluppo Economico).

SPHERA has been financed by the Research Fund for the Italian Electrical System (for "Ricerca di Sistema -RdS-"), at different stages:

- ✓ under the second period of RdS (2003-2005), where CESI SpA was the only beneficiary of the Research Fund for the Italian Electrical System;
- ✓ under the Contract Agreement between CESI Ricerca SpA and the Italian Ministry of Economic Development for the of RdS period 2006-2008, in compliance with the Decree of 8 March 2006;
- ✓ under the Contract Agreement between ERSE and the Ministry of Economic Development-General Directorate for Energy and Mining Resources (for the of RdS period 2009-2011) stipulated on 29 July 2009 in compliance with the Decree of 19 March 2009;
- ✓ under the Contract Agreement between RSE SpA and the Italian Ministry of Economic Development for the of RdS period 2012-2014, in compliance with the Decree of November 9, 2012 (Project Manager Antonella Frigerio);
- ✓ under the Contract Agreement between RSE SpA and the Italian Ministry of Economic Development for the RdS period 2015-2017, in compliance with the Decree of April 21, 2016. Reference project: 'A.5 - Sicurezza e vulnerabilità del sistema elettrico', Project Manager Antonella Frigerio.

- ✓ under the Contract Agreement between RSE S.p.A. and the Ministry of Economic Development - General Directorate for the Electricity Market, Renewable Energy and Energy Efficiency, Nuclear Energy in compliance with the Decree of April 16, 2018; Project: "2.5 Modelli e strumenti di intervento, anche preventivo, per la difesa e il miglioramento della sicurezza e della resilienza delle reti" - Ricerca di Sistema (2.5 Models and action tools for the safety and resilience of the power grids - Research on the Italian Energy System); Project Manager: Francesco Apadula (formerly Antonella Frigerio); Agreement between the Italian Ministry of Economic Development and RSE SpA 2019-2021;
- ✓ by the Research Fund for the Italian Electrical System under the Contract Agreement between RSE S.p.A. and the Ministry of Economic Development - General Directorate for the Electricity Market, Renewable Energy and Energy Efficiency, Nuclear Energy in compliance with the Decree of April 16th, 2018. Project Managers: Stefano Maran, Giovanni Pirovano.

"We acknowledge the CINECA award under the ISCRA initiative, for the availability of High-Performance Computing resources and support": SPHERA tutorials have been executed using the HPC resources of the following instrumental-funding HPC projects: HSPHER22b, HSPHER22, HSPHER21, HSPHER20, HPCEFM7b, HPCEFM17, NMTFEPRA, HPCEFM16, HPCEFM15, HSPHMI14. TNMRA01, HPCEFM18, HSPHERA9, HSPHCS9, HPCNHLW1, HSPHER9b, HPCNHLW2.

The FOSS versions of SPHERA have been supported in RSE and promoted by: the Department Director Michel de Nigris; the Deputy Director, Project Manager and Research Team Manager Antonella Frigerio; the Project Manager and Research Team Manager Francesco Apadula (2020-2022); the Project Managers Stefano Maran and Giovanni Pirovano (2022); the Research Team Managers Guido Pirovano (2016-2020) and Massimo Meghella (2015-2016).

1.7. SPHERA (RSE SpA) registration

SPHERA v.8.0 Copyright is registered ("Registro pubblico speciale per i programmi per elaboratore, SIAE", Italy). Since SPHERA v.8.0, the code modifications are available on public github repository.

2. SPH-ALE$_3$-LC$_{M,P}$C$_0$ SCHEME (0TH-ORDER CONSISTENCY)

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

The traditional SPH method represents a fluid domain as a partition of SPH particles, each one initially equal to a fluid volume. It is a purely Lagrangian approach. However, as for any numerical method, truncation errors are introduced and propagate so that the SPH quantities and trajectories are not equal to the those of the fluid volumes represented at the initial time. The current formulation considers an alternative set of numerical control volumes. The present SPH particles do not aim to mimic fluid volumes, but to represent a proper set of mobile computational nodes for the description of the fluid flow. They move along trajectories different from the fluid volumes and can slightly exchange mass with the neighbouring particles. The following properties are set by construction:

a) compliance with the ALE balance equations and boundary conditions (ALE-balance);
b) reduction of the SPH truncation error ε_p (consistency);
c) reduction of ε_p propagation in space and time (stability);
d) conservation of the internal global fluid mass, net of boundary fluxes (mass conservation);
e) conservation of the internal global fluid momentum, net of external forces (momentum conservation).

Two alternative formulations are proposed:

1) a C$_0$-consistent 3-stage Local-form SPH-ALE scheme whose SPH trajectories describe particle distributions closer to a regular and isotropic Cartesian distribution, condition which reduces the SPH truncation error (Amicarelli et al., 2011, [15]), and whose ALE terms are derived to conserve mass and momentum (Sec.2);
2) a C$_1$-consistency method, fully involving the boundary terms, which replaces the C$_0$ formulation when C$_1$-consistency is more effective than momentum conservation (Sec.3).

The combined use of several approaches to reduce the truncation error is also motivated as follows. A consistency method itself, even of relatively high order, does not guarantee the minimization of the truncation error of any numerical method: the fields to be represented are in general more complex than polynomials of a certain degree, especially under large deformations, where SPH is more commonly used.
ALE velocities are defined to conserve mass and momentum and to give stability to the model. If negligible, some ALE balancing terms are neglected, but the terms involved in the exact conservation of mass and momentum. The SPH particle velocity equals at each point of its trajectory the velocity of the local fluid volume which is temporarily intercepted. This condition makes the control-volume velocity to be representative of the local fluid velocity. The formulation satisfies the Continuity Equation so that the density of a SPH particle is representative of the local density of the fluid. Under these conditions, the conservation of the initial internal global mass of the SPH particles guarantees the conservation of the fluid mass. The SPH mass conservation is imposed at any "pair-particle sub-system", i.e., within each mass flux exchanged by a couple of interacting particles. The velocity increments of the new particles, whose time integration permits to describe different trajectories from the fluid ones, are locally so slight that the SPH particle velocity $\underline{u}_m$ (m/s) is always and everywhere representative of the local fluid velocity. Under this condition, the conservation of the internal global momentum of the SPH particles, depending on $\underline{u}_m$ values, guarantees the conservation of the internal global fluid momentum. The SPH momentum conservation is imposed to any pair-particle sub-system.

2.1. Smoothed Particle Hydrodynamics (SPH)

Smoothed Particle Hydrodynamics (SPH) represents a mesh-less CFD technique, whose computational nodes are represented by numerical fluid particles.
In the continuum, the functions and derivatives in the fluid dynamics balance equations are approximated by convolution integrals, which are weighted by interpolating (or smoothing functions), called kernel functions.
The integral SPH approximation ($<>_I$) of a generic function (f) is defined as:

$$\langle f \rangle_{I,\underline{x}_0} \equiv \int_{V_h} fW dx^3 \tag{2.1}$$

where W (m^{-3}) is the kernel function, $\underline{x}_0$ (m) the position of a generic computational point and V_h (m^3) the integration volume, which is called kernel support. This is represented by a sphere of radius $2h$ (m), possibly truncated by the frontiers of the fluid domain.
Any first derivative of a generic function, calculated along i-axis, can be computed as in (2.1), after replacing f with the targeted derivative. After integration by parts, one obtains:

$$\left\langle \left. \frac{\partial f}{\partial x_i} \right| \right\rangle_{I,\underline{x}_0} = \int_{A_h} fWn_i dx^2 - \int_{V_h} f \frac{\partial W}{\partial x_i} dx^3 \tag{2.2}$$

The integration also involves the surface A_h of the kernel support. The associated surface integral is non-zero in case of a truncated kernel support. The representation of this term noticeably differentiates SPH codes (Adami et al., 2012, [16]; Hashemi et al., 2012, [17]; Macia et al., 2012, [18]; Mayrhofer et al., 2013, [19]; Ferrand et al., 2013, [20]).
Far from boundaries, the SPH particle approximation of (2.1) reads:

$$\left\langle \left. \frac{\partial f}{\partial x_i} \right| \right\rangle_{\underline{x}_0} = -\sum_b f_b \left. \frac{\partial W}{\partial x_i} \right|_b \omega_b \tag{2.3}$$

where a summation on particle volumes ω (m^3) replaces the volume integral. The subscripts "$_0$" and "$_b$" refer to the computational particle and its "neighbouring particles" (fluid particles within the kernel support of the computational particle), respectively.

Usually, the approximation (2.3) is replaced by more complicated and accurate formulas. Further, the SPH technique can also approximate a generic n-th derivative, following the same approach of the cited equation. Several state-of-the-art papers investigated the advantages and shortcomings of SPH modelling: Gomez-Gesteira et al. (2010, [21]); Marongiu et al. (2010, [22]); Price (2012, [23]); Le Touzé et al. (2013, [24]); Violeau & Rogers (2016, [25]); Gotoh & Khayyer (2018, [26]); Saikali et al. (2020, [27]).

2.2. SPH-ALE$_3$-LC$_{m,p}$C$_0$ balance equations

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
The governing equations of the first SPH formulation presented in this study is derived starting from the Local-form ALE balance equations (Sec.2.2.1) and developing step-by-step the intermediate formulations (Sec.2.2.2-Sec.2.2.5) which permitted to obtain the final SPH-ALE$_3$-LC$_{m,p}$C$_0$ balance equations (Sec.2.2.6).

2.2.1. Local-form ALE balance equations

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
The derivative of a function *f* along the arbitrary trajectory of a non-material computational node is reported in Sun et al. (2019, [28]), among the others, where $\delta \underline{u}$ (m/s) is the vector difference between the velocity of the computational node $\underline{u}_m$ (m/s) and the local fluid velocity $\underline{u}$ (m/s):

$$\left.\frac{df}{dt}\right|_{\underline{u}_m} = \frac{\partial f}{\partial t} + \underline{\nabla} f \cdot \underline{u}_m = \frac{df}{dt} + \underline{\nabla} f \cdot \delta \underline{u}, \quad \underline{u}_m \equiv \underline{u} + \delta \underline{u} \tag{2.4}$$

This expression is associated with the Arbitary Lagrangian-Eulerian (ALE) approach (e.g., Marongiu et al., 2010, [22]) whose balance equations are written for control volumes or computational nodes which can move with a velocity equal to the fluid velocity (Lagrangian approach), a null velocity (Eulerian approach) or any arbitrary velocity (ALE approach). Hereafter (2.4) will be cited as the ALE derivative and $\delta \underline{u}$ as the ALE velocity increment, for sake of simplicity. When the generic function is a vector, one obtains:

$$\left.\frac{d\underline{f}}{dt}\right|_{\underline{u}_m} = \frac{\partial \underline{f}}{\partial t} + \left[\underline{\nabla} \otimes \underline{f}\right]^T \delta \underline{u} \tag{2.5}$$

The ALE term of the ALE derivative of a scalar can be rearranged as follows (Sun et al., 2019, [28]):

$$\underline{\nabla} f \cdot \delta \underline{u} = \underline{\nabla} \cdot \left(f \delta \underline{u} \right) - f \left(\underline{\nabla} \cdot \delta \underline{u} \right) \tag{2.6}$$

This form is useful to derive the ALE Continuity Equation. In case of vector functions, the analogous relationship for the ALE term reads:

$$\begin{gathered} \frac{\partial f_i}{\partial x_j} \delta u_j = \frac{\partial}{\partial x_j}\left(f_i \delta u_j \right) - f_i \frac{\partial}{\partial x_j}\left(\delta u_j \right) \\ \left(\underline{\nabla} \otimes \underline{f} \right)^T \delta \underline{u} = \left(\underline{f} \otimes \delta \underline{u} \right) \underline{\nabla} - \underline{f} \left(\underline{\nabla} \cdot \delta \underline{u} \right), \quad \left(\underline{f} \otimes \delta \underline{u} \right) \underline{\nabla} \equiv \left[\underline{\nabla}^T \left(\underline{f} \otimes \delta \underline{u} \right) \right]^T \end{gathered} \tag{2.7}$$

Considering the above equations, the local-form ALE Continuity Equation is alternatively expressed by the following relationships:

$$\left.\frac{d\rho}{dt}\right|_{\underline{u}_m} = -\rho \underline{\nabla} \cdot \underline{u}_m + \underline{\nabla} \cdot \left(\rho \delta \underline{u} \right) \Leftrightarrow \left.\frac{d\rho}{dt}\right|_{\underline{u}_m} = -\rho \underline{\nabla} \cdot \underline{u} + \delta \underline{u} \cdot \underline{\nabla} \rho \tag{2.8}$$

where ρ is density (kg/m^3). The first expression is free from the fluid velocity and the latter shows that the ALE term is non-negligible, even if $\delta \underline{u}$ is locally much smaller than the fluid velocity.
Analogously, the Local/strong-form of the ALE Momentum Equation can be obtained:

$$\left.\frac{du_i}{dt}\right|_{\underline{u}_m} = -\delta_{i3} g - \frac{1}{\rho} \frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j^2} + \frac{\partial u_i}{\partial x_j} \delta u_j \tag{2.9}$$

$$\left.\frac{d\underline{u}}{dt}\right|_{\underline{u}_m} = \underline{g} - \frac{1}{\rho}\underline{\nabla}p + \nu\nabla^2\underline{u} + \left[\underline{\nabla}\otimes\underline{u}\right]^T\delta\underline{u}$$

where ν (m^2/s) is the kinematic viscosity and $\underline{g}$ (m/s^2) is the gravity acceleration. Sun et al. (2019, [28]) wrote the ALE term in the Momentum Equation as a sum of two terms and numerically observed that they are negligible, provided that $\delta\underline{u} << \underline{u}$. The local-form ALE balance equations can be also directly derived from the Reynolds' transport theorem. Under this frame, the ALE balance equation for the volume ω (m^3) of a SPH particle (i.e., a numerical control volume) can be derived (Marongiu et al., 2010, [22]):

$$\left.\frac{d}{dt}\right|_{\underline{u}_m}\int_V dV = \int_V\left(\underline{\nabla}\cdot\underline{u}_m\right)dV, \quad dV\to 0 \Rightarrow \left.\frac{d\omega}{dt}\right|_{\underline{u}_m} = \omega\underline{\nabla}\cdot\underline{u}_m \tag{2.10}$$

The ALE mass equation is obtained combining the ALE density and volume equations:

$$\left.\frac{dm}{dt}\right|_{\underline{u}_m} = \left.\frac{d\left(\rho\omega\right)}{dt}\right|_{\underline{u}_m} = \rho\left.\frac{d\omega}{dt}\right|_{\underline{u}_m} + \omega\left.\frac{d\rho}{dt}\right|_{\underline{u}_m} = \omega\underline{\nabla}\cdot\left(\rho\delta\underline{u}\right) \tag{2.11}$$

where *m* (kg) is mass. The ALE SPH particles are numerical mass-variable control-volumes.

2.2.2. SPH-ALE$_0$-LC$_m$C$_0$ intermediate formulation

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

The starting formulation of this section is unstable, C$_0$ and mass-conservative (C$_m$). It has no ALE velocity increment $\left(\delta\underline{u} = \underline{0}\right)$. The Continuity Equation of this SPH-ALE$_0$-LC$_m$C$_0$ formulation reads:

$$\left.\frac{d\rho}{dt}\right|_{\underline{u}_0} \equiv \frac{d\rho_0}{dt} = -\rho_0\left\langle\underline{\nabla}\cdot\underline{u}\right\rangle_{C0,0} + \Pi_\rho, \quad \left\langle\underline{\nabla}\cdot\underline{u}\right\rangle_{C0,0,IN} = -\sum_b\left(\underline{u}_b - \underline{u}_0\right)\cdot\underline{\nabla}W_b\omega_b \tag{2.12}$$

where W (m^{-3}) is the kernel function and the subscripts "$_0$" and "$_b$" indicate a SPH computational particle and its neighbouring particles in the kernel support, respectively. The sub-particle (or sub-grid) term Π_ρ is represented by the partial smoothing procedure of Di Monaco et al (2011, [8]) and does not alter the mass conservation. This term remains unchanged and is omitted just for sake of simplicity. The Momentum Equation assumes the form:

$$\frac{d\underline{u}_0}{dt} = -\delta_{i3}\underline{g} - \frac{1}{\rho_0}\left\langle\underline{\nabla}p\right\rangle_{C0,0} + \left\langle\nu\nabla^2\underline{u}\right\rangle_{C0,0} + \Pi_M, \quad \left\langle\underline{\nabla}p\right\rangle_{C0,IN} = -\sum_b\left(p_b - p_0\right)\underline{\nabla}W_b\omega_b \tag{2.13}$$

where the subscript "$_{IN}$" denotes an inner term, far from boundaries. The viscous term (Di Monaco et al., 2011, [8]) zeroes when the inter-particle distance or viscosity are null. This term does not alter the momentum conservation, is not modified during this study and it is often omitted hereafter for sake of simplicity. Π_M (m/s^2) is an approximate SPH sub-particle term (Di Mascio et al., 2017, [29]), function of the well-known artificial viscosity (Monaghan, 2005, [12]), and it stabilizes the co-located SPH schemes. This formulation is mass-conservative at the level of the single SPH particle:

$$\frac{dm_0}{dt} = \frac{d\left(\rho\omega\right)_0}{dt} = \rho_0\frac{d\omega_0}{dt} + \omega_0\frac{d\rho_0}{dt} = \rho_0\omega_0\left\langle\underline{\nabla}\cdot\underline{u}\right\rangle_{C0} - \omega_0\rho_0\left\langle\underline{\nabla}\cdot\underline{u}\right\rangle_{C0} = 0 \tag{2.14}$$

The internal global momentum is not conserved. For any pair-particle sub-system one obtains:

$$\begin{aligned}&\frac{d\underline{p}_0}{dt} + \frac{d\underline{p}_b}{dt} = m_0\frac{d\underline{u}_0}{dt} + \underline{u}_0\frac{dm_0}{dt} + m_b\frac{d\underline{u}_b}{dt} + \underline{u}_b\frac{dm_b}{dt} = \\ &= 2\omega_0\omega_b\left(p_b - p_0\right)\underline{\nabla}W_{0b} \neq 0, \quad \underline{\nabla}W_{0b} = -\underline{\nabla}W_{b0}\end{aligned} \tag{2.15}$$

where $\underline{p}$ (kg×m×s^{-2}) is momentum. It is well known that this formulation is unstable due to a fast propagation of the SPH truncation errors associated with ineffective SPH particle distributions.

2.2.3. SPH-ALE$_{1u}$-LC$_{m,p}$C$_0$ intermediate formulation

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

The SPH-ALE$_0$-LC$_m$C$_0$ formulation (Sec.2.2.2) is here improved by introducing a first ALE velocity increment in the Trajectory Equation "$_{TE}$". This 1-stage ALE formulation is considered unbalanced (ALE$_{1u}$) because the associated ALE terms in the other balance equations are not involved. The ALE$_1$ velocity $\delta_1 \underline{u}_0$ (m/s) provides stability to the SPH formulation and is defined to conserve momentum (C$_{m,p}$). The modified formulation is cited as SPH-ALE$_{1u}$-LC$_{m,p}$C$_0$. In the following, only the modifications with respect to Sec.2.2.2 are reported. The ALE$_1$ velocity increment is the difference between the ALE velocity and the fluid velocity:

$$\underline{u}_{m,0} = \underline{u}_0 + \delta_1 \underline{u}_0 \Rightarrow \left.\frac{d\underline{u}_m}{dt}\right|_{\underline{u}_{m,0}} = \left.\frac{d\underline{u}}{dt}\right|_{\underline{u}_{m,0}} + \left.\frac{d\left(\delta_1 \underline{u}\right)}{dt}\right|_{\underline{u}_{m,0}} \tag{2.16}$$

It is defined to make $\delta_1 \underline{u}_0$ directly proportional to the time step duration Δt (s); its ALE derivative is the ratio between the ALE$_1$ velocity itself and Δt:

$$\delta_1 \underline{u}_0 \equiv f_1\left(t - t_{k-1}\right) \Rightarrow \left(\delta_1 \underline{u}_0\right)_{t_k} = \left(f_1 \Delta t\right)_{t_k}, \quad \left.\frac{d\left(\delta_1 \underline{u}\right)}{dt}\right|_{\underline{u}_{m,0}} = f_1 = \frac{\delta_1 \underline{u}_0}{\Delta t} \tag{2.17}$$

where f_1 is a function defined in the following which is non-null and finite, no matter about the time step duration. This ALE$_1$ velocity is a no-memory term as it zeroes at the beginning of every time step and linearly grows with time during the same time step. The subscript "$_{k-1}$" represents the time step preceding the current one. Combining (2.5), (2.16) and (2.17) one obtains:

$$\left.\frac{d\underline{u}_m}{dt}\right|_{\underline{u}_{m,0}} = \frac{d\underline{u}_0}{dt} + \left[\underline{\nabla} \otimes \underline{u}\right]_{C0,0}^T \delta_1 \underline{u}_0 + \frac{\delta_1 \underline{u}_0}{\Delta t} \tag{2.18}$$

whose RHS second term is negligible with respect to the last term as Δt is small enough:

$$\left[\underline{\nabla} \otimes \underline{u}\right]_0^T \delta_1 \underline{u}_0 = o\left(\frac{\delta_1 \underline{u}_0}{\Delta t}\right), \quad \Delta t \to 0 \Rightarrow \left.\frac{d\underline{u}_m}{dt}\right|_{\underline{u}_{m,0}} \simeq \frac{d\underline{u}_0}{dt} + \frac{\delta_1 \underline{u}_0}{\Delta t} \tag{2.19}$$

The Trajectory Equation of ALE$_{1u}$-LC$_{m,p}$C$_0$ is written in terms of accelerations, instead of velocities, not to make the ALE$_1$ term appear as negligible:

$$\left.\frac{d\underline{u}_m}{dt}\right|_{\underline{u}_{m,0}} = \frac{d\underline{u}_0}{dt} + \frac{1}{\rho_0}\sum_b \left[p_0\left(\frac{\rho_b}{\rho_0} + 1\right) + p_b\left(\frac{\rho_0}{\rho_b} - 1\right)\right] \underline{\nabla} W_b \omega_b + BC_{SA,ALE1-TE} + BC_{BT,ALE1-TE} \tag{2.20}$$

where the boundary "BC" terms refer to the SASPH scheme "$_{SA}$" for fixed surface walls (Di Monaco et al., 2011, [8]) and to the body-transport scheme for volume solid bodies ("$_{BT}$"; Amicarelli et al., 2015, [4]). The ALE$_1$ velocity is formulated to conserve momentum:

$$\begin{aligned}\delta_1 \underline{u}_0 = f_1 \Delta t = \Delta t \Biggl\{ \frac{1}{\rho_0}\sum_b \left[p_0\left(\frac{\rho_b}{\rho_0} + 1\right) + p_b\left(\frac{\rho_0}{\rho_b} - 1\right)\right] \underline{\nabla} W_b \omega_b + \\ + BC_{SA,ALE1-TE} + BC_{BT,ALE1-TE} \Biggr\} \simeq -2\frac{p_0}{\rho_0}\left\langle \underline{\nabla} 1\right\rangle_0 \Delta t\end{aligned} \tag{2.21}$$

The last approximation, which clarifies the meaning of $\delta_1 \underline{u}_0$, is not used in the code because it violates the momentum conservation. The second term in the RHS of (2.20) does not represent the neglected term of (2.18) as also confirmed by the fact that such hypothesis would violate the definition of (2.17): the ALE$_1$ velocity increment would not be proportional to the time step duration. The ALE$_1$ velocity provides stability to the numerical model as the particle trajectories are corrected to move from the most to the less populated regions of the fluid domain so that the particle distribution is closer to a regular and isotropic Cartesian distribution, which minimizes the SPH truncation error ε_p. Further, the correction is proportional to pressure so that it vanishes at the very free surface where the kinematic boundary condition, which constrains the trajectories of the fluid particles along the free surface, should not be altered. Furthermore, $\delta_1 \underline{u}_0$ allows to conserve the internal global momentum, as reported in the following. This ALE$_1$ velocity is directly proportional to Δt and, approximately, to the raw SPH approximation of the unity gradient. This means that $\delta_1 \underline{u}_0$ tends to zero with the time step duration and ε_p. These properties are crucial to make the ALE$_1$ velocity locally negligible with respect to the fluid velocity, which can be fairly approximated by the ALE velocity $\underline{u}_m$:

$$\begin{cases} \lim\limits_{\Delta t \to 0} \delta_1 \underline{u}_0 \to 0 \\ \lim\limits_{\varepsilon_p \to 0} \delta_1 \underline{u}_0 \to 0 \end{cases} \Rightarrow \delta_1 \underline{u}_0 \ll \underline{u}_0 \Rightarrow \underline{u}_0 \simeq \underline{u}_{m,0} \tag{2.22}$$

The ALE$_1$ term can be neglected only when it is expressed in terms of velocity: the time and space derivatives/integrals of the ALE$_1$ velocity in general cannot be approximated by the analogous derivatives of the fluid velocity because they appreciably depend on $\delta_1 \underline{u}_0$:

$$\left.\frac{d\underline{u}_m}{dt}\right|_{\underline{u}_{m,0}} \neq \left.\frac{d\underline{u}}{dt}\right|_{\underline{u}_{m,0}}, \quad \left(\underline{\nabla} \otimes \underline{u}_{m,0}\right) \neq \left(\underline{\nabla} \otimes \underline{u}_0\right), \quad \left.\left(\int_{t_0}^{t} \underline{u} dt\right)\right|_{\underline{u}_{m,0}} \neq \left.\left(\int_{t_0}^{t} \underline{u}_m dt\right)\right|_{\underline{u}_{m,0}} \tag{2.23}$$

The stabilizing contribution of the ALE$_1$ velocity roughly cumulates in time as follows:

$$\left.\left(\int_{t_0}^{t} \delta_1 \underline{u} dt\right)\right|_{\underline{u}_{m,0}} \simeq \left.\left(\int_{t_0}^{t} -2\frac{p}{\rho} \langle \underline{\nabla} 1 \rangle \Delta t dt\right)\right|_{\underline{u}_{m,0}} \approx -2\frac{p_{avg}}{\rho_{ref}} \langle \underline{\nabla} 1 \rangle_{0,avg} \Delta t_{avg} \left(t - t_0\right) \tag{2.24}$$

where the subscript "$_{avg}$" here indicates approximate average values along the particle trajectory.
The second property of (2.22) is satisfied as the SPH approximation of the gradient of the unity tends to zero tending to the continuum, provided a suitable accuracy of the boundary treatment method, as those cited above. According to the introduction of Sec.2 and (2.22), the momentum conservation is satisfied by the conservation of the SPH momentum at any pair-particle sub-system:

$$\begin{gathered} \left.\frac{d\underline{p}}{dt}\right|_{\underline{u}_{m,0}} + \left.\frac{d\underline{p}}{dt}\right|_{\underline{u}_{m,b}} \simeq \left.\frac{d\underline{p}_m}{dt}\right|_{\underline{u}_{m,0}} + \left.\frac{d\underline{p}_m}{dt}\right|_{\underline{u}_{m,b}} = \left.\frac{d\left(m\underline{u}_m\right)}{dt}\right|_{\underline{u}_{m,0}} + \left.\frac{d\left(m\underline{u}_m\right)}{dt}\right|_{\underline{u}_{m,b}} = \\ = \omega_0 \omega_b \left(\frac{p_b}{\rho_b^2} + \frac{p_0}{\rho_0^2}\right)\left(\underline{\nabla} W_{0b} + \underline{\nabla} W_{b0}\right) = 0 \end{gathered} \tag{2.25}$$

This intermediate formulation is further discussed in Sec.2.6.

2.2.4. SPH-ALE$_1$-LC$_l$C$_0$ intermediate formulation

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
The SPH-ALE$_{1u}$-LC$_{m,p}$C$_0$ formulation (Sec.2.2.3) is here improved by introducing the balancing term in the ALE Continuity Equation (2.8), due to the presence of the ALE$_1$ velocity in the Trajectory Equation (2.20):

$$\left.\frac{d\rho}{dt}\right|_{\underline{u}_{m,0}} = -\rho_0 \langle \underline{\nabla} \cdot \underline{u}_m \rangle_{C0,0} + \langle \underline{\nabla} \cdot \left(\rho \delta_1 \underline{u}\right) \rangle_{C0,0} \tag{2.26}$$

The balance equation for the SPH particle volume ω (m^3) is introduced according to (2.10):

$$\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,0}} = \omega_0 \langle \underline{\nabla} \cdot \underline{u}_m \rangle_{C0,0} + \Pi_\omega \tag{2.27}$$

where the sub-particle term Π_ω (m^3/s) simply represents the presence of sub-particle term in the Continuity Equation (2.12) which causes a mass-invariant density modification and thus a mass-invariant volume update. As the volume sub-particle term is already conservative, it will be omitted hereafter for sake of simplicity. Mass is not conserved at the 1-particle level, i.e., SPH particles exchange mass fluxes with the neighbouring particles, in the presence of an ALE scheme:

$$\left.\frac{dm}{dt}\right|_{\underline{u}_{m,0}} = \omega_0 \langle \underline{\nabla} \cdot \left(\rho \delta_1 \underline{u}\right) \rangle_{C0,0}, \quad m_0 = \rho_0 \omega_0 \tag{2.28}$$

where the velocity divergence does not impact the global mass conservation. The SPH-ALE$_1$-LC$_I$C$_0$ formulation is featured by a set of properly balanced ALE equations, but it loses both the conservation properties. For example, mass is not conserved in a generic pair-particle sub-system:

$$\left.\frac{dm}{dt}\right|_{\underline{u}_{m,0}} + \left.\frac{dm}{dt}\right|_{\underline{u}_{m,b}} = -2\omega_0\omega_b\left(\rho_b\delta_1\underline{u}_b - \rho_0\delta_1\underline{u}_0\right)\cdot\underline{\nabla}W_{0b} \neq 0 \qquad (2.29)$$

2.2.5. SPH-ALE$_2$-LC$_m$C$_0$ intermediate formulation

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
The SPH-ALE$_1$-LC$_I$C$_0$ formulation (Sec.2.2.4) is here upgraded by imposing an additional ALE$_2$ velocity to provide mass conservation and only alters the Continuity Equation, which reads:

$$\left.\frac{d\rho}{dt}\right|_{\underline{u}_{m,0}} = -\rho_0\left\langle\underline{\nabla}\cdot\left(\underline{u}+\delta_1\underline{u}\right)\right\rangle_{C0,0} + \left\langle\underline{\nabla}\cdot\left(\rho\delta_1\underline{u}\right)\right\rangle_{C0,0} + 2\rho_0\delta_1\underline{u}_0\cdot\left\langle\underline{\nabla}1\right\rangle_0 \qquad (2.30)$$

The above expression was also found by Sun et al. (2019, [28]), with different motivations and considering a different 1-stage ALE approach. The net mass flux for a generic particle far from boundaries reads:

$$\left.\frac{dm}{dt}\right|_{\underline{u}_{m,0}} = \omega_0\left[\left\langle\underline{\nabla}\cdot\left(\rho\delta_1\underline{u}\right)\right\rangle_{C0,0,IN} + 2\rho_0\delta_1\underline{u}_0\cdot\left\langle\underline{\nabla}1\right\rangle_{0,IN}\right] = -\omega_0\sum_b\left(\rho_b\delta_1\underline{u}_b + \rho_0\delta_1\underline{u}_0\right)\cdot\underline{\nabla}W_b\omega_b \neq 0 \qquad (2.31)$$

Following the same procedure of (2.29), it is proved that the last term of (2.30) is introduced to conserve mass in any pair-particle sub-system:

$$\left.\frac{dm}{dt}\right|_{\underline{u}_{m,0}} + \left.\frac{dm}{dt}\right|_{\underline{u}_{m,b}} = -\omega_0\omega_b\left(\rho_b\delta_1\underline{u}_b + \rho_0\delta_1\underline{u}_0\right)\cdot\left(\underline{\nabla}W_{0b} + \underline{\nabla}W_{b0}\right) = 0 \qquad (2.32)$$

The new ALE$_2$ term in the Continuity Equation (ALE$_2$-CE) is associated with the ALE$_2$ velocity $\delta_2\underline{u}_0$ (m/s), according to (2.8):

$$2\rho_0\delta_1\underline{u}_0\cdot\left\langle\underline{\nabla}1\right\rangle_0 = \left\langle\underline{\nabla}\rho\right\rangle_{C0,0}\cdot\left(\delta\underline{u}_2\right) \qquad (2.33)$$

The assumption that ALE$_2$ velocity is aligned with ALE$_1$ velocity does not violate the Continuity Equation and makes the system (2.33) determined, so that $\delta_2\underline{u}_0$ can be defined as follows:

$$\delta_2\underline{u}_0 \equiv c_1\frac{\left\langle\underline{\nabla}1\right\rangle_0}{\left|\left\langle\underline{\nabla}1\right\rangle_0\right|} \Rightarrow \delta_2\underline{u}_0 = \left(2\rho_0\frac{\left(\delta_1\underline{u}_0\right)\cdot\left\langle\underline{\nabla}1\right\rangle_0}{\left\langle\underline{\nabla}\rho\right\rangle_{C0,0}\cdot\left\langle\underline{\nabla}1\right\rangle_0}\right)\left\langle\underline{\nabla}1\right\rangle_0 \qquad (2.34)$$

Introducing the $\delta_1\underline{u}_0$ approximated formula of (2.21) in (2.34), an analogous approximated expression for the ALE$_2$ velocity is obtained:

$$\delta_2\underline{u}_0 \simeq -4p_0\left(\frac{\left\langle\underline{\nabla}1\right\rangle_0\cdot\left\langle\underline{\nabla}1\right\rangle_0}{\left\langle\underline{\nabla}\rho\right\rangle_{C0,0}\cdot\left\langle\underline{\nabla}1\right\rangle_0}\right)\left\langle\underline{\nabla}1\right\rangle_0\Delta t \qquad (2.35)$$

which satisfies the properties of (2.22) and vanishes at the very free surface.
One assumes that the SPH C$_0$ approximation of the density gradient is systematically much larger than the product of density times the SPH approximation of the unity. This implies that the ALE$_2$ velocity is negligible with respect to the ALE$_1$ velocity:

$$\left\langle\underline{\nabla}\rho\right\rangle_{C0,0} \gg \rho_0\left\langle\underline{\nabla}1\right\rangle_0 \Rightarrow \delta_2\underline{u}_0 \ll -4\frac{p_0}{\rho_0}\frac{\left|\left\langle\underline{\nabla}1\right\rangle_0\right|^2}{\left|\left\langle\underline{\nabla}1\right\rangle_0\right|^2}\left\langle\underline{\nabla}1\right\rangle_0\Delta t = 2\delta_1\underline{u}_0 \Rightarrow \delta_2\underline{u}_0 \ll \delta_1\underline{u}_0 \qquad (2.36)$$

The above assumption has two motivations: density is not homogeneous in the continuum according to the Weakly-Compressible approach, whereas the unity is uniform by definition; if density is quite homogeneous, then the particle distribution is regular enough and the role of ALE velocities becomes negligible. The assumption (2.36) permits not to introduce any ALE$_2$ term in the Trajectory Equation, in the Volume Equation, in the definition of the SPH-ALE velocity and in the velocity-divergence term of the Continuity Equation, also because $\delta_2 \underline{u}_0$ is derived from the last expression of (2.8). Nonetheless, if specific numerical simulations violated the above assumption, then the missing ALE$_2$ terms should be restored. Under these exceptions, the following condition should be respected to make the formulation convergent, preventing from a chain of counter-balancing terms with increasing magnitude:

$$\delta_2 \underline{u}_0 < \delta_1 \underline{u}_0 \Leftrightarrow \left\langle \underline{\nabla} \rho \right\rangle_{C0,0} > 2\rho_0 \left\langle \underline{\nabla} 1 \right\rangle_0 \tag{2.37}$$

In case of positive pressure values, the ALE$_2$-CE term reduces density:

$$2\rho_0 \delta_1 \underline{u}_0 \cdot \left\langle \underline{\nabla} 1 \right\rangle_0 \simeq -4 p_0 \left| \left\langle \underline{\nabla} 1 \right\rangle_0 \right|^2 \Delta t = -\frac{\left(\rho_0 \left| \delta_1 \underline{u}_0 \right| \right)^2}{p_0 \Delta t} \tag{2.38}$$

The above approximation is not directly used by the code because it violates the mass conservation. Following the same approach used for (2.25), one derives that the momentum is not conserved in a generic pair-particle sub-system:

$$\left. \frac{d \underline{p}}{dt} \right|_{\underline{u}_{m,0}} + \left. \frac{d \underline{p}}{dt} \right|_{\underline{u}_{m,b}} \simeq \omega_0 \omega_b \left(\underline{u}_{m,b} - \underline{u}_{m,0} \right) \left[\left(\rho_b \delta_1 \underline{u}_b + \rho_0 \delta_1 \underline{u}_0 \right) \cdot \underline{\nabla} W_{0b} \right] \neq 0 \tag{2.39}$$

2.2.6. SPH-ALE$_3$-LC$_{m,p}$C$_0$ formulation

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

The SPH-ALE$_2$-LC$_m$C$_0$ formulation (Sec.2.2.5) imposes the targeted momentum conservation by introducing the following term in the RHS of the Trajectory Equation (2.20):

$$-\frac{1}{2\rho_0} \sum_b \left(\underline{u}_{m,b} - \underline{u}_{m,0} \right) \left[\left(\rho_b \delta_1 \underline{u}_b + \rho_0 \delta_1 \underline{u}_0 \right) \cdot \underline{\nabla} W_{0b} \right] \omega_b \tag{2.40}$$

For any pair-particle sub-system one obtains:

$$\left. \frac{d \underline{p}}{dt} \right|_{\underline{u}_{m,0}} + \left. \frac{d \underline{p}}{dt} \right|_{\underline{u}_{m,b}} \simeq \omega_0 \omega_b \left(\underline{u}_{m,b} - \underline{u}_{m,0} \right) \left[\left(\rho_b \delta_1 \underline{u}_b + \rho_0 \delta_1 \underline{u}_0 \right) \cdot \underline{\nabla} W_{0b} \right] (1-1) = 0 \tag{2.41}$$

Imposing the same form as the ALE$_1$ velocity (2.17), the ALE$_3$ velocity $\delta_3 \underline{u}_0$ (m/s) reads:

$$\delta_3 \underline{u}_0 = -\frac{\Delta t}{2\rho_0} \sum_b \left(\underline{u}_{m,b} - \underline{u}_{m,0} \right) \left[\left(\rho_b \delta_1 \underline{u}_b + \rho_0 \delta_1 \underline{u}_0 \right) \cdot \underline{\nabla} W_{0b} \right] \omega_b + BC_{SA,ALE3-TE} + BC_{BT,ALE3-TE} \tag{2.42}$$

which satisfies the properties of (2.22) and vanishes at the very free surface. The ALE$_3$ explicit term which should integrate the Continuity Equation is neglected according to the following evidence:

$$\delta_3 \underline{u} = o\left(\delta_1 \underline{u} \right) \Rightarrow \left\langle \underline{\nabla} \cdot \left(\rho \delta_3 \underline{u} \right) \right\rangle_{C0,0} \ll \left\langle \underline{\nabla} \cdot \left(\rho \delta_1 \underline{u} \right) \right\rangle_{C0,0} \tag{2.43}$$

Instead, the implicit terms in the velocity-divergence of the Continuity Equation and the Volume Equation are maintained because they prevent the fluid velocity from appearing in CE. These additional terms do not alter mass conservation which is independent on the velocity divergence.

The set of balance equations for the SPH-ALE$_3$-LC$_{m,p}$C$_0$ formulation involves: the Continuity Equation ("$_{CE}$"); the Momentum Equation ("$_{ME}$"); the Trajectory Equation ("$_{TE}$"); the Volume Equation ("$_{VE}$"). The relationships for the fluid mass and momentum, and the SPH-ALE velocity are also reported, where the terms of the boundary treatments (*BC*) for fixed surface walls ("$_{SA}$") and for mobile solid bodies ("$_{BT}$") will be presented in Sec.2.3:

$$\begin{gathered}
\left. \frac{d\rho}{dt} \right|_{\underline{u}_{m,0}} = -\rho_0 \left\langle \underline{\nabla} \cdot \underline{u}_m \right\rangle_{C0,0} + \left\langle \underline{\nabla} \cdot \left(\rho \delta_1 \underline{u} \right) \right\rangle_{C0,0} + 2\rho_0 \delta_1 \underline{u}_0 \cdot \left\langle \underline{\nabla} 1 \right\rangle_0 + \Pi_\rho, \\
\left\langle \underline{\nabla} \cdot \underline{u}_m \right\rangle_{C0,0} = -\sum_b \left(\underline{u}_{m,b} - \underline{u}_{m,0} \right) \cdot \underline{\nabla} W_b \omega_b + BC_{SA,divu,C0} + BC_{BT,divu,C0}, \\
\left\langle \underline{\nabla} \cdot \left(\rho \delta_1 \underline{u} \right) \right\rangle_{C0,0} = -\sum_b \left(\rho_b \delta_1 \underline{u}_b - \rho_0 \delta_1 \underline{u}_0 \right) \cdot \underline{\nabla} W_b \omega_b + BC_{SA,ALE1-CE,C0} + BC_{BT,ALE1-CE,C0},
\end{gathered} \tag{2.44}$$

$$2\rho_0\delta_1\underline{u}_0\cdot\left\langle\underline{\nabla}1\right\rangle_0 = 2\rho_0\delta_1\underline{u}_0\cdot\left(-\sum_b \underline{\nabla}W_b\omega_b + \underline{BC}_{SA,grad1}+\underline{BC}_{BT,grad1}\right);$$

$$\frac{d\underline{u}_0}{dt} = -\delta_{i3}\underline{g} - \frac{1}{\rho_0}\left\langle\underline{\nabla}p\right\rangle_{C0,0} + \left\langle\nu\nabla^2\underline{u}\right\rangle_{C0,0}+\Pi_M,$$

$$\left\langle\underline{\nabla}p\right\rangle_{C0} = -\sum_b \left(p_b-p_0\right)\underline{\nabla}W_b\omega_b + BC_{SA,gradp,C0}+BC_{BT,gradp,C0};$$

$$\left.\frac{d\underline{u}_m}{dt}\right|_{\underline{u}_{m,0}} = \frac{d\underline{u}_0}{dt}+\frac{\delta_1\underline{u}_0}{\Delta t}+\frac{\delta_3\underline{u}_0}{\Delta t},$$

$$\delta_1\underline{u}_0 = \frac{\Delta t}{\rho_0}\sum_b\left[p_0\left(\frac{\rho_b}{\rho_0}+1\right)+p_b\left(\frac{\rho_0}{\rho_b}-1\right)\right]\underline{\nabla}W_b\omega_b + BC_{SA,ALE1-TE}+BC_{BT,ALE1-TE},$$

$$\delta_3\underline{u}_0 = -\frac{\Delta t}{2\rho_0}\sum_b\left(\underline{u}_{m,b}-\underline{u}_{m,0}\right)\left[\left(\rho_b\delta_1\underline{u}_b+\rho_0\delta_1\underline{u}_0\right)\cdot\underline{\nabla}W_{0b}\right]\omega_b + BC_{SA,ALE3-TE}+BC_{BT,ALE3-TE};$$

$$\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,0}} = \omega_0\left\langle\underline{\nabla}\cdot\underline{u}_m\right\rangle_{C0,0}+\Pi_\omega;\ m_0=\left(\rho\omega\right)_0,\ \ \underline{p}_0\simeq\underline{p}_{m,0}=\left(m\underline{u}_m\right)_0,\ \ \underline{u}_{m,0}=\underline{u}_0+\delta_1\underline{u}_0+\delta_3\underline{u}_0$$

The definitions of the following terms of SPHERA are not reported as they were already presented by Di Monaco et al. (2011, [8]): the inner viscous term, the inner density sub-particle term and the inner velocity sub-particle term (which here depends on the ALE velocity). The fluid velocity is still explicitly involved in (2.44) in the inner and boundary contributions to the viscous term of the Momentum Equation. Far from boundaries, the pressure-gradient approximation is used analogously to Khayyer et al. (2022, [30]) as it is split from any other term. Some of the terms of (2.44) might be merged: this procedure is avoided, also in the code writing, because it prevents from integrating the 1st-order consistency scheme of Sec.3. The SPH-ALE$_3$-LC$_{m,p}$C$_0$ formulation is C$_0$-consistent, mass-conservative, momentum-conservative, stable and ALE-balanced.

2.3. SPH-ALE$_3$-LCC boundary terms

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

The new boundary terms in the SPH-ALE$_3$-LC$_{m,p}$C$_0$ formulation (2.44) are hereafter presented (Sec.2.3.1-Sec.2.3.2). Instead, the ALE$_3$ boundary contributions are null because:

$$\delta_3\underline{u}_{BC}=-\delta_3\underline{u}_0,\quad \rho_{BC}=\rho_0 \tag{2.45}$$

according to Sec.2.3.1. Further, the boundary terms for the ALE$_1$ velocity in the Trajectory Equation are not reported because they are extracted from SPHERA non-ALE formulation for the pressure-gradient term (Sec.2.6). They were presented in Di Monaco et al. (2011, [8]) for the fixed surface walls, and Amicarelli et al. (2015, [4]) and Amicarelli et al. (2022, [7]) for the mobile solid bodies. All the new terms are derived omitting analogous passages already presented in the above-cited papers on fixed surface walls and mobile solid bodies. Further details are available in the SPHERA (2022, [1]) guide.

2.3.1. Mass-flux boundary terms in the Continuity Equation

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

All the ALE boundary terms contribute to a more regular and isotropic particle distribution.

In particular, the ALE$_1$ and ALE$_2$ velocities introduce explicit boundary contributions in the second and third terms (ALE$_1$-CE and ALE$_2$-CE) of the Continuity Equation in the ALE-LCC formulations (2.44) and (3.23). These are the ALE terms responsible for the inter-particle mass fluxes.

The ALE$_1$-CE-C$_0$ term assumes the following expression:

$$\left\langle\underline{\nabla}\cdot\left(\rho\delta_1\underline{u}\right)\right\rangle_{C0,0} = -\sum_b\left(\rho_b\delta_1\underline{u}_b-\rho_0\delta_1\underline{u}_0\right)\cdot\underline{\nabla}W_b\omega_b + BC_{SA,ALE1-CE,C0}+BC_{BT,ALE1-CE,C0}, \tag{2.46}$$

$$BC_{SA,ALE1-CE,C0} = -\sum_{w} \left\{ \left(\rho_{SA} \delta_1 \underline{u}_{SA} - \rho_0 \delta_1 \underline{u}_0 \right) \cdot \left\{ \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \right) J_{2,w} \alpha_s \right] \right\} \right\},$$
$$BC_{BT,ALE1-CE,C0} = -\sum_{s} \left(\rho_s \delta_1 \underline{u}_{0s} - \rho_0 \delta_1 \underline{u}_0 \right) \cdot \underline{\nabla} W_s \omega_s$$

where r (m) is the distance of the barycentre of the spherical infinitesimal volume of solid angle α_s from the computational particle, $\underline{\underline{T}}_w$ is the matrix of directional cosines to change from the local reference system of the wall element to the global reference system. The closed-form solution $J_{2,w}$ (m^{-1}) was presented in Di Monaco et al. (2011, [8]). According to (2.44) and (2.46), the net mass flux of a particle close to boundaries reads:

$$\left. \frac{dm}{dt} \right|_{\underline{u}_{m,0}} = \omega_0 \left[\left\langle \underline{\nabla} \cdot \left(\rho \delta_1 \underline{u} \right) \right\rangle_{C0,0} + 2 \rho_0 \delta_1 \underline{u}_0 \cdot \left\langle \underline{\nabla} 1 \right\rangle_0 \right] = -\omega_0 \left\{ \sum_{b} \left(\rho_b \delta_1 \underline{u}_b + \rho_0 \delta_1 \underline{u}_0 \right) \cdot \underline{\nabla} W_b \omega_b + \right.$$
$$\left. + \rho_0 \sum_{w} \left\{ \left(\delta_1 \underline{u}_{SA} + \delta_1 \underline{u}_0 \right) \cdot \left\{ \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \right) J_{2,w} \alpha_s \right] \right\} \right\} + \rho_0 \sum_{s} \left(\delta_1 \underline{u}_{0s} + \delta_1 \underline{u}_0 \right) \cdot \underline{\nabla} W_s \omega_s \right\} \quad (2.47)$$

Thus, a simple way to impose no mass flux at boundaries for C_0 is to assign the following conditions:

$$\rho_{SA} = \rho_{0s} = \rho_0, \ \left. \underline{\nabla} \rho_{SA} \right|_0 = \left. \underline{\nabla} \rho \right|_{0s} = \underline{0}, \ \delta_1 \underline{u}_{SA} = \delta_1 \underline{u}_{0s} = -\delta_1 \underline{u}_0, \ \underline{\nabla} \otimes \left(\delta_1 \underline{u}_{SA} \right) = \underline{\nabla} \otimes \left(\delta_1 \underline{u}_{0s} \right) = \underline{\underline{0}} \quad (2.48)$$

The ALE$_1$-CE C_0 boundary terms are finally expressed by the following relationships:

$$BC_{SA,ALE1-CE,C0} = 2 \rho_0 \delta_1 \underline{u}_0 \cdot \sum_{w} \left\{ \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \right) J_{2,w} \alpha_s \right] \right\}, \ BC_{BT,ALE1-CE,C0} = 2 \rho_0 \delta_1 \underline{u}_0 \cdot \sum_{s} \underline{\nabla} W_s \omega_s \quad (2.49)$$

The ALE$_2$-CE inner term requires the following boundary contributions for the raw SPH approximation of the gradient of the unity:

$$BC_{SA,grad1} = -\sum_{w} \left\{ \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \right) J_{2,w} \alpha_s \right] \right\}, \ BC_{BT,grad1} = -\sum_{s} \underline{\nabla} W_s \omega_s \quad (2.50)$$

The boundary terms cancel each other out in the C_0 scheme, whereas all the four terms survive in C_1:

$$BC_{SA,ALE1-CE,C0} + 2 \rho_0 \delta_1 \underline{u}_0 \cdot BC_{SA,grad1} = BC_{BT,ALE1-CE,C0} + 2 \rho_0 \delta_1 \underline{u}_0 \cdot BC_{BT,grad1} = 0,$$
$$BC_{SA,ALE1-CE,C1} + 2 \rho_0 \delta_1 \underline{u}_0 \cdot BC_{SA,grad1} \neq BC_{BT,ALE1-CE,C1} + 2 \rho_0 \delta_1 \underline{u}_0 \cdot BC_{BT,grad1} \neq 0 \quad (2.51)$$

where the ALE$_1$-CE-C$_1$ boundary contributions (Sec.3) are derived as follows:

$$BC_{SA,ALE1-CE,C1} = -2 \rho_0 \left\{ \left(\delta_1 \underline{u}_0 \right)_j \underline{\underline{B}}_0 \left\{ \sum_{w} \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \right) J_{2,w} \alpha_s \right] \right\} \right\}_j,$$
$$BC_{BT,ALE1-CE,C1} = -2 \rho_0 \left[\left(\delta_1 \underline{u}_0 \right)_j \underline{\underline{B}}_0 \left(\sum_{s} \underline{\nabla} W_s \omega_s \right) \right]_j \quad (2.52)$$

where B_{ij} is the renormalization matrix defined in Sec.3.2, Einstein's notation applies to the subscript "j" and mixed tensor/subscript notations clarify the content of the equation RHSs.

2.3.2. Boundary contributions to the velocity divergence

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
The ALE$_1$ and ALE$_3$ velocities modify the velocity divergence as components of the ALE velocity. At boundaries, they affect the non-ALE definition (2.88) of the boundary velocity $\underline{u}_{BC}$ (m/s). Slip conditions can be assigned either depending on the slip coefficient (schemes "$_{SA}$" and "$_{BT}$") or as a mirror velocity equal to the neighbouring body-particle velocity (scheme "$_{BT}$"). The latter case is a new code feature ("mirror velocity as solid velocity") which is only used in the Continuity Equation if the velocity field of the solid body better represents the virtual fluid beyond the wall, thus integrating no-slip conditions for the viscous term. Free-surface and confined flows are treated differently. In case of free-surface flows, the following boundary conditions apply. In analogy with (2.48), the ALE$_3$ boundary velocities are expressed as follows:

$$\delta_3 \underline{u}_{SA} = \delta_3 \underline{u}_{0s} = -\delta_3 \underline{u}_0, \quad \underline{\nabla} \otimes (\delta_3 \underline{u}_{SA}) = \underline{\nabla} \otimes (\delta_3 \underline{u}_{0s}) = \underline{\underline{0}} \qquad (2.53)$$

where the subscript "$_{0s}$" depends on the fluid-body inter-particle interaction as the velocity reconstructed over the solid bodies is non-homogeneous, contrarily to the fixed surface walls.
Considering (2.88), (2.48) and (2.53), the boundary ALE velocity $\underline{u}_{m,BC}$ (m/s) reads:

$$\begin{gathered} \underline{u}_{m,BC} = \underline{u}_{BC} + \delta_1 \underline{u}_{BC} + \delta_3 \underline{u}_{BC} = \left[\left(2\underline{u}_{wall} - \underline{u}_{m,0} \right) \cdot \underline{n}_{wall} \right] \underline{n}_{wall} + \\ + \left\{ \left[\underline{u}_0 + 2\phi_s \left(\underline{u}_{wall} - \underline{u}_{m,0} \right) - (1 - 2\phi_s) \left(\delta_1 \underline{u}_0 + \delta_3 \underline{u}_0 \right) \right] \cdot \underline{t}_{wall} \right\} \underline{t}_{wall} \end{gathered} \qquad (2.54)$$

whereas its difference with respect to the ALE velocity of the computational particle, present in the velocity divergence (Sec.0), is derived as follows:

$$\begin{gathered} \underline{u}_{m,BC} - \underline{u}_{m,0} = 2 \left\{ \left[\left(\underline{u}_{wall} - \underline{u}_{m,0} \right) \cdot \underline{n}_{wall} \right] \underline{n}_{wall} + \phi_s \left[\left(\underline{u}_{wall} - \underline{u}_{m,0} \right) \cdot \underline{t}_{wall} \right] \underline{t}_{wall} + \right. \\ \left. + (\phi_s - 1) \left[\left(\delta_1 \underline{u}_0 + \delta_3 \underline{u}_0 \right) \cdot \underline{t}_{wall} \right] \underline{t}_{wall} \right\}, \quad \underline{t}_{wall} = \begin{cases} \dfrac{\underline{u}_t}{|\underline{u}_t|}, & \underline{u}_t \equiv \underline{u}_0 - \left(\underline{u}_0 \cdot \underline{n}_{wall} \right) \underline{n}_{wall} \neq \underline{0} \\ \underline{0}, & \underline{u}_t = \underline{0} \end{cases} \end{gathered} \qquad (2.55)$$

This definition depends on the slip coefficient and is valid both for the SASPH approach and the body-transport scheme (Sec.0). In case the condition "mirror velocity as solid velocity" is activated, then (2.55) is replaced by the following expression:

$$\underline{u}_{m,BC} - \underline{u}_{m,0} = \left(\underline{u}_s + \delta_1 \underline{u}_{BC} + \delta_3 \underline{u}_{BC} \right) - \left(\underline{u}_0 + \delta_1 \underline{u}_0 + \delta_3 \underline{u}_0 \right) = \underline{u}_s - \underline{u}_{m,0} - \left(\delta_1 \underline{u}_0 + \delta_3 \underline{u}_0 \right) \qquad (2.56)$$

The boundary conditions (2.48) and (2.53) guarantee no mass flux at wall, but they are not always the best choice for the Volume Equation. Then, for confined flows, where it is necessary to improve the assessment of the global volume, the following ALE boundary conditions are imposed when representing the velocity divergence, which plays no role in the mass conservation:

$$\delta_1 \underline{u}_{BC} = \delta_3 \underline{u}_{BC} = \underline{0}, \quad BC_{divu}, \quad confined \quad flows \qquad (2.57)$$

This volume correction has been used only with confined flows. It might not be the best choice for free-surface flows because it has the shortcoming of introducing a boundary condition which is not the same as the ALE$_1$-CE term. The latter should partially cancel out with the velocity-gradient term.

2.4. SPH-ALE$_3$-LCC conservative time integration

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
Conservative balance equations do not guarantee mass or momentum conservation if they are not associated with a proper time scheme. A conservative Leapfrog time integration scheme for the SPH-ALE$_3$-LCC formulations is defined as follows:

$$\begin{gathered} m_{0,k+1} = m_{0,k} + \left. \frac{dm}{dt} \right|_{u_{m,0},k+\frac{1}{2}} \Delta t_k = m_{0,k} + \left(\rho_{0,k} \left. \frac{d\omega}{dt} \right|_{u_{m,0},k+\frac{1}{2}} + \omega_{0,k} \left. \frac{d\rho}{dt} \right|_{u_{m,0},k+\frac{1}{2}} \right) \Delta t_k, \\ \rho_{0,k+1} = \rho_{0,k} + \left. \frac{d\rho}{dt} \right|_{u_{m,0},k+\frac{1}{2}} \Delta t_k, \quad \omega_{0,k+1} = \frac{m_{0,k+1}}{\rho_{0,k+1}}; \\ \underline{p}_{m,0,k+\frac{1}{2}} = \underline{p}_{m,0,k-\frac{1}{2}} + \left. \frac{d\underline{p}_m}{dt} \right|_{u_{m,0},k} \Delta t_k, \quad \left. \frac{d\underline{p}_m}{dt} \right|_{u_{m,0},k} = \left(m_{0,k} \left. \frac{d\underline{u}_{m,0}}{dt} \right|_{u_{m,0},k} + \underline{u}_{m,0,k-\frac{1}{2}} \left. \frac{dm}{dt} \right|_{u_{m,0},k-\frac{1}{2}} \right), \\ \underline{u}_{m,0,k+\frac{1}{2}} = \frac{\underline{p}_{m,0,k+\frac{1}{2}}}{m_{0,k}}, \quad \underline{x}_{m,0,k+1} = \underline{x}_{m,0,k} + \left(\underline{u}_{m,0,k+\frac{1}{2}} \right) \Delta t_k \end{gathered} \qquad (2.58)$$

Hereafter any subscript including "$_k$" indicates a time step. Integration applies to mass and density, whereas the volume is updated later. The difference between the internal global mass at two following time steps is null thanks to the procedure (2.32) which conserves mass in any pair-particle sub-system:

$$\sum_0 m_{0,k+1} - \sum_0 m_{0,k} = \Delta t_k \sum_0 \left.\frac{dm}{dt}\right|_{u_{m,0},k+\frac{1}{2}} = 0 \qquad (2.59)$$

This Leapfrog time scheme is mass-conservative for both ALE$_3$-LC$_{m,p}$C$_0$ and ALE$_3$-LC$_m$C$_1$.
It is relevant to apply time integration directly to mass. One considers a different Leapfrog time scheme, which operates on volume and density, and then updates the mass:

$$\rho_{0,k+1} = \rho_{0,k} + \left.\frac{d\rho}{dt}\right|_{u_{m,0},k+\frac{1}{2}} \Delta t_k,\quad \omega_{0,k+1} = \omega_{0,k} + \left.\frac{d\omega}{dt}\right|_{u_{m,0},k+\frac{1}{2}} \Delta t_k,\quad m_{0,k+1} = \rho_{0,k+1}\omega_{0,k+1} \qquad (2.60)$$

The resulting error in the internal global mass is quantified by the presence of a second-order term:

$$\begin{aligned}\sum_0 m_{0,k+1} - \sum_0 m_{0,k} &= \sum_0 \rho_{0,k+1}\left[\omega_{0,k} + \left(\frac{1}{\rho_{0,k}}\left.\frac{dm}{dt}\right|_{u_{m,0},k+\frac{1}{2}} - \frac{\omega_k}{\rho_k}\left.\frac{d\rho}{dt}\right|_{u_{m,0},k+\frac{1}{2}}\right)\Delta t_k\right] - \sum_0 \rho_{0,k}\omega_{0,k} = \\ &= \Delta t_k^2 \sum_0 \left.\frac{d\rho}{dt}\right|_{u_{m,0},k+\frac{1}{2}} \left.\frac{d\omega}{dt}\right|_{u_{m,0},k+\frac{1}{2}} \neq 0\end{aligned} \qquad (2.61)$$

The present time scheme (2.58) directly applies to momentum, then updates the ALE velocity and finally applies to the particle position. The difference between the internal global SPH momentum at two following time steps is null thanks to the procedure (2.41) which conserves momentum in any pair-particle sub-system of ALE$_3$-LC$_{m,p}$C$_0$:

$$\sum_0 \underline{p}_{m,0,k+\frac{1}{2}} - \sum_0 \underline{p}_{m,0,k-\frac{1}{2}} = \Delta t_k \sum_0 \left.\frac{d\underline{p}_m}{dt}\right|_{u_{m,0},k} = 0 \qquad (2.62)$$

The conservation of the global internal SPH momentum implies the conservation of the global internal fluid momentum, as discussed in the introduction of Sec.2. Instead, if a Leapfrog time scheme applies to velocity as follows:

$$\underline{u}_{m,0,k+\frac{1}{2}} = \underline{u}_{m,0,k-\frac{1}{2}} + \left.\frac{d\underline{u}_m}{dt}\right|_{u_{m,0},k} \Delta t_k,\quad \underline{p}_{0,k+\frac{1}{2}} = m_k \underline{u}_{m,0,k+\frac{1}{2}} \qquad (2.63)$$

the updated momentum assumes the following expression:

$$\begin{aligned}\underline{p}_{m,0,k+\frac{1}{2}} &= \left(m_{0,k-1} + \left.\frac{dm}{dt}\right|_{u_{m,0},k-\frac{1}{2}} \Delta t_{k-1}\right)\left(\underline{u}_{m,0,k-\frac{1}{2}} + \left.\frac{d\underline{u}_{m,0}}{dt}\right|_{u_{m,0},k} \Delta t_k\right) = \\ &= \underline{p}_{m,0,k-\frac{1}{2}} + \left.\frac{d\underline{p}_m}{dt}\right|_{u_{m,0},k} \Delta t_k + \underline{u}_{m,0,k-\frac{1}{2}} \left.\frac{dm}{dt}\right|_{u_{m,0},k-\frac{1}{2}} \left(\Delta t_{k-1} - \Delta t_k\right)\end{aligned} \qquad (2.64)$$

violating momentum conservation in case of variable time step:

$$\sum_0 \underline{p}_{m,0,k+\frac{1}{2}} - \sum_0 \underline{p}_{m,0,k-\frac{1}{2}} = \left(\Delta t_{k-1} - \Delta t_k\right)\sum_0 \underline{u}_{m,0,k-\frac{1}{2}} \left.\frac{dm}{dt}\right|_{u_{m,0},k-\frac{1}{2}} \neq 0 \qquad (2.65)$$

The Momentum Equation is nested in the Trajectory Equation and does not need an explicit time integration. The SPH particle and its initial co-located fluid volume describe different trajectories:

$$\left.\underline{x}_{m,0}\right|_t = \left.\underline{x}_{m,0}\right|_{t_0} + \left.\left(\int_{t_0}^{t} \underline{u}_m dt\right)\right|_{\underline{u}_{m,0}} \neq \left.\underline{x}_0\right|_{t_0} + \left.\left(\int_{t_0}^{t} \underline{u} dt\right)\right|_{\underline{u}_{m,0}} \simeq \left.\underline{x}_0\right|_{t_0} + \left.\left(\int_{t_0}^{t} \underline{u} dt\right)\right|_{\underline{u}_0} = \left.\underline{x}_0\right|_t,\quad \left.\underline{x}_{m,0}\right|_{t=t_0} = \left.\underline{x}_0\right|_{t=t_0} \qquad (2.66)$$

where the approximation in (2.66) is due to (2.19). The SPH particle velocity equals at each point of its trajectory the velocity of the local fluid volume which is temporarily intercepted. This condition makes the ALE velocity to be representative of the local fluid velocity and determines that the conservation of the global ALE momentum implies the conservation of the global fluid momentum.

2.5. Volume conservation and mass-variable approach in SPH-ALE$_3$-LCC

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

Under the WC approach, the internal global fluid volume should be conserved only for confined flows with infinitely rigid boundaries. Forcing the exact conservation of the fluid volume is not correct in general. The volume should change slightly in free-surface flows, according to the density variations admitted by the WC approach. Under the ALE$_3$-LCC formulations, the volume of any pair-particle sub-system evolves as follows:

$$\begin{gathered}
\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,0}}+\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,b}}=-2\omega_0\omega_b\left(\underline{u}_{m,b}-\underline{u}_{m,0}\right)\cdot\underline{\nabla}W_{0b}, SPH-ALE_3-LC_{m,p}C_0;\\
\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,0}}+\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,b}}=\omega_0\omega_b\left\{\left[\underline{\underline{B}}_0\left(\underline{u}_b-\underline{u}_0\right)_j\underline{\nabla}W_{0b}\right]_j+\left[\underline{\underline{B}}_b\left(\underline{u}_b-\underline{u}_0\right)_j\underline{\nabla}W_{0b}\right]_j\right\}, SPH-ALE_3-LC_mC_1
\end{gathered}\tag{2.67}$$

The new formulations introduce mass-variable particles and are derived without assuming any specific constraint on the homogeneity of the particle volume. At the same time, the formulations of the kernel gradients assume that the kernel does not depend on the length scale *h* of the kernel support. This does not necessarily imply that the particle size *dx* is homogeneous. The kernel gradient does not depend on the kernel support size of the neighbouring particles, also because the set of balance equation of each particle is solved independently from the other particles. Nonetheless, the SPH truncation error increases with non-homogeneous particle distributions, also with variable-mass particles. Thus, a threshold is here introduced: variable-mass SPH particles become mass-frozen particles only since their mass variation is larger than 50% the initial value. For mass-frozen particles a correction term is introduced in the velocity-gradient to exactly restore null mass-fluxes during each time step since the first mass-freezing until the end of the simulation. Any numerical modification of the velocity gradient reduces the pressure accuracy but does not alter the mass and momentum conservation. In case of mass-frozen particles, the volume equation is altered and assumes the following forms:

$$\begin{gathered}
\left.\frac{dm}{dt}\right|_{\underline{u}_{m,0}}=0\Rightarrow\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,0}}=-\frac{\omega_0}{\rho_0}\left.\frac{d\rho}{dt}\right|_{\underline{u}_{m,0}}=\\
=-\frac{\omega_0}{\rho_0}\left[\rho_0\sum_b\left(\underline{u}_{m,b}-\underline{u}_{m,0}\right)\cdot\underline{\nabla}W_b\omega_b-\sum_b\left(\rho_b\delta_1\underline{u}_b+\rho_0\delta_1\underline{u}_0\right)\cdot\underline{\nabla}W_b\omega_b\right],\ SPH-ALE_3-LC_{m,p}C_0;\\
\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,0}}=-\frac{\omega_0}{\rho_0}\left\{\rho_0\left\{\underline{\underline{B}}_0\left[\sum_b\left(\underline{u}_b-\underline{u}_0\right)_j\underline{\nabla}W_b\omega_b\right]\right\}_j+\right.\\
\left.-\sum_b\left(\rho_b\delta_1\underline{u}_b+\rho_0\delta_1\underline{u}_0\right)\cdot\underline{\nabla}W_b\omega_b\right\},\ SPH-ALE_3-LC_mC_1
\end{gathered}\tag{2.68}$$

To conserve the global momentum, the ALE$_3$ velocity in TE is zeroed for mass-frozen particles. Analogously, their variable-mass neighbours are affected: the mass fluxes and the ALE$_3$-velocity momentum fluxes exchanged with the mass-frozen particles are zeroed.

2.6. Comparisons with some state-of-the-art formulations

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

The present formulation is hereafter briefly compared with some state-of-the-art SPH schemes. The discussion is exclusively limited to the few topics of the present study. From this viewpoint, the traditional SPH formulation can be considered as non-consistent because the pressure-gradient term was rearranged and then approximated as follows:

$$-\frac{1}{\rho}\underline{\nabla}p=\underline{\nabla}\left(\frac{p}{\rho}\right)+\frac{p}{\rho^2}\underline{\nabla}\rho,\ \left\langle\underline{\nabla}\left(\frac{p}{\rho}\right)\right\rangle_{0,IN}+\frac{p}{\rho^2}\left\langle\underline{\nabla}\rho\right\rangle_{0,IN}=\sum_b\left(\frac{p_b}{\rho_b^2}+\frac{p_0}{\rho_0^2}\right)\underline{\nabla}W_b\omega_b\tag{2.69}$$

where the SPH approximations are non-consistent. It was a sort of SPH-ALE$_/$-LC$_{m,p}$C$_/$ formulation, at least far from boundaries. Alternatively, one observes that SPH-ALE$_{1u}$-LC$_{m,p}$C$_0$ (Sec.2.2.3) is practically equivalent to traditional SPH, far from boundaries, provided that the traditional pressure-

gradient term implicitly hides the ALE$_1$ velocity term of the Trajectory Equation, which is in turn hidden within the Momentum Equation. However, this second interpretation seems misleading because the traditional-SPH equations would result unbalanced. The discretization (2.69) has been commonly adopted by the SPH models, with few exceptions (e.g., Khayyer et al., 2021, [31]).
Sun et al. (2019, [28]) represents a state-of-the-art advanced example of SPH studies on PSTs for Local-form SPH mono-phase flows. Their quasi-Lagrangian framework seem equivalent to the Local-form ALE balance equations, after removing the volume and mass equations. They stated to apply one velocity increment. However, their ALE$_1$ balancing term in the Continuity Equation seemed to hide a ALE$_2$ term analogous to the one derived in Sec.2.2.5. That term had repercussions on momentum conservation. The pressure-gradient term was approximated as in (2.69). Their ALE$_1$ velocity was directly proportional to a velocity scale U_{scale} (m/s). This was accompanied by a complicated and coefficient-dependent treatment at the free surface and its intersection with walls, beyond a cut-off limiter in the inner domain. At boundaries, the formulation concerned free-slip conditions. The ALE$_1$ velocity of the present study is approximately equal to the ALE$_1$ velocity of Sun et al. (2019, [28]) if $U_{scale}h$=ca.$p_0\Delta t/\rho_0$. Their ALE$_1$ velocity was not directly proportional to the time step duration with possible concerns for the condition $\underline{u}_m$=ca.$\underline{u}_0$. This was counterbalanced by the dependency on h, beyond the gradient of the unity. There was no boundary term for ALE$_1$ velocity as it would result from the approximation of the gradient of the unity at boundaries. Particle mass was constant: the equivalent mass and volume equations, even if not explicitly represented, were not ALE-balanced. Conservative time integration and consistency methods were not considered. They also adopted an advanced tensile-stability correction term. On the other hand, the present 3-stage ALE C$_0$-consistent formulation conserves mass and momentum and does not need any specific treatment at the free-surface. It considers the ALE boundary terms of the balance equations, several slip conditions, a mass-variable approach and a conservative Leapfrog time scheme. Nonetheless, the stability of the present formulation might be insufficient for laminar flows, if the pressure levels are too low far from the free surface. When C$_1$ consistency is more useful than momentum conservation, the present C$_0$ formulation is generalized and replaced by the C$_1$ formulation of Sec.4.

2.7. Viscous inner term in the Momentum Equation

The present section refers to Di Monaco et al. (2011, [8]).
The expression for the viscous inner term used in SPHERA reads (Di Monaco et al., 2011, [8]):

$$\left\langle \nu \frac{\partial^2 u_i}{\partial x_j^2} \right\rangle = -4 \sum_b \left(\frac{m_b \nu_0 \nu_b}{\nu_0 \rho_0 + \nu_b \rho_b + \varepsilon_{\nu\rho}} u_{i,0b} \frac{\left|\underline{\nabla} W_b\right|}{r_{0,b}\left(r_{0,b}^2 + \varepsilon_{r,2}\right)} r_{0,b}^2 \right) \tag{2.70}$$

It is a viscous term, whose formulation is mixes Cleary's and Morris' formulations (Basa et al., 2009, [32]) where $\varepsilon_{\nu\rho}$ and $\varepsilon_{r,2}$ are negligible constants avoiding divergent behaviours. In case $\nu\rho$ and r do not tend to zero, then (2.70) assumes the following form:

$$\left\langle \nu \frac{\partial^2 u_i}{\partial x_j^2} \right\rangle = -4 \sum_b \left(m_b \frac{\nu_0 \nu_b}{\nu_0 \rho_0 + \nu_b \rho_b} \frac{u_{i,0b}}{r_{0,b}} \left|\underline{\nabla} W_b\right| \right) \tag{2.71}$$

One considers Morris' term (Basa et al., 2009, [32]):

$$\left\langle \nu \frac{\partial^2 u_i}{\partial x_j^2} \right\rangle = \sum_b m_b u_{i,0b} \frac{\left(\mu_0 + \mu_b\right)}{\rho_0 \rho_b} \frac{\underline{r}_{0,b} \cdot \underline{\nabla} W_b}{r_{0,b}^2} \tag{2.72}$$

and that the following expression:

$$4 \frac{\nu_0 \nu_b}{\nu_0 \rho_0 + \nu_b \rho_b} = 4 \frac{\mu_0 \mu_b}{\rho_0 \rho_b \left(\mu_0 + \mu_b\right)} \tag{2.73}$$

is a factor analogous to Cleary's formulation (Basa et al., 2009, [32]):

$$\frac{\left(\mu_0 + \mu_b\right)}{\rho_0 \rho_b} \tag{2.74}$$

It follows that (2.71) is equal to Morris' term only in case of uniform density and viscosity:

$$\underline{\nabla} W_b = \frac{\underline{r}_{0,b}}{r_{0,b}} \left.\frac{\partial W}{\partial r}\right|_b$$
$$\left\langle \nu \frac{\partial^2 u_i}{\partial x_j^2} \right\rangle = \sum_b m_b \frac{(\mu_0 + \mu_b)}{\rho_0 \rho_b} \frac{u_{i,0b}}{r_{0,b} + \varepsilon_{r,3}} \left.\frac{\partial W}{\partial r}\right|_b = -\sum_b m_b \frac{(\mu_0 + \mu_b)}{\rho_0 \rho_b} \frac{u_{i,0b}}{r_{0,b} + \varepsilon_{r,3}} \left|\underline{\nabla} W_b\right| \quad (2.75)$$

Zeroing (molecular) viscosity reduces CPU time. A configuration with no-slip conditions and null viscosity makes sense if it is demonstrated that the value of viscosity is non-influential. No-slip conditions might possibly provide the same results as free-slip conditions if both the following conditions are satisfied: the test case approximately provides no-slip conditions even if free-slip conditions are used; dx is small enough.

2.8. Equation of State and its inverse function

Following the Weakly-Compressible approach (Monaghan, 2005, [12]), often used by CFD methods, the Equation of State is linear, barotropic, and dependent on the artificial bulk modulus *K* (Pa):

$$p = K \frac{(\rho - \rho_{ref})}{\rho_{ref}} + p_{ref}, \qquad \rho = \rho_{ref} \left[1 + \frac{(p - p_{ref})}{K} \right] \quad (2.76)$$

where the subscript "*ref*" indicates the mixture reference status at null normal stress.
According to the Weakly-Compressible approach and assuming a linearized barotropic Equation of State, the bulk modulus K (Pa) can be expressed as follows:

$$K = \rho_{ref} \frac{(p - p_{ref})}{(\rho - \rho_{ref})} = A_K \left| p - p_{ref} \right|_{\max}, \quad A_K = \frac{\rho}{\left| \rho - \rho_{ref} \right|_{\max}} \quad (2.77)$$

where A_k is the reciprocal of the maximum density relative displacement. It is suggested to set A_k=20 to guarantee that density variations lie within 5% of the reference density. The first attempt *K*-value is here designed according to the scale quantities of the phenomenon:

$$K_{design} = A_K \max \left\{ \frac{1}{2} \rho U_{scale}^2 \left| C_{p,scale} \right|; \rho g \Delta z_{scale}; \left| p_{scale} - p_{ref} \right| \right\} \quad (2.78)$$

Its final input value is iteratively chosen based on the simulated pressure field.

2.9. Sub-particle term of the Continuity Equation: partial pressure smoothing

In 3D, the partial pressure smoothing procedure is executed according to the following expression:

$$p_{0,\vartheta_p} = p_0 + \vartheta_p \frac{\sum_b (p_b - p_0) W_b \omega_b + BC_{DBSPH,\vartheta_p} + BC_{SASPH,\vartheta_p} + BC_{BODY,\vartheta_p}}{A} + B,$$
$$A = \begin{cases} 1, & \sigma_0 < 0.95 \\ \sigma_0, & \sigma_0 \geq 0.95 \end{cases}, \quad B = \begin{cases} (p_{ref} - p_0)(1 - \sigma_0), & \sigma_0 < 0.95 \\ \sigma_0, & \sigma_0 \geq 0.95 \end{cases} \quad (2.79)$$

where the coefficient θ_p is computed according to Di Monaco et al. (2011, [8]), BC symbols represent the boundary terms provided by the different boundary treatment schemes and σ is the discrete Shepard coefficient. For $\sigma_0 \geq 0.95$ the SPH approximation is 0th-order consistent whereas for $\sigma_0 < 0.95$ the truncated portion of the kernel support is virtually filled with fictitious particles with p=p_{ref}, under the hypothesis that $(1 - \sigma_0) \neq 0$ is only due to zones not covered by SPH elements. This assumption prevents from 0th-order consistency, but it is usually more accurate than the first formulation at the free surface.

2.10. Sub-particle term of the Momentum Equation

The approximate SPH sub-particle term Π_M (m/s^2) in the Momentum Equation (2.44) stabilizes the co-located SPH schemes. Its expression in SPHERA refers to Di Monaco et al. (2011, [8]), according to Monaghan (1992, [33]):

$$-\nu_M \sum_b \frac{m_b}{\rho_0 r_{0b}^2} \left(\underline{u}_b - \underline{u}_0\right) \cdot \left(\underline{x}_b - \underline{x}_0\right) \left.\frac{\partial W}{\partial x_i}\right|_b \tag{2.80}$$

where r (m) is the distance between two interacting particles and ν_M (m^2×s^{-1}) stands for the artificial viscosity:

$$\nu_M = \frac{\alpha h c}{\rho} \tag{2.81}$$

The inner artificial viscosity term is conservative provided that the particle artificial viscosity and the particle density are homogeneous.

2.11. General relationships for the kernel gradient

The kernel function W (m^{-3}) is expressed depending on the non-dimensional distance q from the position of the computational particle:

$$q \equiv f_1\left(x, y, z, x_0, y_0, z_0, h\right) \equiv \frac{\left(\underline{x} - \underline{x}_0\right)}{h} = \frac{r}{h} \tag{2.82}$$

Following the differential form of W, the kernel gradient can be expressed by useful relationships depending on the kernel derivative along q or r, and the unit vector aligned with the distance from the computational particle:

$$\begin{gathered} \underline{\nabla} W = \frac{\partial W}{\partial q} \underline{\nabla} q, \quad \underline{\nabla} q = \frac{1}{h} \underline{\nabla} r = \frac{1}{h} \frac{\underline{r}}{r} \Rightarrow \\ \Rightarrow \underline{\nabla} W = \frac{1}{h} \frac{\partial W}{\partial q} \frac{\underline{r}}{r} = -\left|\underline{\nabla} W\right| \frac{\underline{r}}{r} = \frac{dW}{dr} \frac{\underline{r}}{r} = \frac{dW}{dr} \underline{\nabla} r, \quad \left|\underline{\nabla} W\right| = -\frac{dW}{dr} \geq 0, \quad \underline{\nabla} r = \frac{\underline{r}}{r} \end{gathered} \tag{2.83}$$

In the above equations the length scale of the kernel support h (m) is assumed uniform. This does not necessarily imply that the particle volume is uniform because the kernel gradient should not depend on the kernel size of the neighbouring particles.

2.12. Kernel functions and gradients

SPHERA adopts the two kernel functions reported hereafter.

The beta-spline kernel of Monaghan & Lattanzio (1985, [34]) is expressed by the following function:

$$W = \begin{cases} \frac{1}{\pi h^3}\left(1 - \frac{3}{2} q^2 + \frac{3}{4} q^3\right), & 0 \leq q < 1 \\ \frac{1}{4\pi h^3}\left(2 - q\right)^3, & 1 \leq q < 2 \\ 0, & 2 \leq q \end{cases} \tag{2.84}$$

whose gradient reads:

$$\underline{\nabla} W = \begin{cases} \frac{1}{\pi h^4}\left(\frac{9}{4} q^2 - 3q\right) \frac{\underline{r}}{r}, & 0 \leq q < 1 \\ -\frac{3}{4\pi h^4}\left(2 - q\right)^2 \frac{\underline{r}}{r}, & 1 \leq q < 2 \\ 0, & 2 \leq q \end{cases} \tag{2.85}$$

The anti-cluster kernel of Gallati & Braschi (2000, [35]) is expressed by the following expression:

$$W = \begin{cases} \dfrac{15}{64\pi h^3}\left(2-q\right)^3, & 0 \le q < 2 \\ 0, & 2 \le q \end{cases} \tag{2.86}$$

The resulting 3D gradient reads:

$$\underline{\nabla} W = \begin{cases} -\dfrac{45}{64\pi h^4}\left(2-q\right)^2 \dfrac{\underline{r}}{r}, & 0 \le q < 2 \\ 0, & 2 \le q \end{cases} \tag{2.87}$$

For the anti-cluster kernel, the 2D formulae for both the kernel function and its gradient are obtained by multiplying the 3D formulae by 4*h*/3. In the range $1 \le q \le 2$, the anti-cluster kernel function/gradient equals 15/16 of the beta-spline kernel function/gradient. The beta-spline kernel is used everywhere, except for the pressure-gradient term in the Momentum Equation. Nonetheless, the SASPH boundary scheme uses the beta-spline kernel with no exception.

2.13. The boundary velocity

The boundary functions of the SASPH scheme (Sec.5) and the scheme for mobile solid bodies (Sec.6) rely on the approximation of a boundary velocity $\underline{u}_{BC}$, which depends on the slip coefficient ϕ_s, the wall normal and tangential unit vectors ($\underline{n}_{wall}$, $\underline{t}_{wall}$) and the wall velocity $\underline{u}_{wall}$:

$$\begin{gathered} \underline{f}_s \simeq \underline{u}_{BC}, \quad = \underline{f}_{SA} \simeq \underline{u}_{BC}, \quad \left(\underline{\nabla} \otimes \underline{f}\right)_{SA} = \underline{0}, \\ \underline{u}_{BC} = \left[\left(2\underline{u}_{wall} - \underline{u}_0\right) \cdot \underline{n}_{wall}\right]\underline{n}_{wall} + \left\{\left[\underline{u}_0 + 2\phi_s\left(\underline{u}_{wall} - \underline{u}_0\right)\right] \cdot \underline{t}_{wall}\right\}\underline{t}_{wall}, \\ \underline{u}_{wall,SA} = \underline{0}, \quad \underline{n}_{wall,SA} = \underline{n}_w, \quad \underline{t}_{wall,SA} = \underline{t}_w, \quad \phi_{s,SA,3D} = 0, \quad \phi_{s,SA,2D} = 1, \\ \underline{u}_{wall,BD} = \underline{u}_{0s}, \quad \underline{n}_{wall,BD} = \underline{n}_{0s}, \quad \underline{t}_{wall,BD} = \underline{t}_{0s}, \quad \phi_{s,BD} = 0,1 \end{gathered} \tag{2.88}$$

The SASPH scheme manages fixed surface wall elements "$_w$" under free-slip conditions in 3D and no-slip conditions in 2D. The scheme for solid bodies deals with either slip conditions, both in 3D and 2D, and approximates the fluid-body interface velocity with the interaction velocity $\underline{u}_{0s}$. In both schemes, $\underline{t}_{wall}$ depends on the non-normal component of the velocity of the computational particle.

2.14. Raw, C$_0$ and C1 consistent SPH approximations of the pressure-gradient with SPHERA boundary terms

The code SPHERA uses the "$_{SA}$" (Di Monaco et al., 2011, [8]) and the "$_{BD}$" (Amicarelli et al., 2015, [4]) boundary schemes. The associated C$_0$ SPH approximation for the pressure gradient reads:

$$\begin{gathered} \left\langle \underline{\nabla} p \right\rangle_{C0,0} = -\sum_b \left(p_b - p_0\right)\underline{\nabla} W_b \omega_b - \sum_s \left(p_s - p_0\right)\underline{\nabla} W_s \omega_s + \\ -\rho_0 \sum_w \left\{ \underline{\underline{T}}_w \left\{ \left[\sum_{\alpha_s} \left(\underline{\nabla} r \otimes \underline{\nabla} r\right) J_{3,w} \alpha_s \right] \left(\underline{\underline{T}}_w^T \underline{g}\right) \right\} \right\}, \quad f_s = p_{0s}, \quad f_{SA} = p_0, \quad \underline{\nabla} f\big|_{SA} = \rho_0 \underline{g} \end{gathered} \tag{2.89}$$

SPHERA C$_1$ approximation of the pressure-gradient is expressed as follows:

$$\begin{gathered} \left\langle \underline{\nabla} p \right\rangle_{C1,0} = \sum_b \left(p_b - p_0\right)\left(\underline{\underline{B}}_0 \underline{\nabla} W_b\right)\omega_b + \sum_s \left(p_s - p_0\right)\left(\underline{\underline{B}}_0 \underline{\nabla} W_s\right)\omega_s + \\ +\rho_0 \underline{\underline{B}}_0 \left\{ \sum_w \underline{\underline{T}}_w \left\{ \left[\sum_{\alpha_s} \left(\underline{\nabla} r \otimes \underline{\nabla} r\right) J_{3,w} \alpha_s \right] \left(\underline{\underline{T}}_w^T \underline{g}\right) \right\} \right\} \end{gathered} \tag{2.90}$$

In 2D, the SASPH formulation assumes a different form (Di Monaco et al., 2011, [8]), so that the 2D C$_0$ approximation reads:

$$\left\langle \underline{\nabla} p \right\rangle_{C0,0,2D} = -\sum_b \left(p_b - p_0\right)\underline{\nabla} W_b \omega_b - \sum_s \left(p_s - p_0\right)\underline{\nabla} W_s \omega_s + \tag{2.91}$$

$$+\left[\underline{\underline{I}}-\underline{\underline{R}}_s\int_{A'_h} W dA\right]^{-1}\left[-\underline{n}_w\left(p_{SA}+p_0\right)\int_{P_{1,w}}^{P_{2,w}} W ds+\rho_0\underline{\underline{R}}_n\underline{g}\right],$$

$$p_{SA,2D}=p_0+\rho_0\frac{\left(r_{P_2}+r_{P_1}\right)}{2}\left(\underline{n}_w\cdot\underline{g}\right),\quad R_{n,ij}\equiv n_{w,i}n_{w,j},\quad R_{s,ij}\equiv t_{w,i}t_{w,j}$$

where each solid segment is delimited by the points $P_{1,w}$ and $P_{2,w}$.
The 2D C$_1$ approximation of the pressure-gradient in SPHERA is expressed as follows:

$$\begin{gathered}\left\langle\underline{\nabla}p\right\rangle_{C1,0,2D}=\sum_b\left(p_b-p_0\right)\left(\underline{\underline{B}}_0\underline{\nabla}W_b\right)\omega_b+\sum_s\left(p_s-p_0\right)\left(\underline{\underline{B}}_0\underline{\nabla}W_s\right)\omega_s+\\-\underline{\underline{B}}_0\left\{\left[\underline{\underline{I}}-\underline{\underline{R}}_s\int_{A'_h} W dA\right]^{-1}\left[-\underline{n}_w\left(p_{SA}+p_0\right)\int_{P_{1,w}}^{P_{2,w}} W ds+\rho_0\underline{\underline{R}}_n\underline{g}\right]\right\}\end{gathered}\tag{2.92}$$

2.15. Raw, C$_0$ and C1 SPH approximations of the ALE velocity gradient and divergence with SPHERA boundary terms

The code SPHERA uses the SASPH (Di Monaco et al., 2011, [8]) and the solid-body (Amicarelli et al., 2015, [4]) boundary schemes. The SASPH scheme manages fixed surface wall elements "$_w$" under free-slip conditions in 3D and no-slip conditions in 2D. The scheme for solid bodies deals with either slip conditions, both in 3D and 2D, and approximates the fluid-body interface velocity with the interaction velocity $\underline{u}_{0s}$. In both schemes, $\underline{t}_{wall}$ depends on the non-normal component of the velocity of the computational particle. The associated C$_0$ terms of the SPH approximation of the ALE velocity gradient, here just recalled, were presented in the above papers:

$$\begin{gathered}\left\langle\underline{\nabla}\otimes\underline{u}_m\right\rangle_{C0,0}=-\sum_b\left[\left(\underline{u}_{m,b}-\underline{u}_{m,0}\right)\otimes\underline{\nabla}W_b\right]^T\omega_b-\sum_s\left[\left(\underline{u}_{BD}-\underline{u}_{m,0}\right)\otimes\underline{\nabla}W_s\right]^T\omega_s+\\-\sum_w\left\{\left(\underline{u}_{SA}-\underline{u}_{m,0}\right)\otimes\left\{\underline{\underline{T}}_w\left[\sum_{\alpha_s}\left(\underline{\nabla}r\right)J_{2,w}\alpha_s\right]\right\}\right\}^T\end{gathered}\tag{2.93}$$

SPHERA C$_1$ approximation of the velocity-gradient, with the boundary velocity (2.55)-(2.57), is introduced as follows:

$$\begin{gathered}\left\langle\underline{\nabla}\otimes\underline{u}_m\right\rangle_{C1,0}=\underline{\underline{B}}_0\sum_b\left[\left(\underline{u}_{m,b}-\underline{u}_{m,0}\right)\otimes\underline{\nabla}W_b\right]^T\omega_b+\underline{\underline{B}}_0\sum_s\left[\left(\underline{u}_{BD}-\underline{u}_{m,0}\right)\otimes\underline{\nabla}W_s\right]^T\omega_s+\\+\underline{\underline{B}}_0\sum_w\left\{\left(\underline{u}_{SA}-\underline{u}_{m,0}\right)\otimes\left\{\underline{\underline{T}}_w\left[\sum_{\alpha_s}\left(\underline{\nabla}r\right)J_{2,w}\alpha_s\right]\right\}\right\}^T\end{gathered}\tag{2.94}$$

The C$_1$ approximation of the velocity divergence assumes the following form:

$$\begin{gathered}\left\langle\underline{\nabla}\cdot\underline{u}_m\right\rangle_{C1,0}=\left\{\underline{\underline{B}}_0\left[\sum_b\left(\underline{u}_{m,b}-\underline{u}_{m,0}\right)_j\underline{\nabla}W_b\omega_b\right]\right\}_j+\left\{\underline{\underline{B}}_0\left[\sum_s\left(\underline{u}_{BD}-\underline{u}_{m,0}\right)_j\underline{\nabla}W_s\omega_s\right]\right\}_j+\\+\left\{\underline{\underline{B}}_0\left\{\sum_w\left(\underline{u}_{SA}-\underline{u}_{m,0}\right)_j\underline{\underline{T}}_w\left[\sum_{\alpha_s}\left(\underline{\nabla}r\right)J_{2,w}\alpha_s\right]\right\}\right\}_j\end{gathered}\tag{2.95}$$

where a mixed tensor/subscript notation is adopted to make the formula more accessible, with Einstein's notation applied to the subscript "$_j$". In 2D, the SASPH scheme adopts a different formulation (Di Monaco et al., 2011, [8]), so that the 2D C$_0$ approximation of the velocity gradient reads:

$$\left\langle\underline{\nabla}\otimes\underline{u}_m\right\rangle_{C0,0,2D}=-\sum_b\left[\left(\underline{u}_b-\underline{u}_{m,0}\right)\otimes\underline{\nabla}W_b\right]^T\omega_b-\sum_s\left[\left(\underline{u}_{BD}-\underline{u}_{m,0}\right)\otimes\underline{\nabla}W_s\right]^T\omega_s+\tag{2.96}$$

$$-\sum_{w}\left\{\left[\left(\underline{u}_{SA}-\underline{u}_{m,0}\right)\otimes\underline{n}_{w}\right]^{T}\int_{P_{1,w}}^{P_{2,w}}Wds\right\}$$

The 2D C$_1$ approximation of the velocity gradient and divergence are finally defined:

$$\begin{aligned}
\left\langle\underline{\nabla}\otimes\underline{u}_{m}\right\rangle_{C1,0,2D}&=\underline{\underline{B}}_{0}\sum_{b}\left[\left(\underline{u}_{m,b}-\underline{u}_{m,0}\right)\otimes\underline{\nabla}W_{b}\right]^{T}\omega_{b}+\underline{\underline{B}}_{0}\sum_{s}\left[\left(\underline{u}_{BD}-\underline{u}_{m,0}\right)\otimes\underline{\nabla}W_{s}\right]^{T}\omega_{s}+\\
&+\underline{\underline{B}}_{0}\sum_{w}\left\{\left[\left(\underline{u}_{SA}-\underline{u}_{m,0}\right)\otimes\underline{n}_{w}\right]^{T}\int_{P_{1,w}}^{P_{2,w}}Wds\right\},\\
\left\langle\underline{\nabla}\cdot\underline{u}\right\rangle_{C1,0,2D}&=\left\{\underline{\underline{B}}_{0}\left[\sum_{b}\left(\underline{u}_{m,b}-\underline{u}_{m,0}\right)_{j}\underline{\nabla}W_{b}\omega_{b}\right]\right\}_{j}+\left\{\underline{\underline{B}}_{0}\left[\sum_{s}\left(\underline{u}_{BD}-\underline{u}_{m,0}\right)_{j}\underline{\nabla}W_{s}\omega_{s}\right]\right\}_{j}+\\
&+\left\{\underline{\underline{B}}_{0}\left[\sum_{w}\left(\underline{u}_{SA}-\underline{u}_{m,0}\right)_{j}\underline{n}_{w}\int_{P_{1,w}}^{P_{2,w}}Wds\right]\right\}_{j}
\end{aligned}\qquad(2.97)$$

2.16. Dirichlet's boundary conditions for the water depth

The numerical scheme for Dirichlet's boundary conditions on water depth is reported in the present section. The zone associated with these boundary conditions are defined as $BC_{z,max}$ and has to be width at least 2h along any horizontal direction of the discretized domain. Many $BC_{z,max}$ zones can be defined within the same domain.

Every $BC_{z,max}$ zone is continuously filled with SPH particles from the maximum local height of the SPH particles, or from the height of the solid bottom in case of dry conditions, until the height of the free-surface level provided by the input quantity $z_{BC,max}$ (m), with local hydrostatic or uniform conditions for pressure (according to the user choice).

The velocity of the new emitted particles is imposed by means of a complete SPH smoothing procedure for the velocity field, selectively executed to exclude the new particles from the role of SPH neighbours. It is not necessary to modify the quantities of the SPH particles already present in the zone $BC_{z,max}$. Every zone of this kind has to be delimited by at least 5 adjacent open sections in the input files, in order to correctly impose Dirichlet's boundary conditions for the fluid depth. Nonetheless, the zones $BC_{z,max}$ can also be used with other purposes (e.g., to treat water reservoirs with a fixed level). The solid bottom (e.g., the DEM-DTM) has to be locally represented by a uniform and isotropic Cartesian grid.

A synthetic description of the algorithm of the boundary treatment of the zones $BC_{z,max}$ is reported in the following.

Before the execution of the simulation time steps, and analogously to imposing initial conditions for the inlet sections, the program unit "*BC_zmax_t0*" (executed only in 3D, with the boundary treatment method SASPH) selects the zone vertices and the maximum DEM-DTM heights (useful for dry conditions; "*BC_zmax_t0*" calls the program unit "*z_min_max_DEM_DTM_9p_stencil*").

Every time step, after the update of the particle positions (and analogously to the inlet section update), the program unit "*BC_zmax_anyt*" is called (only in 3D, with the boundary treatment method SASPH), which calls in turn the subroutine "*z_FS_max_9p_stencil*" (both for the Leapfrog time scheme and the last time stage of the explicit Runge-Kutta time schemes).

Every time step, after the SPH neighbouring search (i.e., after the execution of the program unit "*CalcVarLength*" and its sub-units), following the subroutine "*BC_zmax_anyt*", a complete SPH smoothing procedure for velocity is applied only to the new particles emitted in the $BC_{z,max}$ zones (only in 3D, with the boundary treatment method SASPH), excluding the new particles from the role of SPH neighbours. The procedure is executed with any time integration scheme of SPHERA. At the end of the initialization of the new particles, the particle zone is formally changed from the $BC_{z,max}$

zone to the zone of the associated solid bottom *Car_top_zone*, so that the new particles can be treated as standard computational nodes after the initialization, even during their staying in the $BC_{z,max}$ zone. The new program units for the treatment of the $BC_{z,max}$ zones are "*BC_zmax_t0*", "*BC_zmax_anyt*" and "*z_FS_max_9p_stencil*". The following program units have been extracted from the subroutine "*GeneratePart*" to make the code more effective: "*domain_edges*", "*pos_plus_white_noise*" "*z_min_DEM_DTM_9p_stencil*", "*particle_position_extrusion*", "*particles_in_out_dams*". The program unit "*z_min_DEM_DTM_9p_stencil*" has been generalized and renamed "*z_min_max_DEM_DTM_9p_stencil*".

2.17. Inlet sections for Bazin's sharp-crested weirs

The present section is reported as a preprint of Amicarelli et al. (2022, [36]).
A non-stationary rectangular inlet section of SPHERA can be defined as a weir-like inlet section. In this case, SPHERA estimates the inlet weir water depth and imports the inlet weir flow rate from input time series, when available. Otherwise, the inlet flow rate is reconstructed by any pre-processing model. Swamee's formula (1988, [37]) is the reference solution for the flow rate overtopped Q_w (m^3/s) on a sharp-crested Bazin's weir with non-negligible upstream velocity:

$$Q_w = \frac{2}{3}\mu_w\left(h, H_w\right)L_w h\sqrt{2gh} \tag{2.98}$$

where h (m) is the reservoir free-surface height over the weir top. L_w (m) and H_w (m) are the weir length and height, respectively. In SPHERA, a weir inlet section approximates (2.98) by means of a constant discharge coefficient μ_w. This is equal to the "Least-Relative-Error Average" (LRRA) of the maximum and minimum values of μ_w. Further, h approximates the water depth at the weir section. This approach avoids the a-priori knowledge of the time series of the inlet water depth.
The LRRA average x_{LRRA} of the extreme values of a generic quantity x belonging to the interval [x_{min}, x_{max}] is here defined as the constant value which represents the above interval minimizing the relative error ε_r. The error obtained by replacing the minimum value x_{min} with x_{LRRA} is set equal to the error associated with the replacement of x_{max} with x_{LRRA}:

$$\frac{x_{LRRA}-x_{\min}}{x_{\min}} = \frac{x_{\max}-x_{LRRA}}{x_{\max}} \Rightarrow x_{LRRA} = \frac{x_{\max}x_{\min}}{x_{mp}} = \frac{x_{GM}^2}{x_{mp}} \tag{2.99}$$

where x_{mp} is the arithmetic (mid-point) average. Any other relative error within the above interval is smaller than the error at the extremes. The application of Swamee's formula (1988, [37]) reveals that $\mu_{w,max}$=1.20 and $\mu_{w,min}$=0.61. After (2.99), $\mu_{w,LRRA}$=0.81, with a relative error ε_r for both μ_w and Q equal to $\varepsilon_{r,Q,LRRA}$=33%. The arithmetic average and the geometric average "$_{GM}$" do not achieve this result: $x_{LRRA} \le x_{GM} \le x_{mp}$. Their estimates and relative errors are $\mu_{w,GM}$=0.85, $\varepsilon_{r,Q,GM}$=39%, $\mu_{w,mp}$=0.90 and $\varepsilon_{r,Q,mp}$=50%. Swamee's formula (1988, [37]) is approximated with the constant coefficient $\mu_{w,LRRA}$:

$$Q_{w,LRRA} = \frac{2}{3}\mu_{w,LRRA}L_w h\sqrt{2gh}, \quad \mu_{w,LRRA} = 0.81 \tag{2.100}$$

The inversion of (2.100) provides a closed-form solution for the water depth of an inlet weir section:

$$h_{LRRA} = \frac{3\sqrt{3}}{2\sqrt[3]{g}}\left(\frac{Q_w}{\mu_{w,LRRA}L_w}\right)^{\frac{2}{3}} = 2.99\frac{Q_w^{\frac{2}{3}}}{L_w^{\frac{2}{3}}g^{\frac{1}{3}}} \tag{2.101}$$

The application of the error propagation law to (2.101) reads:

$$\varepsilon_h = \frac{\partial h}{\partial Q_w}\varepsilon_Q = \frac{2}{3}2.99\frac{\varepsilon_Q}{Q_w^{\frac{1}{3}}L_w^{\frac{2}{3}}g^{\frac{1}{3}}} \tag{2.102}$$

The relative error on the water depth $\varepsilon_{r,h}$ is equal to 2/3 the relative error on the flow rate:

$$\varepsilon_{r,h} \equiv \frac{\varepsilon_h}{h} = \frac{2}{3}\frac{\varepsilon_Q}{Q_w} = \frac{2}{3}\varepsilon_{r,Q} \tag{2.103}$$

The replacement of $\mu_w(h,H_w)$ with $\mu_{w,LRRA}$ determines a maximum relative error on the inlet water depth of $\varepsilon_{r,h,LRRA}$=22%, but three benefits are obtained: the assessment of the time dependent inlet water depth is only function of the flow rate overtopped and the weir length; the execution of numerical methods to invert Swamee's formula (1988, [37]) is avoided; the relative error associated with a constant discharge coefficient is minimized.

2.18. Stability criteria

Time integration is constrained by the following stability criteria:

$$dt = \min_0\left\{CFL\frac{2h}{c+|\underline{u}|};C_\nu\frac{2h^2}{\nu};C_\nu\frac{2h^2}{\nu_{M,0}}\right\} \tag{2.104}$$

where *dt* (s) is the time step duration and *CFL* the Courant-Friedrichs-Lewy number. Following Adami et al. (2021, [16]), the viscous term stability parameter is set to C_ν=0.05.
An a-priori estimation of the elapsed time (t_e) can assume the following form:

$$t_e \propto \frac{1}{dx^{D+A}} \propto N^{\frac{D+A}{3}} \Rightarrow \frac{t_{e1}}{t_{e2}} = \left(\frac{dx_2}{dx_1}\right)^{D+A} \tag{2.105}$$

where *D* is the domain dimensionality and *A* varies from 1 to 2.
Under the simplifying hypothesis that the particle system of equations are independent, one obtains (at a fixed time step duration):

$$t_e \propto \left.\frac{1}{dx^D}\right|_{dt=\overline{dt}} \propto N \tag{2.106}$$

In case all the time steps are ruled by the *CFL* criterion, then *A*=1 and the following relationships are valid (at a fixed number of particles):

$$dt \propto h \propto dx \;\Rightarrow\; t_e \propto \left.\frac{1}{dx}\right|_{N=\overline{N}} \tag{2.107}$$

In case all the time steps are ruled by the stability criterion on the viscous term, then *A*=2 and the following relationships are valid (at a fixed number of particles):

$$dt \propto h^2 \propto dx^2 \;\Rightarrow\; t_e \propto \left.\frac{1}{dx^2}\right|_{N=\overline{N}} \tag{2.108}$$

In order to understand which stability criterion dominates, one considers the following ratio:

$$\frac{dt_{C_\nu}}{dt_{CFL}} = \frac{C_\upsilon h\left(c+|\underline{u}|_{\max}\right)}{CFL\cdot\nu_{\max}} \tag{2.109}$$

Provided a time step, if the ratio (2.109) is greater than 1, then the *CFL* criterion dominates. Otherwise, the viscosity stability criterion dominates.
When the *CFL* criterion dominates, the maximum viscosity has never any effect on *dt* if this condition is respected:

$$\nu_{\max} \le \frac{C_\nu hc}{CFL} \Rightarrow dt \ne f\left(\nu_{\max}\right) anytime \tag{2.110}$$

The elapsed time also depends on the global specific surface of the fluid domain. The highest the latter, the lowest the first. The global specific surface of the fluid domain is related to the mean number of neighbouring particles.

2.19. SPH monitors

SPHERA makes use of point and line monitors where the particle pressure, velocity and density are interpolated. The introduction of the C_1 consistency method for the first derivatives of the balance equations is accompanied by the presentation of a C_1 consistency method applied to the functions to be approximated at the monitoring points by means of the SPH method. The whole procedure of SPH monitoring is revised. In the absence of C_1 consistency option for monitors ("C1_monitors"), SPH approximation is C_0-consistent. All the SPH fluid particles, body particles and SASPH elements are involved in the SPH C_1 approximation of pressure. Inner body particles are excluded for the SPH C_1 approximation of velocity, due to a lower-order reconstruction scheme with respect to pressure. Analogously, all the body and SASPH elements are excluded in the reconstruction of density, also to prevent ambiguous definitions for possible multi-phase applications.

3. SPH-ALE$_3$-LC$_M$C$_1$ SCHEME (1ST-ORDER CONSISTENCY)

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
A generalized version of the C_1 method of renormalization is presented (Sec.3.1), with boundary terms for mobile solid bodies and fixed surface walls involved both in the definition of the renormalization matrix (Sec.3.2) and as object of the renormalization operator. The integration in the C_0 formulation of Sec.2 provides the SPH-ALE$_3$-LC$_m$C$_1$ formulation (Sec.3.3). The free-surface boundary conditions and the negative-pressure regions are finally discussed (Sec.3.4).

3.1. A C_1 consistency procedure for linear convolution methods with any boundary scheme and application to SPH

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
Taylor's approximation of a generic linear function *f* provides the following expression:

$$f = f_0 + \left.\frac{\partial f}{\partial x_j}\right|_0 \left(x_j - x_{j,0}\right) \qquad (3.1)$$

where the subscript "*j*" follows Einstein's notation. A 1st-order consistency "C1" approximation is a flawless approximation of a linear function and its gradient. Thus, the C_1 approximations at $\underline{x}_0$ are the unknowns of (3.1):

$$\left\langle f \right\rangle_{c1,0} = f_0, \ \left\langle \underline{\nabla} f \right\rangle_{c1,0} = \left.\underline{\nabla} f\right|_0 \qquad (3.2)$$

One assumes a generic convolution numerical method to approximate the above function and its gradient; the symbol "$\langle\ \rangle$" denotes the raw non-corrected approximation. Such method can include any kind of boundary term and is assumed to be linear so that the following conditions are satisfied:

$$\left\langle f_0 \right\rangle_0 = f_0 \left\langle 1 \right\rangle_0, \ \left\langle \left.\underline{\nabla} f\right|_0 \right\rangle_0 = \left.\underline{\nabla} f\right|_0 \left\langle 1 \right\rangle_0 \qquad (3.3)$$

The numerical approximation of the generic function *f* at $\underline{x}_0$ reads:

$$\left\langle f \right\rangle_0 = f_0 \left\langle 1 \right\rangle_0 + \left.\frac{\partial f}{\partial x_j}\right|_0 \left\langle x_j - x_{j,0} \right\rangle_0 \qquad (3.4)$$

The differentiation and approximation of (3.1) provides the following expression:

$$\left\langle \frac{\partial f}{\partial x_i} \right\rangle_0 = f_0 \left\langle \frac{\partial 1}{\partial x_i} \right\rangle_0 + \left.\frac{\partial f}{\partial x_j}\right|_0 \left\langle \frac{\partial \left(x_j - x_{j,0}\right)}{\partial x_i} \right\rangle_0 \qquad (3.5)$$

The resulting system is made of 4 equations in 4 unknowns:

$$\begin{Bmatrix} \left\langle f \right\rangle_0 \\ \left\langle \frac{\partial f}{\partial x} \right\rangle_0 \\ \left\langle \frac{\partial f}{\partial y} \right\rangle_0 \\ \left\langle \frac{\partial f}{\partial z} \right\rangle_0 \end{Bmatrix} = \begin{bmatrix} \left\langle 1 \right\rangle_0 & \left\langle x - x_0 \right\rangle_0 & \left\langle y - y_0 \right\rangle_0 & \left\langle z - z_0 \right\rangle_0 \\ \left\langle \frac{\partial 1}{\partial x} \right\rangle_0 & \left\langle \frac{\partial (x - x_0)}{\partial x} \right\rangle_0 & \left\langle \frac{\partial (y - y_0)}{\partial x} \right\rangle_0 & \left\langle \frac{\partial (z - z_0)}{\partial x} \right\rangle_0 \\ \left\langle \frac{\partial 1}{\partial y} \right\rangle_0 & \left\langle \frac{\partial (x - x_0)}{\partial y} \right\rangle_0 & \left\langle \frac{\partial (y - y_0)}{\partial y} \right\rangle_0 & \left\langle \frac{\partial (z - z_0)}{\partial y} \right\rangle_0 \\ \left\langle \frac{\partial 1}{\partial z} \right\rangle_0 & \left\langle \frac{\partial (x - x_0)}{\partial z} \right\rangle_0 & \left\langle \frac{\partial (y - y_0)}{\partial z} \right\rangle_0 & \left\langle \frac{\partial (z - z_0)}{\partial z} \right\rangle_0 \end{bmatrix} \begin{Bmatrix} f_0 \\ \left.\frac{\partial f}{\partial x}\right|_0 \\ \left.\frac{\partial f}{\partial y}\right|_0 \\ \left.\frac{\partial f}{\partial z}\right|_0 \end{Bmatrix} \qquad (3.6)$$

From the first equation, the following expression is obtained:

$$f_0 = \frac{\left\langle f \right\rangle_0 - \left.\frac{\partial f}{\partial x_j}\right|_0 \left\langle x_j - x_{j,0} \right\rangle_0}{\left\langle 1 \right\rangle_0} \tag{3.7}$$

which paves the way to the solution for the C$_1$ approximation of a generic function:

$$\left\langle f \right\rangle_{c1,0} = \frac{\left\langle f \right\rangle_0}{\left\langle 1 \right\rangle_0} + \left\langle \frac{\partial f}{\partial x_j} \right\rangle_{c1,0} \left(x_{j,0} - \frac{\left\langle x_j \right\rangle_0}{\left\langle 1 \right\rangle_0} \right) \tag{3.8}$$

which is introduced in the SPH approximations of the monitoring points and lines of SPHERA.

This is equivalent to assigning the C$_0$ approximation of *f* at the point $\frac{\left\langle x_j \right\rangle_0}{\left\langle 1 \right\rangle_0}$ and then applying a linearization to $\underline{x}_0$. The C$_1$ approximation in the Right-Hand Side (RHS) of (3.8) is hereafter obtained from the other equations of the system.

The renormalization matrix B_{ij} and the 0th-order consistency C$_0$ approximation are introduced:

$$B_{ij,0}^{-1} \equiv -\left\langle \underline{\nabla} \otimes \underline{x} \right\rangle_{c0,0} = -\left\langle \frac{\partial x_j}{\partial x_i} \right\rangle_{C0,0}, \quad \left\langle \underline{\nabla} f \right\rangle_{c0,0} \equiv \left\langle \underline{\nabla}\left(f - f_0 \right) \right\rangle_0 \tag{3.9}$$

where the symbol "$\otimes$" indicates the tensor product. The solution for the C$_1$ approximation of a gradient is written for any linear convolution method and any boundary treatment scheme:

$$\left\langle \underline{\nabla} f \right\rangle_0 = \left\langle \underline{\nabla} f_0 \right\rangle_0 - \underline{\underline{B}}_0^{-1} \left\langle \underline{\nabla} f \right\rangle_{c1,0} \Rightarrow \left\langle \underline{\nabla} f \right\rangle_{c1,0} = -\underline{\underline{B}}_0 \left\langle \underline{\nabla} f \right\rangle_{c0,0} \tag{3.10}$$

When the SPH method is selected, the inverse of the renormalization matrix assumes the following expression far from boundaries:

$$B_{ij,0}^{-1} = \sum_b \left(\underline{x}_b - \underline{x}_0 \right)_j \left.\frac{\partial W}{\partial x_i}\right|_b \omega_b \tag{3.11}$$

In the continuum, the SPH renormalization matrix equals the opposite of the identity matrix. Under this ideal case, B_{ij} does not introduce any correction. Once computed, the same renormalization matrix applies to all the interactions with the neighbours. As the matrix product is a non-commutative linear operator, the renormalization matrix can enter/exit the SPH summations only if the order of the matrix products is not altered, as in the following expression, valid far from boundaries:

$$\left\langle \underline{\nabla} f \right\rangle_{c1,0} = \underline{\underline{B}}_0 \left[\sum_b \left(f_b - f_0 \right) \underline{\nabla} W_b \omega_b \right] = \sum_b \left(f_b - f_0 \right) \left(\underline{\underline{B}}_0 \underline{\nabla} W_b \right) \omega_b \tag{3.12}$$

During the post-processing sub-steps, the code SPHERA assesses several variables along some input monitors. Here each function *f* is not known yet. Under such cases, the following expression, equivalent to (3.12), prevents from doubling the interaction cycle over the neighbouring particles:

$$\left\langle \underline{\nabla} f \right\rangle_{c1,0} = \underline{\underline{B}}_0 \left(\left\langle \underline{\nabla} f \right\rangle - f_0 \left\langle \underline{\nabla} 1 \right\rangle \right) = -\underline{\underline{B}}_0 \left(\sum_b f_b \underline{\nabla} W_b \omega_b - f_0 \sum_b \underline{\nabla} W \omega_b \right) \tag{3.13}$$

After the summations in (3.13) are elaborated, the function and the gradient are computed.

The C$_0$ approximation of a function can be obtained from (3.8) by zeroing the gradient:

$$\left\langle f \right\rangle_{c0,0} = \frac{\left\langle f \right\rangle_0}{\left\langle 1 \right\rangle_0}, \quad \left\langle 1 \right\rangle_0 \equiv \sigma_0 \equiv \sum_b W_b \omega_b, \quad \left\langle f \right\rangle_0 \equiv \sum_b f_b W_b \omega_b \tag{3.14}$$

When the SPH method is selected, the raw approximation of the unity far from boundaries is defined as the discrete Shepard coefficient σ. Keeping far from boundaries, the raw and C$_0$ approximation of a function gradient are represented by the following expressions:

$$\left\langle \underline{\nabla} f \right\rangle_0 = -\sum_b f_b \underline{\nabla} W_b \omega_b, \quad \left\langle \underline{\nabla} f \right\rangle_{c0,0} = -\sum_b \left(f_b - f_0 \right) \underline{\nabla} W_b \omega_b \tag{3.15}$$

3.2. A SPH renormalization matrix with boundary terms

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

The C$_1$, C$_0$ and raw SPH approximations of (3.12) and (3.15) are valid far from boundaries for a generic SPH model. The code SPHERA here uses two boundary methods: the semi-analytical approach "$_{SA}$" (Di Monaco et al., 2011, [8]) for fixed surface walls and the scheme for body transport "$_{BD}$" (Amicarelli et al., 2015, [4]; Amicarelli et al., 2020, [5]; RSE code modifications in Sec.5 of Paggi et al. 2021, [6]; Amicarelli et al., 2022, [7]) for mobile solid bodies. The first scheme introduces surface wall elements and fills the truncated part of the kernel support with virtual infinitesimal spherical volumes. The second scheme defines solid body particles in the truncated part of the kernel support. The raw and C$_0$ SPH approximations, already presented in the studies mentioned above, are first recalled. Then, the new C$_1$ SPH approximations are presented.

The raw SPH approximation of SPHERA reads:

$$\begin{aligned} \left\langle \underline{\nabla} f \right\rangle_0 = & -\sum_b f_b \underline{\nabla} W_b \omega_b - \sum_b f_s \underline{\nabla} W_s \omega_s + \\ & - f_{SA} \sum_w \left\{ \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \right) J_{2,w} \alpha_s \right] \right\} - \sum_w \left\{ \underline{\underline{T}}_w \left\{ \left[\sum_{\alpha_s} \left(\underline{\nabla} r \otimes \underline{\nabla} r \right) J_{3,w} \alpha_s \right] \left(\underline{\underline{T}}_w^T \underline{\nabla} f_{SA} \right) \right\} \right\} \end{aligned} \tag{3.16}$$

where the "$_{SA}$" boundary terms are the last two ones, and the "$_{BD}$" boundary term is the third to last. The subscript "$_s$" refers to a generic neighbouring volume body particle, the subscript "$_w$" denotes a "$_{SA}$" wall surface element, α_s is a generic solid angle, $J_{2,w}$ (m^{-1}) and $J_{3,w}$ are integral closed-form solutions presented in Di Monaco et al. (2011, [8]), $\underline{r}$ (m) is the distance from the computational particle. $\underline{\underline{T}}_w$ is the matrix of directional cosines to change from the local reference system of the SASPH wall element to the global reference system; the transposed matrix of $\underline{\underline{T}}_w$ is equal to its inverse matrix and permits the inverse change of reference system. The definition of the boundary quantities f_s, f_{SA} and $\underline{\nabla} f_{SA}$ depends on the peculiar spatial reconstruction scheme of the boundary method for a specific function. The C$_0$ approximation of SPHERA assumes the following expression:

$$\begin{aligned} \left\langle \underline{\nabla} f \right\rangle_{C0,0} = & -\sum_b \left(f_b - f_0 \right) \underline{\nabla} W_b \omega_b - \sum_b \left(f_s - f_0 \right) \underline{\nabla} W_s \omega_s + \\ & - \left(f_{SA} - f_0 \right) \sum_w \left\{ \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \right) J_{2,w} \alpha_s \right] \right\} - \sum_w \left\{ \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \otimes \underline{\nabla} r \right) J_{3,w} \alpha_s \right] \left(\underline{\underline{T}}_w^T \underline{\nabla} f_{SA} \right) \right\} \end{aligned} \tag{3.17}$$

The C$_1$ approximation of SPHERA is presented as follows:

$$\begin{aligned} \left\langle \underline{\nabla} f \right\rangle_{C1,0} = -\underline{\underline{B}}_0 \left\langle \underline{\nabla} f \right\rangle_{C0,0} = & \sum_b \left(f_b - f_0 \right) \underline{\underline{B}}_0 \underline{\nabla} W_b \omega_b + \sum_b \left(f_s - f_0 \right) \underline{\underline{B}}_0 \underline{\nabla} W_s \omega_s + \\ & + \left(f_{SA} - f_0 \right) \underline{\underline{B}}_0 \sum_w \left\{ \underline{\underline{T}}_w \left[\sum_{\alpha_s} \left(\underline{\nabla} r \right) J_{2,w} \alpha_s \right] \right\} + \underline{\underline{B}}_0 \sum_w \left\{ \underline{\underline{T}}_w \left\{ \left[\sum_{\alpha_s} \left(\underline{\nabla} r \otimes \underline{\nabla} r \right) J_{3,w} \alpha_s \right] \left(\underline{\underline{T}}_w^T \underline{\nabla} f_{SA} \right) \right\} \right\} \end{aligned} \tag{3.18}$$

The renormalization matrix applies to all the inner and boundary terms. At the same time, B_{ij} is defined including other boundary contributions, where the following relationships apply:

$$\underline{f} = \underline{x}, \quad f_{SA} = \underline{x}_0, \quad \left(\underline{\nabla} \otimes \underline{f} \right)_{SA} = \underline{\underline{I}}, \quad f_{BD} = \underline{x}_s \tag{3.19}$$

Starting from (3.9), the definition of the inverse of the renormalization matrix in SPHERA, boundary terms included, reads:

$$\underline{\underline{B}}_0^{-1} = \underline{\underline{B}}_{0,in}^{-1} + \underline{\underline{B}}_{0,BD}^{-1} + \underline{\underline{B}}_{0,SA}^{-1} = \sum_b \left[\left(\underline{x}_b - \underline{x}_0 \right) \otimes \underline{\nabla} W_b \right]^T \omega_b + \sum_s \left[\left(\underline{x}_s - \underline{x}_0 \right) \otimes \underline{\nabla} W_s \right]^T \omega_s + \tag{3.20}$$

$$+\sum_{w}\left\{\underline{\underline{T}}_w\left\{\left[\sum_{\alpha_s}\left(\underline{\nabla}r\otimes\underline{\nabla}r\right)J_{3,w}\alpha_s\right]\underline{\underline{T}}_w^T\right\}\right\}$$

The inverse of the renormalization matrix is the sum of the inverse of the inner renormalization matrix $B_{ij,in}$, the body renormalization matrix $B_{ij,BD}$ and the SASPH renormalization matrix $B_{ij,SA}$. B_{ij} is symmetric. For the inner and the body matrices, this relies on the kernel properties. For the inner term one obtains:

$$\sum_b\left(\underline{x}_b-\underline{x}_0\right)_i\left.\frac{\partial W}{\partial x_j}\right|_b\omega_b=\sum_b\frac{\left(\underline{x}_b-\underline{x}_0\right)_i\left(\underline{x}_b-\underline{x}_0\right)_j}{r}\left.\frac{dW}{dr}\right|_b\omega_b=\sum_b\left(\underline{x}_b-\underline{x}_0\right)_j\left.\frac{\partial W}{\partial x_i}\right|_b\omega_b \tag{3.21}$$

The symmetry of the SASPH renormalization matrix is intuitively and numerically observed since it is a double matrix product where the first matrix is the inverse of the third one and the second matrix is symmetric. The order of the tensor product has not to be altered.

It is worth noticing that the 2D version of the SASPH scheme in SPHERA has a different formulation with respect to its 3D version. The 2D SASPH renormalization matrix is presented as follows:

$$\underline{\underline{B}}_{0,SA,2D}^{-1}=-\left[\underline{\underline{I}}-\underline{\underline{R}}_s\int_{A_h'}WdA\right]^{-1}\left[\underline{\underline{R}}_n\underline{\underline{I}}\right],\quad R_{n,ij}\equiv n_{w,i}n_{w,j},\quad R_{s,ij}\equiv t_{w,i}t_{w,j} \tag{3.22}$$

where $\underline{t}$ is the tangential unit vector and A_h' (m^2) is the truncated portion of the 2D kernel support.

3.3. SPH-ALE$_3$-LC$_m$C$_1$ balance equations

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).

The application of the formulae (2.90) and (2.95) to the C$_0$ formulation (2.44) provides the following SPH-ALE$_3$-LC$_m$C$_1$ scheme:

$$\begin{gathered}\left.\frac{d\rho}{dt}\right|_{\underline{u}_{m,0}}=-\rho_0\left\langle\underline{\nabla}\cdot\underline{u}_m\right\rangle_{C1,0}+\left\langle\underline{\nabla}\cdot\left(\rho\delta_1\underline{u}\right)\right\rangle_{C0,0}+2\rho_0\delta_1\underline{u}_0\cdot\left\langle\underline{\nabla}1\right\rangle_0+\Pi_\rho;\\ \frac{d\underline{u}_0}{dt}=-\delta_{i3}\underline{g}-\frac{1}{\rho_0}\left\langle\underline{\nabla}p\right\rangle_{C1,0}+\left\langle\nu\nabla^2\underline{u}\right\rangle_{C0,0}+\Pi_M;\\ \left.\frac{d\underline{u}_m}{dt}\right|_{\underline{u}_{m,0}}=\frac{d\underline{u}_0}{dt}+\frac{\delta_1\underline{u}_0}{\Delta t}+\frac{\delta_3\underline{u}_0}{\Delta t};\quad\left.\frac{d\omega}{dt}\right|_{\underline{u}_{m,0}}=\omega_0\left\langle\underline{\nabla}\cdot\underline{u}_m\right\rangle_{C1,0}+\Pi_\omega\end{gathered} \tag{3.23}$$

The SPH approximations of the first derivatives appearing in the continuum ALE balance equations are C$_1$-consistent, but momentum is not conserved. If requested in input, the ALE$_2$-CE term (a native derivative in the continuum) can be made C$_1$-consistent in SPHERA as follows:

$$\begin{gathered}\left\langle\underline{\nabla}\cdot\left(\rho\delta_1\underline{u}\right)\right\rangle_{C1,0}=\left\langle\underline{\nabla}\cdot\left(\rho\delta_1\underline{u}\right)\right\rangle_{C1,0,in}+BC_{SA,ALE1-CE,C1}+BC_{BT,ALE1-CE,C1},\\ \left\langle\underline{\nabla}\cdot\left(\rho\delta_1\underline{u}\right)\right\rangle_{C1,0,in}=\left\{\underline{\underline{B}}_0\sum_b\left(\rho_b\delta_1\underline{u}_b-\rho_0\delta_1\underline{u}_0\right)_j\underline{\nabla}W_b\omega_b\right\}_j\end{gathered} \tag{3.24}$$

where the boundary terms are defined in (2.52). However, this is not suggested as a default choice because the formulation would lose mass conservation. The C$_1$ consistency method should not be applied to the SPH raw approximations within the ALE$_1$ and ALE$_3$ terms as they are not approximations of derivatives in the continuum, but ALE conservation and stability terms. In fact, they do not involve 0th-order consistency SPH derivatives in the C$_0$ formulation. Any consistency method directly applied to these terms would make them reduce or vanish and possibly destabilize the formulation. The approximation of the viscous term is left C$_0$-consistent as it plays a secondary role in this study. The C$_1$ approximation of the ALE velocity divergence required the partial recoding of the C$_0$ boundary terms due to algorithm needs, but their expressions, here omitted, are still defined as in Di Monaco et al. (2011, [8]) and Amicarelli et al. (2022, [7]). The current formulation conserves mass as the velocity-divergence and the pressure-gradient terms do not influence the mass balance equation.

3.4. Free-surface boundary conditions and negative pressure

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
The free-surface boundary conditions and the presence of negative-pressure values are discussed.
ALE terms of the present formulations automatically vanish towards the free surface as they are directly proportional to the pressure of the computational particle p_0. However, the SPH particles closest to the air phase do not exactly lie on the free surface. Each node has a particle volume: the layer of SPH particles describing the free surface is lowered of roughly $dx/2$ with respect to the simulated free surface and is interested by a slightly positive pressure. Further, pressure fluctuations might also occur. In the following, hydrostatic conditions are discussed to provide some simple quantifications, but the qualitative comments and the numerical solution have a more general scope.
Any ALE formulation should not alter the kinematic boundary conditions to be imposed at the free surface, where SPH particles should not be displaced normally with respect to the local free surface.
When the ALE$_3$-LC$_m$C$_1$ formulation applies, the free-surface boundary contribution to the pressure-gradient term in the Momentum Equation is relatively small (null in case of hydrostatic conditions). The application of the ALE$_1$ velocity $\delta_1\underline{u}_0$ at the layer of particles representing the free surface makes them artificially oscillate vertically. The associated errors and pressure waves rapidly propagate throughout the whole domain. During these oscillations, particle distribution tends to become less homogeneous and isotropic, especially at the free surface. Here, the determinants of the inverse of the renormalization matrices decrease and the matrices might become bad-conditioned. The above oscillations have to be avoided. The solution here adopted is the replacement of the ALE$_3$-LC$_m$C$_1$ formulation with ALE$_3$-LC$_{m,p}$C$_0$ at the free surface. It is not necessary to exactly detect the free surface, as spurious oscillatory motions of isolated particles can be damped by the fluid bulk. At the same time, where the free surface is over-detected, the C$_1$ improvements are absent, but the ALE$_3$-LC$_{m,p}$C$_0$ still works without appreciable issues. The free-surface detection is ruled by introducing a minimum threshold on the absolute value of the determinant of the renormalization matrix, which also prevents from error propagation due to matrix bad-conditioning. The default threshold assumes the value $\left|\underline{\underline{B}}_0^{-1}\right|_{thr}$ =0.614. This is provided as an input variable: its possible change is meant to better detect the numerical free surface, not to match experimental data for tuning purposes. The application of this procedure is permitted by the peculiar C$_1$ consistency approach which involves boundary terms in the definition of the renormalization matrix and the C$_1$ application to the boundary terms of the balance equations.
When the C$_1$ formulation is not chosen, ALE$_3$-LC$_{m,p}$C$_0$ applies all over the domain. The missing free-surface boundary contribution $BC_{air,gradp}$ (m/s^2) to the pressure-gradient term in the Momentum Equation can be roughly quantified under hydrostatic conditions:

$$BC_{air,gradp} = \frac{1}{\rho_0}\int_{V_{air,BC}} \left(p_{air,BC} - p_0\right)\underline{\nabla}W dV, \; p_{air,BC} = -\rho_{ref,liq}\, g\left(z_{air} - z_0\right), \; \int_{V_{air,BC}} \underline{\nabla}W dV - \sum_b \underline{\nabla}W_b \omega_b \simeq 0 \tag{3.25}$$

where $V_{air,BC}$ (m^3) is the portion of the kernel support truncated by the free surface. Here no air particle is simulated (mono-phase approach). The air boundary pressure is not the null air pressure, but the negative pressure over the air sub-domain as reconstructed from the fluid sub-domain to impose the correct pressure boundary conditions at the free surface. The integral of the kernel gradient over the truncated part of the kernel support is approximately equal to the opposite of the SPH approximation of the gradient of the unity. Without renormalization, the numerical free surface lowers due to the absence of the BC contribution from the air sub-domain. Solving (3.25) one obtains an approximate relationship between the ALE$_1$ velocity term in the Trajectory Equation and the missing free-surface boundary term in the Momentum Equation:

$$BC_{air,gradp} \simeq -\frac{1}{\rho_0}\left[\rho_0 g h + p_0\right]\left\langle \underline{\nabla}1 \right\rangle_0 \Rightarrow \tag{3.26}$$

$$\Rightarrow \frac{\delta_1 \underline{u}_0}{\Delta t} \simeq \frac{2p_0}{p_0+\rho g h} BC_{air,gradp} \in \begin{cases} [2,\infty) BC_{air,gradp}, p_0 \le -\rho g h \\ (-\infty, 0] BC_{air,gradp}, -\rho g h \le p_0 < 0 \\ [0,2] BC_{air,gradp}, p_0 \ge 0 \end{cases}$$

As p_0 of the free-surface particles is systematically slightly positive, $\delta_1\underline{u}_0$ partly plays the role of the missing free-surface boundary term in the Momentum Equation. The generalized oscillation of the free surface, already discussed in the presence of ALE_3-LC_mC_1 at the very free surface, is strongly reduced or becomes negligible. If pressure is negative and its absolute value is larger than the average air boundary pressure ($-\rho gh$), then an analogous, but less pronounced, damping effect is present. If pressure is negative and its absolute value is smaller than the average boundary pressure, than oscillations are amplified. However, negative-pressure particles with small pressure fluctuations are usually isolated so that their oscillations seem to be effectively damped by the fluid bulk.
In conclusion, under C_0 formulation, the ALE_1 term in the Trajectory Equation partly plays the role of the missing free-surface boundary contribution to the pressure-gradient term in the Momentum Equation. This advantage seems systematically more relevant than the errors due to applying an ALE correction normal at the very free surface. Instead, C_1 formulation and ALE_1 velocity should not co-exist on extended compact portions of the very free surface: this condition is avoided by $\left|\underline{\underline{B}}_0^{-1}\right|_{thr}$ which also plays the role of a free-surface detector.
For SPH particles with negative pressure, $\delta_1\underline{u}_0$ is a destabilizing term because it has the same sign as the SPH approximation of the gradient of the unity: particles locally tend to cluster. This is another point of view to motivate the well-known SPH tensile instability. This was associated with the "p_0+p_b" formulation of the pressure-gradient term in the SPH traditional Momentum Equation (Colagrossi & Landrini, 2003, [38]). Under confined flows, the background pressure can be used to avoid negative-pressure values. Under free-surface flows, the possible occurrence of negative-pressure values is generally limited to the free-surface whose management has been already presented above.

4. SPH NON-ALE SCHEME FOR DENSE GRANULAR FLOWS AND MONO-PHASE FLOWS (NO CONSISTENCY)

This mixture scheme of SPHERA for dense granular flows (e.g., bed-load transport, fast landslides) is consistent with the "packing limit" of the Kinetic Theory of Granular Flow (KTGF) and no tuning parameter is used to represent the mixture viscosity. Boundary and sub-particle terms are omitted: they are the same ones associated with the intermediate formulation SPH-ALE_{1u}-$LC_{m,p}C_0$ (Sec.2.2.3), with the definition of the mixture quantities reported in the present section.

4.1. Mixture model for dense granular flows

This section refers to Amicarelli et al. (2017, [2]) and more recent code updates.
This SPH model represents the mixture of pure fluid and non-cohesive solid granular material, under the "packing limit" of the Kinetic Theory of Granular Flow (KTGF; Armstrong et al., 2010, [39]) for dense granular flows. This limit refers to the maximum values of the solid phase volume fraction and is peculiar of bed-load transport (e.g., erosional dam breaks) and fast landslides. The continuity equation for the fluid phase ("f") can be expressed as follows (Armstrong et al., 2010, [39]):

$$\frac{d(\rho_f \varepsilon_f)}{dt} = -\frac{\partial(\rho_f \varepsilon_f u_{f,j})}{\partial x_j} \qquad (4.1)$$

where ρ (kg×m^{-3}) is density, ε the phase volume fraction, $\underline{u}$ (m×s^{-1}) the velocity vector, t represents time and $\underline{x}$ (m) the position vector. Einstein's notation applies to the subscript "j", hereafter.
The continuity equation for the incompressible solid phase ("s") can be expressed as follows (Armstrong et al., 2010, [39]):

$$\frac{d(\rho_s \varepsilon_s)}{dt} = -\frac{\partial(\rho_s \varepsilon_s u_{s,j})}{\partial x_j} \qquad (4.2)$$

The volume balance equation assumes the following form:

$$\varepsilon_s + \varepsilon_f = 1 \qquad (4.3)$$

After defining the mixture density and velocity (the subscript "$_m$" is always omitted):

$$\rho = \rho_f \varepsilon_f + \rho_s \varepsilon_s, \quad u_i \equiv \frac{\rho_f \varepsilon_f u_{f,i} + \rho_s \varepsilon_s u_{s,i}}{\rho} \qquad (4.4)$$

the summation of (4.1) and (4.2) provides:

$$\frac{d\rho}{dt} = -\frac{\partial(\rho u_j)}{\partial x_j} \qquad (4.5)$$

The model assumes that SPH particles are conservative (i.e. mixture particles do not exchange net mass fluxes with the surrounding environment), which is a reasonable hypothesis for high solid volume fractions in saturated soils, according to the "packing limit" of the Kinetic Theory of Granular Flow (KTGF; Armstrong et al., 2010, [39]):

$$\frac{d\rho}{dt} = 0 \Rightarrow \frac{\partial u_j}{\partial x_j} = 0 \qquad (4.6)$$

Starting from (4.6), the model adopts a Weakly Compressible approach to obtain:

$$\frac{d\rho}{dt} = -\rho \frac{\partial u_j}{\partial x_j} \qquad (4.7)$$

Following the multi-phase approach of Colagrossi & Landrini (2003, [38]), the SPH approximation of (4.7) can be expressed as follows:

$$\frac{d\rho_0}{dt} = \rho_0 \sum_b \left(u_{b,j} - u_{0,j}\right) \left.\frac{\partial W}{\partial x_j}\right|_b \omega_b \qquad (4.8)$$

where $\underline{n}$ is the unit vector normal to the frontier. The subscripts "$_0$", "$_b$" and "$_w$" refer to the computational particle, a neighbouring particle and a wall frontier, respectively. The integral boundary term is computed according to Di Monaco et al. (2011, [8]) and represents the effects of wall frontiers.

This approximation of the continuity equation does not violate the mass conservation as the particle volume is undetermined.

Considering the KTGF, the momentum equation for the fluid phase can be expressed as follows (Armstrong et al., 2010, [39]):

$$\frac{d(\rho_f \varepsilon_f u_{f,i})}{dt} = -\delta_{i3} g \rho_f \varepsilon_f - \varepsilon_f \frac{\partial p_f}{\partial x_i} + \frac{\partial \tau_{f,ij}}{\partial x_j} - K_{gs}\left(u_{f,i} - u_{s,i}\right) \qquad (4.9)$$

where τ_{ij} (Pa) is the deviatoric (or shear) stress tensor, g (m×s^{-2}) gravity acceleration and p (Pa) pressure. The last term depends on the relative velocity between the phases (filtration process) through the drag coefficient K_{gs} (kg×m^{-3}×s^{-1}).

The momentum equation for the solid phase can be expressed as follows (Armstrong et al., 2010, [39]):

$$\frac{d(\rho_s \varepsilon_s u_{s,i})}{dt} = -\delta_{i3} g \rho_s \varepsilon_s - \frac{\partial p_s}{\partial x_i} + \frac{\partial \tau_{s,ij}}{\partial x_j} - \varepsilon_s \frac{\partial p_f}{\partial x_j} + K_{gs}\left(u_{f,i} - u_{s,i}\right) \qquad (4.10)$$

Provided the volume equation and the definitions of the mixture velocity and density, the sum of (4.9) and (4.10) provides:

$$\frac{d(\rho u_i)}{dt} = -\delta_{i3} g \rho - \frac{\partial p_f}{\partial x_i} - \frac{\partial p_s}{\partial x_i} + \frac{\partial \tau_{f,ij}}{\partial x_j} + \frac{\partial \tau_{s,ij}}{\partial x_j} \qquad (4.11)$$

Considering the assumption on conservative SPH particles, the shear stress gradient term of the fluid phase can be expressed as follows (Armstrong et al., 2010, [39]):

$$\frac{\partial \tau_{f,ij}}{\partial x_j} = \varepsilon_f \left[\frac{\mu_f}{3} \frac{\partial}{\partial x_i} \left(\frac{\partial u_{f,j}}{\partial x_j} \right) + \mu_f \frac{\partial^2 u_{f,i}}{\partial x_j^2} \right] \tag{4.12}$$

where μ (Pa×s) represents viscosity. Under the hypothesis of plain strain, the shear stress gradient term is represented by Schaeffer (1987, [40]; visco-plastic model for dry granular material based on internal friction), by means of a parameter, which KTFG names frictional viscosity, as described in the following.
The pressure of the solid phase is treated as follows:

$$-\frac{\partial p_{s,m}}{\partial x_i} = -\frac{\partial \sigma'_m}{\partial x_i}, \quad \sigma'_m \equiv \sum_{i=1}^{3} \frac{\sigma'_i}{3} \tag{4.13}$$

where σ'_i (Pa) are the effective stresses along the principal directions and σ'_m (Pa) is the mean effective stress (Schaeffer, 1987, [40], refers to a smoothed approximation of the 3D Mohr-Coulomb criterion; the extension to Mohr-Coulomb-Terzaghi criterion for saturated soils is straightforward). The shear stress gradient term in the momentum equation can be expressed as follows (Schaeffer, 1987, [40]):

$$\frac{\partial \tau_{s,ij}}{\partial x_j} = -k \frac{\partial}{\partial x_j} \left(\sigma'_m \left| -e_{ij} \right|^{-1} \left(-e_{ij} \right) \right), \quad e_{ij} \equiv \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$$

$$\left| e_{ij} \right| \equiv \left(\sum_{i,j} e_{ij}^2 \right)^{\frac{1}{2}} = \sqrt{2 I_2 \left(e_{ij} \right)}, \quad k \equiv \sqrt{2} \left(\sin \phi \right) \tag{4.14}$$

where φ (rad) is the internal friction angle, e_{ij} (s^{-1}) is the strain-rate tensor and $I_2(e_{ij})$ (s^{-2}) represents its quadratic invariant (formulation for incompressible fluids). One may notice that the term (4.14) is potentially unstable at high internal friction angles and that, in the "packing limit" of the KTGF, the shear stress terms of the collisional-kinetic regime are zeroed. Eq.(4.14) can be rearranged as follows:

$$\frac{\partial \tau_{s,ij}}{\partial x_j} = \sqrt{2} (\sin \varphi) \frac{\partial}{\partial x_j} \left(\frac{\sigma'_m}{\sqrt{2 I_2 (e_{ij})}} e_{ij} \right) \tag{4.15}$$

The strain-rate tensor reads:

$$e_{ij} \equiv \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{4.16}$$

whereas its quadratic invariant (in case of free divergence flows) assumes the following expression:

$$I_2 (e_{ij}) = -\frac{1}{2} \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 + \left(\frac{\partial w}{\partial z} \right)^2 \right] - \frac{1}{4} \left[\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right)^2 \right] \tag{4.17}$$

where the renormalization without boundary terms of (3.11) applies only in 2D.
The KTGF introduces the following definition for the frictional viscosity μ_{fr} (Pa×s):

$$\mu_{fr} \equiv \left(\frac{\sigma'_m (\sin \varphi)}{2 \sqrt{I_2 (e_{ij})}} \right) \tag{4.18}$$

to obtain:

$$\frac{\partial \tau_{s,ij}}{\partial x_j} = \frac{\partial}{\partial x_j} \left(2 \mu_{fr} e_{ij} \right) \tag{4.19}$$

Eq.(4.19) is consistent with internal friction, as it does not depend on the magnitude of the strain-rate tensor. In analogy to the Navier-Stokes equation (for incompressible flows), under the strong but accepted hypothesis of smooth spatial variations of μ_{fr}, Eq.(4.19) provides:

$$\frac{\partial \tau_{s,ij}}{\partial x_j} = \mu_{fr} \frac{\partial^2 u_i}{\partial x_j^2} \tag{4.20}$$

which is valid in the "packing limit" of the KTGF (i.e. for ε_s close enough to the value of 0.59, which is the maximum attainable volume fraction for a sheared inelastic hard sphere fluid, Kumaran, 2015, [41]). The model then defines the mixture total pressure p (Pa) as:

$$p = p_f + \sigma_m^{'} \tag{4.21}$$

One may consider that the mean effective stress can only be formulated under simplifying assumptions (e.g., x, y and z need to be the principal axes). Thus, $\sigma_m^{'}$ is computed as the difference between the total pressure and the fluid pressure:

$$\sigma_m^{'} = \max\left(p - \max(p_f, 0), 0\right) \tag{4.22}$$

Both fluid and solid pressures are limited to positive values as soils, which are either fully saturated or dry, do not bear tension. Considering the continuity equation, the momentum equation for the mixture can be rearranged as:

$$\frac{d(\rho u_i)}{dt} = -\delta_{i3} g \rho - \frac{\partial p_f}{\partial x_i} - \frac{\partial \sigma_m^{'}}{\partial x_i} + \varepsilon_f \mu_f \frac{\partial^2 u_{f,i}}{\partial x_j^2} + H\left(\varepsilon_s - \varepsilon_{s,p}\right) \mu_{fr} \frac{\partial^2 u_{s,i}}{\partial x_j^2} \tag{4.23}$$

where $\varepsilon_{s,p}$ = ca.0.59 and H(x) is the Heaviside step function:

$$H(x) = \frac{d}{dx} \max\{x, 0\} \tag{4.24}$$

Assuming SPH conservative particles implies that the velocity of each phase is basically equal to the one of the mixture:

$$u_{f,i} \cong u_i, \quad u_{s,i} \cong u_i \tag{4.25}$$

Considering (4.21) and the assumption of SPH conservative particles, (4.23) reduces to:

$$\frac{du_i}{dt} = -\delta_{i3} g - \frac{1}{\rho} \frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j^2} \tag{4.26}$$

where the mixture viscosity μ is finally defined as:

$$\mu \equiv \varepsilon_f \mu_f + H\left(\varepsilon_s - \varepsilon_{s,p}\right) \mu_{fr} \tag{4.27}$$

and $\nu \equiv \frac{\mu}{\rho}$ ($m^2 \times s^{-1}$) is the mixture kinematic viscosity.

Following the multi-phase approach of Colagrossi & Landrini (2003, [38]), the SPH approximation of (4.26) becomes:

$$\left\langle \frac{du_i}{dt} \right\rangle_0 = -\delta_{i3} g + \frac{1}{\rho_0} \sum_b \left(p_b + p_0\right) \left.\frac{\partial W}{\partial x_i}\right|_b \omega_b + 2\nu \sum_b \frac{m_b}{\rho_0 r_{0b}} \left(\underline{u}_b - \underline{u}_0\right) \left.\frac{\partial W}{\partial r}\right|_b \tag{4.28}$$

where r (m) is the distance between two interacting particles. The boundary value $\underline{u}_{SA}$ ($m \times s^{-1}$) of the velocity in the external portion V_h' (m^3) of the kernel support is assigned according to Di Monaco et al. (2011, [8]). So far, the last term, representing the bottom drag, has been validated in 2D.
The inner pressure gradient term in (4.28) is conservative provided that the particle mass is homogeneous because:

$$m_0 = m_b = m \Rightarrow \frac{\omega_b}{\rho_0} = \frac{m}{\rho_0 \rho_b} = \frac{\omega_0}{\rho_b} \tag{4.29}$$

The inner viscous shear stress term in (4.28) is conservative provided that the particle density and the particle kinematic viscosity are homogeneous. The inner artificial viscosity term in (4.28) is conservative provided that the particle artificial viscosity and the particle density are homogeneous.
The artificial viscosity is always activated, both for approaching and separating particles (the latter configuration was not considered in Di Monaco et al., 2011, [8]).

Despite its formulation as a mono-phase mixture, the model needs to adopt a simplified approach to represent fluid pressure in the granular material. This parameter can be related to two different soil conditions, i.e., fully saturated soil and dry soil:

$$p_f = \begin{cases} p_{f,blt-top} + \rho_f g (z_{blt-top}\big|_{x_0,y_0} - z_0)\cos^2(\alpha_{TBT}), & \textit{fully saturated soil} \\ 0, & \textit{dry soil} \end{cases} \tag{4.30}$$

where the subscript "$_{blt-top}$" refers to the top of the bed-load transport layer (or the layer of saturated material). The water content of a generic SPH mixture particle is constant all over the simulation, but saturation conditions depend on the SPH particle. This is the default Lagrangian saturation scheme. An alternative Eulerian saturation scheme, based on tuning parameters, is also available. It is described at the end of this section.
Eq.(4.30) assumes a 1D filtration flow parallel to the slope of the granular material. This simplifying hypothesis is still consistent with SPH conservative particles; α_{TBT} (rad) is the topographic angle at the top of the bed-load transport layer and lies between the local interface normal $\underline{n}_{blt-top}$ and the vertical:

$$\alpha_{TBT} = \max\left(\arccos\left(n_{blt-top,3}\right), \frac{\pi}{2}\right) \tag{4.31}$$

The angle limiter in (4.31) allows one to assign null p_f values in case of slope anomalies (very rare and unstable). The mixture pressure is computed by means of a barotropic equation of state (linearized around a reference state indicated by subscript "$_{ref}$"). A unique speed of sound can be chosen (i.e. the highest among the SPH particle values, no matter about their phase volume fractions).
To reduce the computational time and avoid the unbounded growth of (4.18), a threshold for the mixture viscosity can be defined (ν_{max}). Mixture particles with a higher viscosity are considered in the elasto-plastic regime of soil deformation and are kept fixed, whereas their pressure is derived from the mixture particles flowing above them. The threshold value is assumed to be high enough not to influence the simulation. At a fixed time step, ν_{max} does not influence even the computational time (the Courant-Friedrichs-Lewy *CFL* number criterion dominates over the viscous term criterion), if the following condition is satisfied:

$$\upsilon_{\max} \leq \frac{C_\upsilon h c}{CFL} \tag{4.32}$$

It follows that the time step duration is more probably ruled by either the CFL stability criterion, if the spatial resolution is coarse enough, or the viscous term stability criterion, if the spatial resolution is fine enough. In this case, it might be convenient to adopt a multi-step approach, where the time integration of the equations of motion for the fluid particles would be obtained with a longer time step than the one needed for the mixture particles.
No-slip conditions are suggested to be imposed on solid walls for 2D simulations at very fine spatial resolutions. The associated solid boundaries are in general in contact with the bottom of the fixed bed, so the choice of a no-slip rather than a slip condition did not play any role. On the other hand, for 3D simulations imposing free-slip conditions is suggested. In fact, the depth of the granular material layer is generally high enough and the interactions with solid walls quite exclusively concern either fluid particles at high Reynolds number or mixture particles with null velocities, so that the choice of a slip condition everywhere appeared to be an appropriate compromise. Nevertheless, no-slip conditions should be in general applied to those mobile mixture particles which are interested by a locally laminar regime. This issue, which plays a minor role in the test cases of Amicarelli et al. (2017, [2]), represents a matter of on-going developments.
The present model does not need any tuning for the mixture viscosity. The only case-dependent numerical parameters refer to the spatial resolution (*dx, h*) and possibly to *CFL* number.
The sound speed (c_{ref}) should be at least 10 times higher than the maximum velocity in the fluid (WC approach). It is sufficient to define a unique speed of sound for both mixture and pure water, as the maximum value resulting from considering all the numerical particles. The sound speed is computed by providing the bulk modulus as an input parameter for each medium.

In order to impose the initial pressure field for granular flows, a dynamic setup of hydrostatic conditions is suggested within the elasto-plastic layer because "mono-phase" hydrostatic conditions are instantaneously imposed.
One notices that $\varepsilon_{s,p}$ is used in the balance equations of this model, but the value of ε_s at the initial conditions should respect the mass conservation of the mixture, which is not necessarily under the packing limit at the beginning of the simulations. Sometimes, the void ratio e_v is available instead of the phase volume fractions. The mixture porosity is related to the void ratio by means of the following expression:

$$e_v = \frac{\varepsilon_f}{\varepsilon_s} = \frac{\varepsilon_f}{(1-\varepsilon_f)} \Rightarrow \varepsilon_f = \frac{e_v}{(1+e_v)} \tag{4.33}$$

Under these circumstances, the initial density of the solid phase is assessed as function of the mixture specific weight, the porosity, the fluid density and the gravity acceleration:

$$\begin{gathered} \gamma = g(\rho_s\varepsilon_s + \rho_f\varepsilon_f) \Rightarrow \gamma = g[\rho_s\varepsilon_s + \rho_f(1-\varepsilon_s)] \Rightarrow \\ \Rightarrow \rho_s = \frac{\gamma}{\varepsilon_s g} - \rho_f\frac{(1-\varepsilon_s)}{\varepsilon_s} = \frac{\gamma}{(1-\varepsilon_f)g} - \rho_f\frac{\varepsilon_f}{(1-\varepsilon_f)} \end{gathered} \tag{4.34}$$

Further, a criterion is applied to detect the Landslide Sliding Surfaces LSSs.
In the following, an alternative Eulerian saturation condition is reported.
Two supportive and user-defined parameters are needed: the time with minimum saturation ($t_{min,sat}$, s) and the time with maximum saturation ($t_{max,sat}$, s). They represent two arbitrary times, which identify a relative minimum and a following relative maximum in soil saturation. When $t \leq t_{min,sat}$, there is always a phreatic zone where a free surface ("FS") is detected along the vertical (and dry soil elsewhere). When $t \geq t_{min,sat}$, the saturation zones (phreatic or dry) are frozen as they were at $t_{max,sat}$. For intermediate times and zones, the infiltration front advances linearly in time. Details are provided in (28), (29) and (30). For $t \leq t_{\min,sat}$, the following zones are defined:

$$\begin{cases} phreatic: & z_{FS}|_{x_0,y_0,t} > z_{blt-top}|_{x_0,y_0} \\ dry: & z_{FS}|_{x_0,y_0,t} \quad undefined \end{cases} \tag{4.35}$$

For $t_{\min,sat} < t < t_{\max,sat}$, the following zones are defined:

$$\begin{cases} phreatic: & z_{FS}|_{x_0,y_0,t_{\min,sat}} > z_{blt-top}|_{x_0,y_0,t_{\min,sat}} \\ \inf iltration: & z_{FS}|_{x_0,y_0,t_{\min,sat}} \; undefined, \quad z_{FS}|_{x_0,y_0,t} > z_{blt-top}|_{x_0,y_0} \\ dry: & z_{FS}|_{x_0,y_0,t_{\min,sat}} \; undefined, \quad z_{FS}|_{x_0,y_0,t} \; undefined \end{cases} \tag{4.36}$$

For $t_{\max,sat} \leq t$, the following zones are defined:

$$\begin{cases} phreatic: & z_{FS}|_{x_0,y_0,t_{\max,sat}} > z_{blt-top}|_{x_0,y_0} \\ dry: & z_{FS}|_{x_0,y_0,t_{\max,sat}} \quad undefined \end{cases} \tag{4.37}$$

In the phreatic zone the fluid pressure is expressed by (4.30), whereas in the infiltration zone it reads:

$$p_f = \begin{cases} p_{f,blt-top}\left(1 - \dfrac{(z_{blt-top}|_{x_0,y_0} - z_0)}{(z_{blt-top}|_{x_0,y_0} - z_{\inf}|_{x_0,y_0})}\right), z_{\inf}|_{x_0,y_0} \leq z_0 \leq z_{blt-top}|_{x_0,y_0} \\ 0, \quad z_{\inf}|_{x_0,y_0} \geq z_0 \end{cases} \tag{4.38}$$

when the height of the infiltration front is linear in time:

$$z_{\inf}|_{x_0,y_0} = z_{blt-top}|_{x_0,y_0} + -\frac{(t - t_{\min,sat})}{(t_{\max,sat} - t_{\min,sat})}(z_{blt-top}|_{x_0,y_0} - z_{blt-bot}|_{x_0,y_0}) \tag{4.39}$$

4.2. 2-interface 3D erosion criterion

A 2-interface 3D erosion criterion is implemented to speed-up the computational velocity of the model for bed-load transport (Sec.4.1), if the erosion is the only cause of mobilization of the solid grains. The erosion criterion aims to select those mixture particles, which needs the bed-load transport model to be applied.

The main erosion scheme is the 1-interface ("pure fluid - fixed bed") 2D erosion criterion of Manenti et al. (2012, [11]), based on the formulation of Shields - van Rijn. Two modifications to this scheme are integrated: the extension to the third dimension and the treatment of a second interface ("bed-load transport layer - fixed bed").
The erosion criterion refers to the interaction of a generic fixed mixture particle and the fluid flow above (pure fluid or mixture). Its reference parameters are represented by the closest mobile particle (of mixture or pure fluid) above the fixed particle. In any case, the interactions with the pure fluid are privileged, if available. The formulation of van Rijn (1993, [42]) reads:

$$\vartheta_c = \begin{cases} 0.1, & \mathrm{Re}_* \leq 1 \\ 0.010595\ln(\mathrm{Re}_*) + \dfrac{0.110476}{\mathrm{Re}_*} + 0.0027197, & 1 \leq \mathrm{Re}_* \leq 500 \\ 0.068, & \mathrm{Re}_* > 500 \end{cases} \tag{4.40}$$

where θ_c is Shields parameter and Re_* is the grain Reynolds number:

$$\mathrm{Re}_* \equiv \frac{k_s u_*}{\upsilon_f} \tag{4.41}$$

where k_s is a roughness length scale with $k_s=3d_{50}$ as an approximation for $k_s=3d_{90}$. It is a local erosion criterion, thus roughness is computed locally, instead of adopting a formulation for bed forms (van Rijn, 1982, [43]).
The assessment of the friction velocity (u_*) follows the procedure below.
If the reference height of the fluid (z) belongs to the Surface Neutral Boundary Layer (SNBL), the model computes the roughness coefficient z_0, according to the formula of Manenti et al. (2012, [11]) and those associated to the similarity theory or the SNBL:

$$u_* = \frac{k_v U}{\ln\left(z / z_0 \right)}, \quad z_0 = 0.11\frac{\nu_f}{u_*} + \frac{3d_{50}}{30} \tag{4.42}$$

where k_v is von Karman constant and U is the flow velocity at the reference height.
If z refers to the SNBL, the model considers the velocity profile of the Sub-Viscous Layer, with a direct estimation of the friction velocity:

$$u_* = \sqrt{\frac{U\nu_f}{z}} \tag{4.43}$$

In this case, U can be smaller than u_*. This usually happens at the lower interface ("bed-load transport layer - fixed bed").
In synthesis, the model estimates u_* (by means of an iterative procedure if z refers to the SNBL -u_* depends on z_0, which is in turn function of u_*-), then Re_* and θ_c. Shield parameter is computed:

$$\vartheta \equiv \frac{\tau_*}{g(\rho_s - \rho_f)d_{50}}, \qquad \tau_* \equiv \rho_f u_*^2 \tag{4.44}$$

and compared with θ_c. The erosion criterion is satisfied if $\theta \geq \theta_c$.
In practise, Shields criterion is derived under 1D stationary and uniform conditions, and does not explicitly depend on the friction angle. This is explicitly taken into account to quantify the effects of the fixed bed slope, as explained in the following.
The 2D erosion criterion for horizontal beds can be extended to 3D generic slopes, by means of the coefficient $k_{\beta\gamma}$, which is defined as follows:

$$\vartheta_{c,\beta\gamma} = k_{\beta\gamma}\vartheta_{c,00}, \quad 0 \leq k_{\beta\gamma} \leq k_{\beta 0} \tag{4.45}$$

$k_{\beta\gamma}$ is always non-negative and smaller than (or equal to) its 2D value $k_{\beta 0}$ (γ=0). In fact, if the slope angle transversal to the main flow direction (β) is not null, erosion is enhanced. Further, in the presence of a bed with a locally ascendant slope (β<0), $k_{\beta\gamma}$ can be higher than the unity. In this case, (4.45) can possibly provide a second non-physical solution, with $k_{\beta\gamma}$<1, which is not taken into account because it corresponds to a flow with an inverted direction.

The normal at the interface "bed-load transport - fixed bed" is defined by a means of a normalized SPH approximation of the relative distance between the mobile sub-domain and the generic SPH particle of the fixed bed:

$$\underline{n}_{\text{int},0} = \frac{\sum_{bf} \left(\underline{x}_{bf} - \underline{x}_0\right) W_{bf}\,\omega_{bf}}{\left|\sum_{bf} \left(\underline{x}_{bf} - \underline{x}_0\right) W_{bf}\,\omega_{bf}\right|} \tag{4.46}$$

In the absence of a free surface, the normal is aligned with gravity, by definition.
The main slope angle quantifies the slope of the fixed bed in the direction of the main flow. Assuming that, close to the interface, the mixture velocity is parallel to the fixed bed, β only depends on the direction of the velocity vector of the closest particle (3D definition):

$$\beta = \arcsin\left(-u_{b,3}\right) \tag{4.47}$$

In 2D, one could alternatively define β as function of the velocity direction or the interface normal. The latter assumption reduces the model errors and is used in 2D:

$$\beta = -\left[\frac{\pi}{2} - \arcsin\left(n_{\text{int},0,z}\right)\right] sign\left(u_{f,s,3}\right), \quad u_{f,s} = \underline{u}_f - \underline{u}_f \cdot \underline{n}_{\text{int}} \tag{4.48}$$

The transversal slope angle γ is defined as:

$$\gamma = \arcsin\left|n_{2,z}\right|, \quad \underline{n}_2 = \underline{n}_{\text{int}} \times \underline{u}_f \tag{4.49}$$

The unity vector $\underline{n}_2$ represents the bi-normal to the fluid particle trajectory and is independent on the sign of γ.
The value of $k_{\beta\gamma}$ is a solution of the quadratic equation of Seminara et al. (2002, [44]):

$$\begin{gathered} a k_{\beta\gamma}^2 + b k_{\beta\gamma} + c = 0 \quad \Rightarrow \quad k_{\beta\gamma} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \\ a \equiv 1 - \Delta, \quad b \equiv 2\left\{\frac{\Delta}{\sqrt{1 + \left(\tan\beta\right)^2 \left(\tan\gamma\right)^2}} + \frac{\sin\beta}{\tan\varphi}\right\} \\ c \equiv \frac{1+\Delta}{1 + \left(\tan\beta\right)^2\left(\tan\gamma\right)^2}\left(-1 + \frac{\left(\tan\beta\right)^2\left(\tan\gamma\right)^2}{\left(\tan\varphi\right)^2}\right) \\ \Delta \equiv \frac{4}{3}\tan\varphi \frac{C_L}{C_D}\frac{u_* d_{50}}{k_v U\left(z - z_{\text{int}}\right)} \end{gathered} \tag{4.50}$$

In the presence of two admissible roots, the model chooses the closest to $k_{\beta 0}$, provided $k_{\beta\gamma} \leq k_{\beta 0}$; in the absence of roots, the model assumes $k_{\beta\gamma} = k_{\beta 0}$.
The drag coefficient C_D is approximated by the formula of Morrison (2013, [45]) for a fluid flow around a sphere:

$$C_D = \begin{cases} 1.0, & \text{Re} < 1.0 \cdot 10^2 \\ \dfrac{24}{\text{Re}} + \dfrac{\frac{2.6\,\text{Re}}{5}}{1 + \left(\frac{\text{Re}}{5}\right)^{1.52}} + \dfrac{0.411\left(\frac{\text{Re}}{263000}\right)^{-7.94}}{1 + \left(\frac{\text{Re}}{263000}\right)^{-8}} + \dfrac{\text{Re}^{0.8}}{461000}, & 1.0 \cdot 10^2 \leq \text{Re} \leq 1.0 \cdot 10^6 \\ 0.2, & \text{Re} > 1.0 \cdot 10^6 \end{cases} \tag{4.51}$$

with C_D varying between 0.1 and 1. Reynolds number is here defined as follows:

$$\text{Re} \equiv \frac{U_\infty d_{50}}{\nu} \tag{4.52}$$

with U_∞ equal to the absolute value of velocity at the closest particle and d_{50} representing the 50-th percentile of the particle-size distribution of the soil.

In this context, the lift is assumes the form:

$$F_L = \frac{4}{3}\rho C_L \frac{\pi D^3}{8} U \frac{u_*}{k_v\left(z - z_{\text{int}}\right)} \tag{4.53}$$

where z_{int} is the interface height. A formula for the lift coefficient is derived, by interpolating the experimental data of Seminara et al. (2002, [44]):

$$C_L = \begin{cases} 0.07, \quad \text{Re} < 2.0 \cdot 10^3 \\ \left(9.0 \cdot 10^{-5}\right)\text{Re}^{0.882}, \quad 2.0 \cdot 10^3 \le \text{Re} \le 1.75 \cdot 10^4 \\ 0.5, \quad \text{Re} > 1.75 \cdot 10^4 \end{cases} \tag{4.54}$$

with C_L varying between 0.07 and 0.5.
The mixture pressure of a generic fixed SPH particle is computed, after assuming hydrostatic conditions within the fixed bed:

$$p_0 = p_b + \left(z_b - z_0\right)\gamma_m + \frac{1}{2}\rho U_{nor,b}^2, \quad U_{nor} = \left|\left(\underline{u}_b - \underline{u}_0\right)\cdot\left(\underline{x}_b - \underline{x}_0\right)\right| \tag{4.55}$$

Provided the absence of a fixed bed along the vertical and the simultaneous presence of fixed particles (or frontiers) within the kernel support, the mixture SPH particle is held fixed.

4.3. A simplified approach for soil liquefaction

This section refers to Amicarelli et al. (2016, [46]).
A simplified approach is implemented for soil liquefaction, by means of linearized pore pressure functions, depending on the critical number N of equivalent uniform cycles:

$$\frac{p_f\left(\underline{x},t\right) - p_{f,nq}\left(\underline{x},t\right)}{p_{nq}\left(\underline{x},t\right) - p_{f,nq}\left(\underline{x},t\right)} = f_{liq}\left(\frac{N(t)}{N_L}\right) \tag{4.56}$$

where the subscript "$_{nq}$" represents "no-quake" conditions, N_L the critical value of N (when liquefaction occurs) and N/N_L is named cyclic number ratio.
In case the liquefaction time (t_{liq}) is known and f_{liq} is approximately linear, (4.56) becomes:

$$\sigma_m^{'}\left(\underline{x},t\right) = \begin{cases} \sigma_{m,nq}^{'}\left(\underline{x},t\right), & t \le t_{q0} \\ \sigma_{m,nq}^{'}\left(\underline{x},t\right)\left[1 - \frac{\left(t - t_{q0}\right)}{\left(t_{liq} - t_{q0}\right)}\right], & t_{q0} < t \le t_{liq} \\ 0, & t_{liq} < t \le t_{qf} \\ \sigma_{m,nq}^{'}\left(\underline{x},t\right), & t > t_{qf} \end{cases} \tag{4.57}$$

where t_{q0} and t_{qf} are the quake starting and ending times.

4.4. Mono-phase formulation

In case the solid volume fraction is null, the multi-phase scheme for dense granular flows degenerates in the non-consistent mono-phase formulation of SPHERA.

4.5. Time integration schemes for the non-consistent SPH formulation

In case of non-consistent multi-phase or mono-phase formulations, time integration for fluid and solid body particles is ruled by a second-order Leapfrog scheme, as described in Amicarelli et al. (2015, [4]) and Di Monaco et al. (2011, [8]):

$$\begin{gathered} x_i\Big|_{t+dt} = x_i\Big|_t + u_i\Big|_{t+dt/2} dt, \quad i = 1,2,3 \\ u_i\Big|_{t+dt/2} = u_i\Big|_{t-dt/2} + \left\langle \frac{du_i}{dt} \right\rangle\Big|_t dt, \quad i = 1,2,3 \\ \rho\Big|_{t+dt} = \rho\Big|_t + \left\langle \frac{d\rho}{dt} \right\rangle\Big|_{t+dt/2} dt \end{gathered} \tag{4.58}$$

Two alternative explicit Runge-Kutta time integration schemes are also implemented: Euler scheme (RK1; first order) and Heun scheme (RK2, second-order).
According to RK1, the generic parameter f is integrated as follows:

$$f(t_0+dt)=f(t_0)+f'(t_0)dt(t_0),\ \frac{df}{dt}\equiv f' \tag{4.59}$$

The scheme above can be rearranged in the following form:

$$f_{RK1,i+1}=f_i+f'_i\ dt_i \tag{4.60}$$

where the subscripts here represent the time step ID.
RK2 assumes the following form:

$$\begin{aligned} f_{RK2,i+1} &= f_i+\frac{dt}{2}\{f'(t_i,f_i)+f'\{t_{i+1},[f_i+f'(t_i,f_i)dt_i]\}\}= \\ &= f_i+\frac{dt}{2}\left[f'(t_i,f_i)+f'(t_{i+1},f_{RK1,i+1})\right] \end{aligned} \tag{4.61}$$

This 2-stage formulation implies 2 stages (sub-loops) for each time step. During the first stage, the temporary value $f_{RK1,i+1}$ is computed. During the second stage, the time step value $f_{RK2,i+1}$ is assessed. However, several procedures do not need a double loop (e.g., the neighbouring search algorithm, the estimation of the time step duration, the inlet/outlet section management, the result printing, the erosion criterion).

5. BOUNDARY TREATMENT SCHEME FOR FIXED SURFACE WALLS: THE SEMI-ANALYTICAL APPROACH (SASPH)

5.1. The Semi-Analytical approach

The present section refers to Di Monaco et al. (2011, [8]).
Consider the discretization as provided by the SPH approximation of the first derivative of a generic function (*f*), according to the semi-analytical approach ("SA") defined in Vila (1999, [47]) and later developed in Di Monaco et al. (2011, [8]):

$$\left\langle\frac{\partial f}{\partial x_i}\right\rangle_{0,IN+BC,SA}=-\sum_b(f_b-f_0)\frac{\partial W_b}{\partial x_i}\omega_b-\int_{V_h'}(f-f_0)\frac{\partial W}{\partial x_i}dx^3 \tag{5.1}$$

The inner fluid domain here involved is filled with numerical particles. At boundaries, the kernel support is (formally) not truncated because it can partially lie outside the fluid domain. The summation in (5.1) is performed over all the fluid particles "$_b$" (neighbouring particles with volume ω) in the kernel support of the computational fluid particle ("$_0$"). At the same time, the volume integral in (5.1) represents the boundary term, which is a convolution integral on the truncated portion of the kernel support. In this fictitious and outer volume (V_h'), one needs to define the generic function f (pressure, velocity or density alternatively).
The semi-analytical approach ("$_{SA}$") (the version of Di Monaco et al., 2011, [8]) hypothesizes the following linearization and assumptions to compute f in V_h':

$$f\cong f_{SA}+\left.\frac{\partial f}{\partial x_i}\right|_{SA}(\underline{x}-\underline{x}_0)\Rightarrow\left\langle\frac{\partial f}{\partial x_i}\right\rangle_{BC,SA}=-\int_{V_h'}(f_{SA}-f_0)\frac{\partial W}{\partial x_i}dx^3-\int_{V_h'}\left.\frac{\partial f}{\partial x_i}\right|_{SA}(\underline{x}-\underline{x}_0)\frac{\partial W}{\partial x_i}dx^3 \tag{5.2}$$

The peculiar "$_{SA}$" values of the functions and derivatives within V_h' are assigned to represent a null normal gradient of reduced pressure at the frontier interface (while considering uniform density):

$$p_{SA}=p_0,\ \left\langle\frac{\partial p}{\partial x_i}\right\rangle_{SA}=-\delta_{i3}g,\ \ \rho_{SA}=\rho_0,\ \left\langle\frac{\partial \rho}{\partial x_i}\right\rangle_{SA}=0 \tag{5.3}$$

The integral computation considers the solid angle (α_s) as the integration variable. The functions g_1 (m^{-1}) and g_2 are summations (over the surface boundary elements intercepting the truncated kernel support) of summations (over *r*, distance between "$_0$" and each external discretized volume) of analytical functions depending on *W* and r_e (s). This is the distance between the computational particle and the point on the line passing for "$_0$" and the point on the frontier described by the local solid angle. Eq.(5.2) is rearranged as follows:

$$\left\langle \frac{\partial f}{\partial x_i} \right\rangle_{BC,SA} = (f_{SA} - f_0) \int_{V_e} g_1 (W, r_e, \underline{n}) d\alpha + \left. \frac{\partial f}{\partial x_i} \right|_{SA} \int_{V_e} g_2 (W, r_e, \underline{n}) d\alpha \tag{5.4}$$

The two integrals in (5.4) are discretized using 64 integration nodes (16 meridians and 4 sectors):

$$\Delta\alpha_s = \frac{\Delta\theta (2h)(\cos\varphi_z)\Delta z}{(2h)^2},$$
$$\Delta z = (2h)\left[\sin\left(\varphi_z + \frac{\Delta\varphi_z}{2}\right) - \sin\left(\varphi_z - \frac{\Delta\varphi_z}{2}\right) \right] = 2(2h)\sin\left(\frac{\Delta\varphi_z}{2}\right) \Rightarrow \tag{5.5}$$
$$\Rightarrow \Delta\alpha_s = 2\Delta\theta\cos\varphi_z \sin\left(\frac{\Delta\varphi_z}{2}\right)$$

where φ_z is the zenith angle and θ is the azimuth angle within the spheric kernel support. Considering this discretization, Eq.(5.4) is further rearranged as follows:

$$\left\langle \frac{\partial f}{\partial x_i} \right\rangle_{BC,SA} = (f_{SA} - f_0) \sum_s g_{1,s} (W, r_e, \underline{n}) \Delta\alpha_s + \left. \frac{\partial f}{\partial x_i} \right|_{SA} \sum_s g_{2,s} (W, r_e, \underline{n}) \Delta\alpha_s \tag{5.6}$$

The velocity vector is taken as uniform in the outer part of the kernel support. Here $\underline{u}_{SA}$ is decomposed in the sum of a vector normal to boundary $(\underline{u}_{SA,n})$ and a tangential vector $(\underline{u}_{SA,T})$.

Under free-slip conditions, the tangential component of the velocity vector in the truncated portion of the kernel support is:

$$\underline{u}_{SA,T} = \underline{u}_{0,T} \tag{5.7}$$

whereas it assumes the following expression under no-slip conditions:

$$\underline{u}_{SA,T} = \underline{u}_{w,T} - (\underline{u}_{0,T} - \underline{u}_{w,T}) = 2\underline{u}_{w,T} - \underline{u}_{0,T} = \left[(2\underline{u}_w - \underline{u}_0)\cdot \hat{\underline{t}}_w\right]\hat{\underline{t}}_w \tag{5.8}$$

where $\underline{t}$ is the unit vector aligned with the tangential component of the fluid velocity (with respect to the wall).

The normal component of the velocity vector in the truncated portion of the kernel support, no matter about the slip conditions, reads:

$$\underline{u}_{SA,n} = \underline{u}_{w,n} - (\underline{u}_{0,n} - \underline{u}_{w,n}) = 2\underline{u}_{w,n} - \underline{u}_{0,n} = \left[(2\underline{u}_w - \underline{u}_0)\cdot \hat{\underline{n}}_w\right]\hat{\underline{n}}_w \tag{5.9}$$

where $\underline{n}_w$ is the normal vector of the wall surface, as defined by its local orientation.

Thus, the associated velocity vectors can be assigned to the truncated portion of the kernel support both under no-slip conditions:

$$\underline{u}_{SA} = \underline{u}_{SA,n} + \underline{u}_{SA,T} = \left[(2\underline{u}_w - \underline{u}_0)\cdot \hat{\underline{n}}_w\right]\hat{\underline{n}}_w + \left[(2\underline{u}_w - \underline{u}_0)\cdot \hat{\underline{t}}_w\right]\hat{\underline{t}}_w = 2\underline{u}_w - \underline{u}_0 \tag{5.10}$$

and free-slip conditions:

$$\underline{u}_{SA} = \underline{u}_{SA,n} + \underline{u}_{SA,T} = \left[(2\underline{u}_w - \underline{u}_0)\cdot \hat{\underline{n}}_w\right]\hat{\underline{n}}_w + \left[\underline{u}_0 \cdot \hat{\underline{t}}_w\right]\hat{\underline{t}}_w \tag{5.11}$$

together with the following velocity differences, both under free-slip conditions:

$$\underline{u}_{SA} - \underline{u}_0 = \underline{u}_{SA,n} + \underline{u}_{SA,T} = 2\left[(\underline{u}_w - \underline{u}_0)\cdot \hat{\underline{n}}_w\right]\hat{\underline{n}}_w \tag{5.12}$$

and no-slip conditions:

$$\underline{u}_{SA} - \underline{u}_0 = 2(\underline{u}_w - \underline{u}_0) \tag{5.13}$$

5.2. SASPH boundary contribution to the pressure-gradient term in the Momentum Equation

The present section refers to Di Monaco et al. (2011, [8]).

The SASPH "$_{SA}$" boundary contribution to the pressure-gradient term in the 3D Momentum Equation was presented in Di Monaco et al. (2011, [8]):

$$BC_{ME,p,SA} = \int_{V_h'} \rho_{SA} \left(\frac{p_{SA}}{\rho_{SA}^2} + \frac{p_0}{\rho_0^2} \right) \underline{\nabla} W dx^3 \tag{5.14}$$

After defining the SASPH boundary pressure and density values as follows (Di Monaco et al., 2011, [8]):

$$p_{SA} = p_0 + \rho r \left(\underline{\nabla} r \cdot \underline{g} \right), \quad \rho_{SA} = \rho_0 \tag{5.15}$$

and considering (2.83), the boundary term (5.14) reads (Di Monaco et al., 2011, [8]):

$$BC_{ME,p,SA} = 2 \frac{p_0}{\rho_0} \int_{V_h'} \underline{\nabla} W dx^3 + \int_{V_h'} r \left(\underline{\nabla} r \cdot \underline{g} \right) \underline{\nabla} W dx^3 \tag{5.16}$$

The integral of the kernel gradient over the truncated portion of the kernel support is expressed in local spherical coordinates, depending on the solid angle α_s. When applied to a generic wall element "w", the integral above is partitioned in a summation of integrals over discrete infinitesimal spherical volumes (Di Monaco et al., 2011, [8]):

$$\begin{aligned} BC_{ME,p,SA} = {} & 2 \frac{p_0}{\rho_0} \sum_w \left\{ \underline{\underline{T}}_w \left[\int_{\alpha_s,w} \left(\int_{r_w(\vartheta,\varphi_z)}^{2h} \underline{\nabla} W r^2 dr \right) d\alpha_s \right] \right\} + \\ & + \sum_w \left\{ \underline{\underline{T}}_w \left\{ \int_{\alpha_s,w} \left[\underline{\nabla} r \cdot \left(\underline{\underline{T}}_w^T \underline{g} \right) \right] \left(\int_{r_w(\vartheta,\varphi_z)}^{2h} r^3 \underline{\nabla} W dr \right) d\alpha_s \right\} \right\} \end{aligned} \tag{5.17}$$

where r (m) is the distance of the barycentre of the spherical infinitesimal volume from the computational particle, θ and φ_z are the azimuthal and zenith angles. $\underline{\underline{T}}_w$ is the matrix of directional cosines to change from the local reference system of the wall element to the global reference system. The transposed matrix of $\underline{\underline{T}}_w$ is equal to its inverse matrix and permits the inverse change of reference system. This formula is simplified by isolating the integrals over the distance and recalling (2.83):

$$\begin{aligned} & BC_{ME,p,SA} = 2 \frac{p_0}{\rho_0} \sum_w \left\{ \underline{\underline{T}}_w \left[\int_{\alpha_s,w} \left(\underline{\nabla} r \right) J_{2,w} d\alpha_s \right] \right\} + \sum_w \left\{ \underline{\underline{T}}_w \left\{ \int_{\alpha_s,w} \underline{\nabla} r \left[\underline{\nabla} r \cdot \left(\underline{\underline{T}}_w^T \underline{g} \right) \right] J_{3,w} d\alpha_s \right\} \right\}, \\ & J_{2,w} \equiv \left(\int_{r_w(\vartheta,\varphi_z)}^{2h} \frac{dW}{dr} r^2 dr \right), \quad J_{3,w} \equiv \int_{r_w(\vartheta,\varphi_z)}^{2h} r^3 \frac{dW}{dr} dr \end{aligned} \tag{5.18}$$

The following equality is valid for any triplet of vectors v_i:

$$\left[\left(\underline{v}_1 \otimes \underline{v}_2 \right) \underline{v}_3 \right] = \begin{bmatrix} v_{1,x} v_{2,x} & v_{1,x} v_{2,y} & v_{1,x} v_{2,z} \\ v_{1,y} v_{2,x} & v_{1,y} v_{2,y} & v_{1,y} v_{2,z} \\ v_{1,z} v_{2,x} & v_{1,z} v_{2,y} & v_{1,z} v_{2,z} \end{bmatrix} \begin{Bmatrix} v_{3,x} \\ v_{3,y} \\ v_{3,z} \end{Bmatrix} = \underline{v}_1 \left(\underline{v}_2 \cdot \underline{v}_3 \right) \tag{5.19}$$

and is here applied to the following multiple product:

$$\underline{\nabla} r \left[\underline{\nabla} r \cdot \left(\underline{\underline{T}}_w^T \underline{g} \right) \right] = \left[\underline{\nabla} r \otimes \underline{\nabla} r \right] \left(\underline{\underline{T}}_w^T \underline{g} \right) \tag{5.20}$$

Introducing (5.20) in (5.18), the SASPH boundary contribution to the pressure-gradient term in the Momentum Equation finally assumes the following expression (Di Monaco et al., 2011, [8]):

$$BC_{ME,p,SA} = 2 \frac{p_0}{\rho_0} \sum_w \left\{ \underline{\underline{T}}_w \left[\int_{\alpha_s,w} \left(\underline{\nabla} r \right) J_{2,w} d\alpha_s \right] \right\} + \sum_w \left\{ \underline{\underline{T}}_w \left\{ \left[\int_{\alpha_s,w} \left(\underline{\nabla} r \otimes \underline{\nabla} r \right) J_{3,w} d\alpha_s \right] \left(\underline{\underline{T}}_w^T \underline{g} \right) \right\} \right\} \tag{5.21}$$

The closed-form solutions for $J_{2,w}$ (m^{-1}) and $J_{3,w}$ were presented in Di Monaco et al. (2011, [8]):

$$J_{2,w} = \frac{1}{\pi h}\begin{cases} -\frac{7}{10}+\frac{3}{4}q^4-\frac{9}{20}q^5, & 0 \le q < 1 \\ -\frac{4}{5}+q^3-\frac{3}{4}q^4+\frac{3}{20}q^5, & 1 \le q < 2, \\ 0, \quad q \ge 2, \quad q \equiv \frac{r}{h} \end{cases}$$

$$J_{3,w} = \frac{1}{\pi}\begin{cases} -\frac{3}{4}+\frac{3}{5}q^5-\frac{3}{8}q^6, & 0 \le q < 1 \\ -\frac{4}{5}+\frac{3}{4}q^4-\frac{3}{5}q^5+\frac{1}{8}q^6, & 1 \le q < 2, \\ 0, \quad q \ge 2, \quad q \equiv \frac{r}{h} \end{cases} \tag{5.22}$$

The 2D SASPH contribution to the pressure-gradient term in the Momentum Equation is a 1D term integrated along the kernel truncated boundary, which is composed by the solid segments of the SASPH walls (Di Monaco et al., 2011, [8]):

$$BC_{ME,p,SA,2D} = -\left[\underline{\underline{I}} - \underline{\underline{R}}_s \int_{A_h'} W dA\right]^{-1}\left[-\underline{n}_w\left(\frac{p_{SA}+p_0}{\rho_0}\right)\int_{P_{1,w}}^{P_{2,w}} W ds + \underline{\underline{R}}_n \underline{g}\right],$$

$$p_{SA,2D} = p_0 + \rho_0 \frac{\left(r_{P_2}+r_{P_1}\right)}{2}\left(\underline{n}_w \cdot \underline{g}\right), \quad R_{n,ij} = n_{w,i}n_{w,j}, \quad R_{s,ij} = s_{w,i}s_{w,j} \tag{5.23}$$

where $\underline{s}_w$ is the unit vector tangential to the wall element "$_w$", A_h' (m^2) is the truncated portion of the kernel support and each solid segment is delimited by the points $P_{1,w}$ and $P_{2,w}$.

5.3. SASPH boundary contribution to the velocity-divergence term in the Continuity Equation

The present section refers to Di Monaco et al. (2011, [8]).

The SASPH "$_{SA}$" boundary term for the 3D Continuity Equation was presented in Di Monaco et al. (2011, [8]):

$$BC_{CE,SA} = 2\rho_0 \int_{V_h'} \left(\underline{u}_{SA} - \underline{u}_0\right)\cdot \underline{\nabla} W dx^3 = 2\rho_0 \int_{V_h'} \left[\left(\underline{u}_w - \underline{u}_0\right)\cdot \underline{n}_w\right]\underline{n}_w \cdot \underline{\nabla} W dx^3 \tag{5.24}$$

Fixed a wall element "$_w$", the integral of the kernel gradient over the truncated portion of the kernel support is expressed in local spherical coordinates (depending on the solid angle α_s) and is partitioned in a summation of integrals over discrete infinitesimal spherical volumes (Di Monaco et al., 2011, [8]):

$$\int_{V_h'} \underline{\nabla} W dx^3 = \underline{\underline{T}}_w \left[\int_{\alpha_s, w}\left(\int_{r_w(\vartheta,\varphi_z)}^{2h} \underline{\nabla} W r^2 dr\right) d\alpha_s\right] \tag{5.25}$$

where $\underline{\underline{T}}_w$ is the matrix of directional cosines between the global reference system and the local reference system of the wall element, r (m) is here the distance of the barycentre of the spherical infinitesimal volume from the computational particle, θ and φ_z are the azimuthal and zenith angles. After recalling (2.83), the SASPH boundary term for the Continuity Equation involves a summation over all the neighbouring SASPH walls (Di Monaco et al., 2011, [8]):

$$BC_{CE,SA} = 2\rho_0 \sum_w \left\{\left[\left(\underline{u}_w - \underline{u}_0\right)\cdot \underline{n}_w\right]\underline{n}_w \cdot \left[\underline{\underline{T}}_w\left(\int_{\alpha_s,w} \underline{\nabla} r J_{2,w} d\alpha_s\right)\right]\right\}, \quad J_{2,w} \equiv \left(\int_{r_w(\vartheta,\varphi_z)}^{2h} \frac{dW}{dr} r^2 dr\right) \tag{5.26}$$

This 3D boundary term refers to free-slip conditions and the beta-spline cubic kernel.

SASPH represents fixed surface frontiers: the wall velocity $\underline{u}_w$ is always null.

The 2D SASPH term for the Continuity Equation is a 1D term integrated along the kernel truncated boundary, which is composed by the solid segments of the SASPH walls:

$$BC_{CE,SA,2D} = 2\rho_0 \sum_w \left\{ \left[\left(\underline{u}_w - \underline{u}_0 \right) \cdot \underline{n}_w \right] \int_{P_{1,w}}^{P_{2,w}} W ds \right\} \tag{5.27}$$

where each solid segment is delimited by the points *P*$P_{1,w}$ and $P_{2,w}$. This 2D boundary term refers to no-slip conditions and the beta-spline cubic kernel.

5.4. SASPH elastic-reaction boundary term

The SASPH boundary contribution to the pressure-gradient term in the Momentum Equation might activate an elastic reaction boundary term, which introduces a further stability criterion for the assessment of the time step. Further details are available in Di Monaco et al. (2011, [8] , eq.35).

5.5. SASPH boundary contribution to the viscous term of the Momentum Equation

The SASPH boundary contribution to the viscous term of the Momentum Equation reads (Di Monaco et al., 2011, [8]):

$$2\nu \left(u_{w,i} - u_{0,i} \right) \cdot \left(\int_{V_h'} \frac{1}{r_{0w}} \frac{\partial W}{\partial r} dx^3 \right) \tag{5.28}$$

5.6. SASPH boundary contribution to the sub-particle term of the Momentum Equation

The SASPH boundary contribution to the sub-particle term of the Momentum Equation reads (Di Monaco et al., 2011, [8]):

$$-\nu_M \left(\underline{u}_{SA} - \underline{u}_0 \right) \cdot \left(\int_{V_h'} \frac{1}{r_{0w}^2} \left(\underline{x} - \underline{x}_0 \right) \frac{\partial W}{\partial x_i} dx^3 \right) \tag{5.29}$$

5.7. Wall drag and wall-function similarity theory: adaptation to flash floods

The present section is reported as a preprint of Amicarelli et al. (2022, [36]).
The formulation for the slip coefficient presented in this section is an alternative to imposing the free-slip or no-slip conditions of Sec.5.1.
SPHERA simulates the solid walls impacted by a flash flood in terms of "explicit roughness" and "sub-grid roughness". The first is integrated in the topographic surface as explicit solid obstacles with fluid-structure interactions. The latter is not explicitly reproduced, but its effects are considered by the wall-function similarity theory, as reported in this section. A formulation for the wall shear stress τ_w (Pa) locally exerted by the sub-grid roughness to a flash flood is adapted from the wall-function theory. Here the wall shear stress is equivalent to the wall drag per unit of surface as the formulation of the current section applies to the sub-grid roughness. The representation of the wall shear stress (Sec.5.7.1) for the Neutral Surface BL (NSBL) involves the turbulent viscosity (Sec.5.7.2), the slip coefficient (Sec.5.7.3) and the roughness characteristic length. The latter is provided with a formulation for the rough-wall regime of flash floods (5.7.5). This accounts for the CORINE Land Cover, Digital Elevation Models (DEMs) and Digital Terrain Models (DTMs).

5.7.1. Wall shear stress

The present section is reported as a preprint of Amicarelli et al. (2022, [36]).
The SASPH boundary term in the momentum equation of SPHERA involves a shear-stress gradient boundary term $BC_{ME,F,\tau}$ (m/s^2) presented in Di Monaco et al. (2011, [8]) and validated in following studies. Fixed a SPH computational particle close to a wall, the term $BC_{ME,F,\tau}$ depends on the dynamic viscosity, a fictitious boundary velocity, the wall geometry, the velocity and position of the computational particle. The above viscosity was either the mixture viscosity of dense granular flows (Amicarelli et al., 2017, [48]) or the molecular viscosity of mono-phase fluids under laminar regimes

(Paggi et al., 2021, [49]). In case of flash floods, the viscosity included in $BC_{ME,F,\tau}$ has to be represented by the turbulent viscosity μ_T (Pa×s), since molecular viscosity is negligible in the SBL and here floods are mono-phase flows. Thus, a simple formulation for μ_T in the NSBL is adopted (Sec.5.7.2).
For each particle-wall interaction, a fictitious boundary velocity is defined outside the fluid domain, within the truncated portion of the SPH kernel support, namely the influence region of the SPH particle. This velocity is assigned by means of the slip coefficient ϕ_S, which was previously treated as an input parameter to represent either free-slip conditions or no-slip conditions. Here a formulation for the slip coefficient of computational particles in the NSBL is presented (Sec.5.7.3).
Both the formulations for μ_T and ϕ_s are consistent with the wall-function similarity theory.

5.7.2. Turbulent viscosity for the Neutral Surface Boundary Layer

The present section is reported as a preprint of Amicarelli et al. (2022, [36]).
A simple formulation for the turbulent viscosity is adopted for the wall functions. This is consistent with the NSBL similarity theory. In the Prandtl logarithmic function for the NSBL:

$$\left|\underline{u}(r)-\underline{u}_w\right| = \frac{u_*}{k_v}\ln\left(\frac{r}{z_0}\right), \quad u_* \equiv \sqrt{\frac{\tau}{\rho}}, \quad \max\left\{30\frac{\nu}{u_*}, z_0\right\} \le r \le 0.2\delta_T \tag{5.30}$$

r (m) is the distance from the wall, τ (Pa) is the shear stress and k_v=0.40 is the von Karman constant. The turbulent shear stress is approximately constant in the SBL and is approximately equal to the molecular shear stress at wall. The spatial constraint in (5.30) reduces to the condition $r \ge z_0$ as the particles representing the flood domain close to the solid walls lie in the SBL. Adopting the mixing-length turbulence scheme and considering the friction velocity in (5.30), the turbulent viscosity of a generic particle-wall interaction "$_{0w}$" is approximated as follows:

$$\mu_{T,0w} = \left[\rho k_v u_* r\right]_{0w} = \begin{cases} \dfrac{\rho_0 k_v^2\left(u_{t,0}-u_{t,w}\right) r_{0w}}{\ln\left(\dfrac{r_{0w}}{z_0}\right)}, & r_{0w} \ge z_0 e \\ z_0 e \rho_0 k_v^2\left(u_{t,0}-u_{t,w}\right), & r_{0w} < z_0 e \end{cases} \tag{5.31}$$

where the SPH particle lies in the NSBL and the bottom line relates to the limiter of (5.37).

5.7.3. Slip coefficient for the Neutral Surface Boundary Layer

The present section is reported as a preprint of Amicarelli et al. (2022, [36]).
In many CFD models, the slip coefficient is associated with the computational nodes close to the walls and allows to represent the wall shear stress (Sec.5.7.1). The slip coefficient is here a proxy variable associated with a numerical boundary velocity $\underline{u}_{BC}$ (m/s) to impose Neumann's boundary conditions. A flawless numerical assessment of the wall shear stress would require an expression to relate $\underline{u}_{BC}$ or ϕ_s with the SPH wall shear stress, which could be constrained to the wall-function theory. Unfortunately, such SPH approximation does not allow any suitable formula inversion. Alternatively, a simplified expression for ϕ_s is proposed, according to the NSBL wall-function theory.
The shear stress τ exerted by the fluid on a wall "$_w$" is expressed as follows:

$$\tau_w \equiv \left|\underline{\tau}_{n,w}\cdot\underline{t}_w\right|, \quad \underline{\tau}_{n,w} \equiv \underline{\underline{\tau}}_w \underline{n}_w, \quad \underline{t}_w \equiv \frac{\underline{u}-\left(\underline{u}\cdot\underline{n}_w\right)\underline{n}_w}{\left|\underline{u}-\left(\underline{u}\cdot\underline{n}_w\right)\underline{n}_w\right|} \tag{5.32}$$

where $\underline{\underline{\tau}}$ (Pa) is the stress tensor, $\underline{\tau}_n$ is the stress vector exerted on a wall with normal $\underline{n}$, $\underline{t}$ is the unit vector tangential to the wall and aligned with the local velocity vector.
Combining (5.32) and (5.30), the wall shear stress associated with a point in the NSBL reads:

$$\tau_w = \rho \left[k_v \frac{\left(\left| \underline{u}(r) - \underline{u}_w \right| \right)}{\ln\left(\frac{r}{z_0} \right)} \right]^2 \tag{5.33}$$

The wall shear stress relates to a computational particle "$_0$" interacting with the wall "$_w$" by means of a linear Centred-Finite-Difference approximation of the velocity gradient:

$$\tau_{w,0} = \left[\left(\mu + \mu_T \right) \frac{\partial \left(\underline{u} \cdot \underline{t}_w \right)}{\partial r} \right]_0 \cong \mu_{T,0w} \frac{u_{t,0} - u_{t,BC,0w}}{2 r_{0w}}, \quad \mu << \mu_T \tag{5.34}$$

where u_t (m/s) indicates the tangential component of velocity with respect to the wall and $\underline{u}_{BC}$ is formally associated with a virtual particle outside the fluid domain, whose position is symmetric to $\underline{x}_0$ with respect to the wall. The definition of the slip coefficient ϕ_s permits to express the wall shear stress as function of the wall velocity instead of $\underline{u}_{BC}$:

$$\tau_{w,0} = \phi_{s,0w} \mu_{T,0w} \frac{\left(u_{t,0} - u_{t,w} \right)}{r_{0w}}, \quad \phi_{s,0w} \equiv \frac{\tau_{w,0} r_{0w}}{\mu_{T,0w} \left(u_{t,0} - u_{t,w} \right)} \tag{5.35}$$

The equivalent expression for the fictitious boundary velocity reads:

$$u_{t,BC,0w} = u_{t,0} + 2\phi_{s,0w} \left(u_{t,w} - u_{t,0} \right), \quad 0 \le \phi_{s,0w} \le 1 \tag{5.36}$$

In the region between the computational particle and the wall, velocity cannot be larger than the particle velocity or smaller than the wall velocity. The associated limits are represented by symmetric free-slip conditions with ϕ_s=0 and anti-symmetric no-slip conditions with ϕ_s=1.

Provided the expressions for the NSBL wall shear stress (5.33) and turbulent viscosity (5.31), a formulation for the slip coefficient of the computational particles in the NSBL is proposed:

$$\phi_{s,0w} = \begin{cases} \left[\ln\left(\frac{r_{0w}}{z_0} \right) \right]^{-1}, & r_{0w} \ge z_0 e \\ 1, \quad r_{0w} < z_0 e \end{cases} \tag{5.37}$$

where the bottom line is a numerical limiter avoiding ϕ_s>1 and z_0 depends on the flow regime (Sec.5.7.4). Free-slip conditions are asymptotically represented by $r_{0w} \to \infty$. Each "boundary element - particle" interaction has its own slip coefficient.

5.7.4. Roughness characteristic length

The present section is reported as a preprint of Amicarelli et al. (2022, [36]).

The formulations for the slip coefficient and the turbulent viscosity of Sec.5.7.2-5.7.3 need the definition of the roughness length z_0. This section applies to the rough-wall regime and is integrated in the z_0 formulation for flash floods (Sec.5.7.5). The other regimes are not represented in SPHERA. However, the intermediate-roughness regime might be useful for certain floods for which an alternative formulation is proposed at the end of the current section.

In the rough-wall regime, the Nikuradse wall function is extended for covers of uniform granular material to non-uniform materials:

$$z_0 = \frac{k_s}{30} = \frac{d_{50}}{10}, \quad \mathrm{Re}_* \ge 70 \tag{5.38}$$

The hydraulic roughness k_s (m) can be expressed as function of the 90th percentile of the wall roughness-size distribution d_{90} (m), according to van Rijn (1982, [43]), and can be approximated by the median roughness diameter d_{50} (m), more commonly available:

$$k_s = 3 d_{90} \simeq 3 d_{50} \tag{5.39}$$

The roughness median diameter d_{50} is interpreted as the sub-grid roughness depth H_{sgr} (m) halved. The roughness length in the rough-wall regime assumes the following form:

$$z_0 = \frac{H_{sgr}}{5}, \quad \mathrm{Re}_* \geq 70 \tag{5.40}$$

Under the same regime, the top height of the BFL reads:

$$z_{TOP,BFL} = 30\frac{\nu}{u_*} = 30\frac{k_s}{\mathrm{Re}_*} = 900\frac{z_0}{\mathrm{Re}_*} \leq 13 z_0, \quad \mathrm{Re}_* \geq 70 \tag{5.41}$$

In particular, $z_{TOP,BFL}<z_0$ when $Re_*>900$. Under these conditions, the part of the SBL below z_0 is not represented by the logarithmic law ($r \geq z_0$) as the mean velocity of the resolved turbulent scales is null. For instance, this is the case of a flood sub-domain close to the ground in a dense forest whose trees are sub-grid roughness elements. This is also a physical interpretation to the no-slip low-pass filter in the formulation (5.37) for the slip coefficient of SPHERA, which numerically imposes an analogous and more rigid constraint ($r \geq z_0 e$) to the application of the logarithmic law.
The SBL Nikuradse wall function for the smooth-wall regime provides the following expression for the roughness length, after minor rearrangements to use the natural logarithm:

$$z_0 = \frac{1}{9.0}\frac{\nu}{u_*}, \quad \mathrm{Re}_* \leq 5 \tag{5.42}$$

5.7.5. Roughness length for flash floods: CORINE Land Cover, DEM and DTM

The present section is reported as a preprint of Amicarelli et al. (2022, [36]).
A formulation for the roughness length under flash floods is proposed. It is an adaptation of the formulation of Sec.5.7.4 for the rough-wall regime to consider the CORINE Land Cover (CLC) maps, Digital Elevation Models (DEMs), Digital Terrain Models (DTMs) and the obstacle heights.
The topographic surface is represented in SPHERA by a DTM, a DEM or a hybrid DEM/DTM surface. In the last case, the DEM is locally replaced by the DTM only where their height difference is smaller than $H_{f,DEM}$ (m), which is a high-pass filter on the height of the DEM roughness elements. The roughness length z_0 is expressed as follows:

$$z_0\left(\underline{x}\right) = z_{0,\varepsilon,DEM} + \left[z_{0,\varepsilon,DTM} + z_{0,LC}\left(\underline{x}\right)\right] \cdot H_s\Big|_{\left[z_{0,f,DEM} - z_{0,LC}\left(\underline{x}\right)\right]} \cong z_{0,\varepsilon,DEM} + z_{0,CLC}\left(\underline{x}\right) \cdot H_s\Big|_{\left[z_{0,f,DEM} - z_{0,CLC}\left(\underline{x}\right)\right]} \tag{5.43}$$

where the term $z_{0,\varepsilon}$ (m) is associated with the truncation error ε_{σ_z} (m) of the topographic surface, $z_{0,LC}$ (m) relates to the soil-cover roughness obtained by experimental assessments, $z_{0,CLC}$ (m) is the soil-cover contribution linked to the CLC classes (Table 5.1) and H_s is the Heaviside step function. This activates the soil-cover term only where the DEM is locally removed by the filter $z_{0,f,DEM}$, associated with $H_{f,DEM}$. The DEM absence is represented by the formal condition $H_{f,DEM}=H_{r,max}$, where H_r is the depth of the overall roughness, considering both its sub-grid and explicit components. After (5.40), the following expressions apply to (5.43):

$$z_{0,\varepsilon} = \frac{\varepsilon_{\sigma_z}}{5}, \quad \varepsilon_{\sigma_z} = \sigma_z\big|_{\Delta x} - \sigma_z\big|_{2\Delta x}, \quad z_{0,f,DEM} = \frac{H_{f,DEM}}{5}, \quad z_{0,LC} = \frac{H_{sgr}}{5} \tag{5.44}$$

where σ_z (m) is the standard deviation of the DEM or DTM heights. The truncation error of each topographic surface is uniform within the same file and is treated as a numerical roughness depth in (5.44). It is assessed considering the σ_z variations between two representative spatial resolutions, e.g., Δx (m) and $2\Delta x$, but only if Δx is fine enough. Otherwise, any local convergence of σ_z is unreliable. This condition is peculiar of any coarse DEM (e.g., a DEM with Δx=31m as in the dataset SRTM3, 2014, [50]), whose truncation error is negligible with respect to $z_{0,LC}$. This means that the local roughness elements of a coarse DEM are not explicitly represented. Thus, a coarse DEM can be treated as a DTM with negligible $z_{0,\varepsilon}$, when using (5.44).
According to (5.43), z_0 values are associated with the CLC 2018 classes (Kosztra et al., 2019, [51]). These values are obtained by means of a simple elaboration of the z_0-values reported in Jancewicz & Szymanowski (2017, [52]). Those values were validated on numerical meso-scale meteorological studies. Under those applications, the DTM truncation errors were implicitly involved in $z_{0,CLC}$. The similarity between the Atmospheric SBL and the SBL of flash floods considers that the z_0 does not depend on the fluid type under rough-wall regimes, but only on the wall geometry via (5.40).

CLC class code	2018 CLC class description	$z_{0,CLC}$ (m) SPHERA (CLC class mid-point)	z_0 (m) Jancewicz & Szymanowski (2017)
10000	**Artificial areas**		
11000	Urban fabric	1.0	1.5 ("continuous urban fabric") 0.6 ("discontinuous urban fabric")
12000	Industrial, commercial and transport units	1.0	1.5 ("continuous urban fabric") 0.6 ("discontinuous urban fabric") (0.04) ("roads and associated lands")
13000	Mine, dump and constructon sites	0.15	0.01 ("dump sites; mineral extraction sites") 0.3 ("construction sites")
14000	Artificial non-agricultural vegetated areas	0.3	0.3
20000	**Agricultural areas**		
21000	Arable land	0.03	0.03
22000	Permanent crops	0.3	0.3 ("agro-forestry areas") 0.3 ("fruit trees and berry plantations")
23000	Pastures	0.03	0.03
24000	Heterogeneous agricultural areas	0.2	0.3 ("agro-forestry areas") 0.1 ("complex cultivation patterns; land principally occupied by agriculture with significant areas of natural vegetation")
30000	**Forests and semi-natural areas**		
31000	Forests	1.4	1.4 ("forests") (0.4) ("transitional woodland-shrub")
32000	Shrubs and/or herbaceous vegetation associations	0.2	0.2 ("sparsely vegetated areas")
33000	Open spaces with little or no vegetation	0.02	0.03 ("natural grassland") 0.01 ("bare rock")
40000	**Wetlands**		
41000	Inland wetlands	0.005	0.005 (Davenport updated table, "marsh")
42000	Coastal wetlands	0.005	0.005 (Davenport updated table, "marsh")
50000	**Water bodies**		
51000	Inland waters	0.0002	0.0002 (Davenport's updated table, "open sea or lake irrespective of wave size")
52000	Marine waters	0.0002	0.0002 (Davenport's updated table, "open sea or lake irrespective of wave size")

Table 5.1. Values of the roughness length $z_{0,CLC}$ related to the CLC 2018 classes (Kosztra et al., 2019, [51]), obtained by simple elaboration of the z_0 values validated in Jancewicz & Szymanowski (2017, [52]) on meteorological studies.

The CLC maps are usually available as shapefiles (".shp"). In the numerical chain of SPHERA, these files are elaborated by means of GDAL (2021, [53]) and Paraview (Kitware, 2021, [54]) to obtain a set of ".ply" files of CLC triangulated polygons and a text file which links each polygon to its CLC class. These files are elaborated by SPHERA at the resolution requested.
Delaunay's triangulation executed by Paraview presents some surface overestimations outside the concave polygons. This causes some local polygon-overlapping. To solve this shortcoming, a priority index is introduced in SPHERA to associate the most proper CLC polygon with each point where z_0 is assessed. This index is called "vertex-usage code of a face" F_{VUC}. It reduces the discretization errors of Delaunay's triangulation. The complete removal of such errors, if necessary, is a peculiar task for visualization tools more than SPH codes. Nonetheless, the following algorithm is adopted in SPHERA for F_{VUC}. Any barycentre of a cell of the positioning grid of SPHERA can simultaneously lie within the spurious face of a locally concave CLC polygon and the correct face of a locally convex CLC polygon. Usually, all the vertices of the wrong face are already used by other faces of the same triangulated polygon. Contrarily, at least one vertex of the correct face is not used by the other faces

of the locally convex polygon. The vertices of a spurious face are usually overused, with respect to the vertices of a correct face. Starting from this principle, often useful but not always exact, a strategy has been adopted to assign higher priorities to the faces with underused vertices.
For each triangular face of a CLC polygon, F_{UVC} is defined as follows:

$$F_{VUC} = \min_{i,j,k} \left\{ V_{o,i} \cdot 10^6 + V_{o,j} \cdot 10^3 + V_{o,k} \right\} \tag{5.45}$$

where V_o is the occurrence of the generic vertex in all the faces of the same CLC polygon of the current face. The minimum operator applies to alle the $\prod_{i=1}^{3} \binom{i}{1} = 6$ possible combinations of ordered lists of the three vertices "$_{i,j,k}$" of the current face. This operator and the powers of ten in (5.45) guarantee that F_{UVC} is dominated by the least used vertex and then, in case of tie, by the following vertices, still ordered according to their underuse in the triangulated polygon. The polygon searching adopts the same "dynamic-vector method" executed in SPHERA to search the neighbouring SPH particles, i.e., those particles influencing the balance equations of any current computational particle.
Finally, where the CLC maps are unavailable, the field of $z_{0,LC}$ is reconstructed by reliable z_0-values measured in the Atmospheric BL for analogous soil covers or using the last formula in (5.44) after an assessment of the maximum height of the sub-grid obstacles.
The same slip coefficients also interest the partial smoothing of the velocity field (on the fly computation is preferred to saving the values), even if this procedure is not activated by default.
One notices that the SASPH method does not apply the slip coefficient to the continuity equation.
Analogous approaches based on wall functions are commonly adopted by CFD codes (e.g., Ferrand et al., 2013, [20]; CFX, 2020, [55]).

6. BOUNDARY TREATMENT SCHEME FOR THE TRANSPORT OF SOLID BODIES

6.1. SPH balance equations for the transport of rigid solid bodies

This section relies on Amicarelli et al. (2015, [4]), Amicarelli et al. (2020, [5]), the RSE developments in Sec.5 of Paggi et al. (2021, [56]) and Amicarelli et al. (2022, [7]), whose reading is suggested for further details.
Body dynamics is ruled by Newton-Euler equations, whose discretization takes advantage from the SPH formalism and the coupling terms derived in the following sections:

$$\frac{d\underline{u}_{CM}}{dt} = \frac{\underline{F}_{TOT}}{m_B}, \quad \frac{d\underline{\chi}_B}{dt} = \underline{\underline{I}}_C^{-1} \left[\underline{M}_{TOT} - \underline{\chi}_B \times \left(\underline{\underline{I}}_C \underline{\chi}_B \right) \right] \tag{6.1}$$

These represent the balance equations of the linear and angular momentum of a solid rigid body. The following kinematic formulae are used for time integrating the body velocity and angular velocity to obtain the barycentre position and the orientation of the body:

$$\frac{d\underline{x}_{CM}}{dt} = \underline{u}_{CM}, \quad \frac{d\underline{\alpha}}{dt} = \underline{\chi}_B \tag{6.2}$$

where $\underline{\alpha}$ is the vector of the angles lying between the body axes and the global reference system.
Here the subscript "$_B$" refers to a generic computational body and "$_{CM}$" to its centre of mass.
The first formula of (6.1) represents the balance equations for the momentum. $\underline{F}_{TOT}$ is the global/resultant force acting on the solid body. The last formula of (6.1) expresses the balance equation of the body angular momentum, where χ_B denotes the angular velocity of the generic body. $\underline{M}_{TOT}$ represents the associated torque acting on the body and $\underline{\underline{I}}_C$ the matrix of the moments of inertia of the computational body (Einstein's notation works for the subscript "$_l$"):

$$I_{c,ij} = \int_{V_B} \rho\left(r_l^2 \delta_{ij} - r_i r_j\right) dV = \left\{ \begin{array}{l} \int_{V_B} \rho\left(r_k^2 + r_n^2\right) dV, i = j; k, n \neq i \\ - \int_{V_B} \rho\left(r_i r_j\right) dV, i \neq j \end{array} \right\},$$

$$I_{c,ij} = \begin{bmatrix} \int_{V_B} \rho\left(r_y^2 + r_z^2\right) dV & -\int_{V_B} \rho r_x r_y dV & -\int_{V_B} \rho r_x r_z dV \\ sym & \int_{V_B} \rho\left(r_x^2 + r_z^2\right) dV & -\int_{V_B} \rho r_y r_z dV \\ sym & sym & \int_{V_B} \rho\left(r_x^2 + r_y^2\right) dV \end{bmatrix} \quad (6.3)$$

In this sub-section, $\underline{r}$ implicitly represents the relative distance from the body centre of mass.
One considers the final formulation of the second formula of (6.1). When the first/second term in the squared parentheses is negligible, then the resulting rotation is called torque-free/torque-induced precession. The minor effect of the torque-free term on a torque-induced precession is called nutation.
Torque-free precession occurs when the integral of the external forces, internal forces and external torques are null, but the integral of the internal torques is non-null. The internal torques are induced by the internal forces. These normal and shear stresses within the body material are necessary to permit the centripetal accelerations of the material points of the body.
Torque-free precessions are prevented if at least one of the following requirements is satisfied: body symmetry around the current rotation axis; body symmetry around the plane normal to the current rotation axis and passing for the body barycentre; null angular velocity.
The inverse of the matrix of the moment of inertia acts like a rotation matrix for the torque in the second formula of (6.1) and thus can induce a change in the rotation axis in the presence of an external torque. Torque-induced precession is prevented if at least one of the following conditions is satisfied: body symmetry around the current rotation axis; body symmetry around the plane normal to the current rotation axis and passing for the body barycentre; null torque.
In order to solve the system (6.1), we need to model the global force and torque, as described in the following. The resultant force is composed of several terms:

$$\underline{F}_{TOT} = \underline{G} + \underline{P}_F + \underline{T}_F + \underline{P}_S + \underline{T}_S \quad (6.4)$$

$\underline{G}$ represents the gravity force, whereas $\underline{P}_F$ and $\underline{T}_F$ the vector sums of the pressure and shear forces provided by the fluid. Analogously, $\underline{P}_S$ and $\underline{T}_S$ are the vector sums of the normal and the shear forces provided by other bodies or boundaries (solid-solid interactions). In case of inertial and quasi-inertial fluid flows, we do not need to refer to neither turbulence scheme nor tangential stresses (simplifying hypothesis). Nonetheless, shear forces can be represented under more generic conditions (Sec.6.6).
Gravity is always active. The expressions "gravity deactivated" (input file template) only refers to the activation of drag and reaction forces, which temporarily balances gravity components, during an impingement or in case of sliding. The approximations refer to drag and reaction forces, not to gravity.
The fluid-solid interaction is expressed by the following pressure force:

$$\underline{P}_F = \sum_s p_s A_s \underline{n}_s \quad (6.5)$$

The computational body is numerically represented by solid volume elements, here called (solid) "body particles" ("$_s$"). Some of them describe the body surface and are referred to as "surface body particles". These particular elements are also characterized by an area and a vector $\underline{n}$ of norm 1. This is perpendicular to the body face of the particle (it belongs to) and points outward the fluid domain (inward the solid body).

The pressure of a body particle is computed by means of the boundary treatment of Adami et al. (2012, [16]), here implemented and adapted as described in Sec.6.1. Further, the solid-solid interaction term ($\underline{P}_s$) is presented in Sec.6.8.
On the other hand, the torque in (6.1) is discretized as the summation of each vector product between the relative position $\underline{r}_s$, of a surface body particle with respect to the body centre of mass, and the corresponding total particle force:

$$\underline{M}_{TOT} = \sum_s \underline{r}_s \times \underline{F}_s \tag{6.6}$$

Time integration of (6.1) is performed using a Leapfrog scheme synchronized with the fluid dynamics balance equations. This means that the body particle pressure is computed simultaneously to fluid pressure, so that this parameter is staggered of around *dt*/2 with respect to all the other body particle parameters.
After time integration, the model obtains the velocity of a body particle as the vector sum of the velocity of the corresponding body barycentre and the relative velocity:

$$\underline{u}_s = \underline{u}_{CM} + \underline{\chi}_B \times \underline{r}_s \tag{6.7}$$

Finally, the model updates the body particle normal vectors and absolute positions, according to the following kinematics formulas ($\underline{d\alpha}$ is the increment in the body rotation angle during the on-going time step and R_{ij} is the body rotation matrix):

$$\underline{n}_s(t+dt) = \underline{\underline{R}}_B \underline{n}_s(t), \qquad \underline{x}_s(t+dt) = \underline{x}_{CM}(t+dt) + \underline{\underline{R}}_B \underline{r}_s(t)$$
$$\underline{\underline{R}}_B = \underline{\underline{R}}_x \underline{\underline{R}}_y \underline{\underline{R}}_z, \qquad \underline{d\alpha}_B = \underline{\chi}_B dt$$
$$\underline{\underline{R}}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(d\alpha_x) & -\sin(d\alpha_x) \\ 0 & \sin(d\alpha_x) & \cos(d\alpha_x) \end{bmatrix}, \underline{\underline{R}}_y = \begin{bmatrix} \cos(d\alpha_y) & 0 & \sin(d\alpha_y) \\ 0 & 1 & 0 \\ -\sin(d\alpha_y) & 0 & \cos(d\alpha_y) \end{bmatrix}, \underline{\underline{R}}_z = \begin{bmatrix} \cos(d\alpha_z) & -\sin(d\alpha_z) & 0 \\ \sin(d\alpha_z) & \cos(d\alpha_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.8}$$

At the moment, the following simplifications are assumed: the drag force $\underline{T}_F$ exerted on the body by the fluid is not represented even if locally the body exerts a shear force on the fluid; each body is given in input a uniform density.

6.2. Improving 3D rotations

The present section refers to Amicarelli et al. (2022, [7]). The equations:

$$\frac{\underline{d\alpha}}{dt} = \underline{\chi}_B, \quad \underline{\underline{R}}_B = \underline{\underline{R}}_x \underline{\underline{R}}_y \underline{\underline{R}}_z \tag{6.9}$$

are not exact and represent a limit when working with a rotation matrix based on Euler angles: the first equation of (6.9) is exact only for 1D rotations, the second one is never exact (non-commutative property of the rotation matrices based on Euler angles).
The following formula represents a generalized Euler-Newton equation for 3D rotations and provides exact results in 3D:

$$\frac{d\underline{\underline{R}}_B}{dt} = \underline{\underline{skew}}(\underline{\chi}_B)\underline{\underline{R}}_B \tag{6.10}$$

Consider rotation matrices based on Euler's theorem ("each 3D rotation can be represented as a 1D rotation about a generic axis" of a unique rotation angle θ_R). This approach is also called "matrix to axis-angle" and is used by Rodrigues formula to define the rotation matrix:

$$\underline{\underline{R}}_B = \underline{\underline{I}} + (\sin\theta_R)\underline{\underline{skew}}(\underline{n}_R) + (1-\cos\theta_R)\underline{\underline{skew}}^2(\underline{n}_R) \tag{6.11}$$

where the rotation axis is represented by the unit vector $\underline{n}_R$.
The cross-product matrix is a skew symmetric matrix:

$$\underline{\underline{skew}}(\underline{\hat{n}}_R) \equiv \begin{bmatrix} 0 & -n_{R,3} & n_{R,2} \\ n_{R,3} & 0 & -n_{R,1} \\ -n_{R,2} & n_{R,1} & 0 \end{bmatrix} \tag{6.12}$$

The expansion of R_{ij} reads:

$$\underline{\underline{R}}_B = \begin{bmatrix} n_{R,1}^2 + \left(1 - n_{R,1}^2\right)\cos\theta_R & n_{R,1}n_{R,2}\left(1-\cos\theta_R\right) - n_{R,3}\sin\theta_R & n_{R,1}n_{R,3}\left(1-\cos\theta_R\right) + n_{R,2}\sin\theta_R \\ n_{R,1}n_{R,2}\left(1-\cos\theta_R\right) + n_{R,3}\sin\theta_R & n_{R,2}^2 + \left(1 - n_{R,2}^2\right)\cos\theta_R & n_{R,2}n_{R,3}\left(1-\cos\theta_R\right) - n_{R,1}\sin\theta_R \\ n_{R,1}n_{R,3}\left(1-\cos\theta_R\right) - n_{R,2}\sin\theta_R & n_{R,2}n_{R,3}\left(1-\cos\theta_R\right) + n_{R,1,}\sin\theta_R & n_{R,3}^2 + \left(1 - n_{R,3}^2\right)\cos\theta_R \end{bmatrix} \quad (6.13)$$

In 2D, the Rodrigues' rotation matrix is equal to Euler's rotation matrix.
The rotation angle depends on the angular velocity and the time step duration:

$$\underline{\hat{n}_R}\theta_R = \underline{\chi}_B dt, \;\; \underline{\hat{n}_R} = \frac{\underline{\chi}_B}{\left|\underline{\chi}_B\right|}, \;\; \theta_R = \left|\underline{\chi}_B\right| dt \quad (6.14)$$

When working with rotation matrices, the gimbal lock can be only prevented by Rodrigues formula. So far, SPHERA only applies the approach described in this section to provide output data in order to initialize a following restarted simulation. In this frame, SPHERA determines the rotation matrix which provides a rotation from the initial time of the former simulation to a generic time for which output data are available for a following restarted simulation. This represents an approximated procedure which only guarantees a correct re-initialization of the relative position of the first body particle. As the body centre of mass is correctly imposed, the error in the body orientation (of a reinitialized simulation) is limited.
Hereafter is reported the procedure to determine the rotation matrix to move from a first vector (the relative position of the first body particle at the initial time) to a second vector (the relative position of the first body particle at a generic output time). The present approach should not be used to monitor rotations as it does not always describe the most effective rotation. There is more than one rotation which transform a vector from a certain initial position to a fixed final position, but only the one with the minimum rotation angle is interesting to be monitored.
The rotation axis unit vector is estimated as follows:

$$\underline{\hat{n}_R} = \frac{\underline{v_A} \times \underline{v_B}}{\left|\underline{v_A} \times \underline{v_B}\right|} \quad (6.15)$$

The cross product is not sufficient to distinguish between θ_R and (π-θ_R):

$$\sin\theta_R = \left(\frac{\left|\underline{v_A} \times \underline{v_B}\right|}{\left|\underline{v_A}\right|\left|\underline{v_B}\right|} \right) \quad (6.16)$$

whereas the following formula cannot be used:

$$\theta_R = \arcsin\left(\frac{\left|\underline{v_A} \times \underline{v_B}\right|}{\left|\underline{v_A}\right|\left|\underline{v_B}\right|} \right) \quad (6.17)$$

However, consider that θ_R from the cross product automatically and correctly provides an angle between 0 and π ($\sin(\theta_R)$ is non-negative). The sine and cosine functions of the rotation angle read:

$$\sin\theta_R = \frac{\left|\underline{v_A} \times \underline{v_B}\right|}{\left|\underline{v_A}\right|\left|\underline{v_B}\right|}, \;\; \cos\theta_R = \frac{\underline{v_A} \cdot \underline{v_B}}{\left|\underline{v_A}\right|\left|\underline{v_B}\right|} \quad (6.18)$$

Provided its sine and cosine functions, the value of the rotation angle can be estimated by means of the "atan2" function:

$$\theta_R = ATAN_2(\sin\theta_R, \cos\theta_R) = \begin{cases} ATAN\left(\dfrac{\sin\theta_R}{\cos\theta_R}\right), & \cos\theta_R > 0 \\ \pi + ATAN\left(\dfrac{\sin\theta_R}{\cos\theta_R}\right), & \cos\theta_R < 0 \\ \dfrac{\pi}{2}, \quad \cos\theta_R = 0, & \sin\theta_R > 0 \\ -\dfrac{\pi}{2}, \quad \cos\theta_R = 0, & \sin\theta_R < 0 \end{cases} \tag{6.19}$$

Once obtained the rotation axis and angle, Rodrigues formula is used to build the rotation matrix and rotate the whole body by means of a single rotation, from the initial time to the time chosen.

6.3. Reaction forces

The present section refers to Amicarelli et al. (2022, [7]).
Following the superposition principle of effects, the reaction forces of a generic body dynamics configuration are computed as vector sums of the reaction forces associated with two different configurations: the associated hyperstatic configuration and the associated dynamic configuration in the absence of other forces. Under hyperstatic conditions, the system of governing equations concern 6 balance equations (associated with the 6 degrees of freedom of the 3D body) and n compatibility equations (representing the deformation consistency among different parts of the same body).
The contributions to the reaction forces in the dynamic configuration should superpose the effects of the following dynamics mechanisms: 1D impingements in any direction in 3D; 1D sliding in any direction in 3D; pitch; roll; yaw.
SPHERA treats hyperstatic conditions as a dynamic configuration with spurious negligible oscillations: the relative velocities between the solid elements allow to dynamically develop the reactions necessary to maintain the hyperstatic configuration. The normal reaction in SPHERA is formally null if the normal velocity is zero. However, according to Monaghan (2005, [57]), the normal forces of the boundary force particles dynamically restore the normal reaction forces. This approach is like a simple spring-model approach, but the reactions are triggered by the relative velocities, not the relative positions. In any case, these relative velocities and the associated displacements should be negligible with respect to the spatial resolution of the simulation.
SPHERA directly treats 1D impingements in 3D and 1D sliding mechanisms in 3D. SPHERA models the boundary reaction forces associated with the pitch and roll mechanisms thanks to the particle-based assessment of both the impact velocity and the sliding friction limiter. Instead, SPHERA does not represent the boundary reaction forces to the yaw mechanism as the simulated sliding friction force is based on a body approach. Treating the sliding friction force as the vector sum of particle-based friction forces would allow in future to treat the reaction forces associated with the yaw dynamics mechanism.
Contrarily to the normal reaction force, which is exerted gradually during the particle interaction, the sliding friction force is instantaneously applied at its maximum value as the particle interaction begins. This is correct in the frame of rigid bodies and point contacts. On the other hand, the formulation for the normal reaction guarantees the correct representation of the bouncing velocity, no matter about the maximum force. This means that the maximum normal reaction could be reproduced either instantaneously or gradually for rigid bodies. However, the current modelling of sliding friction might represent a systematic overestimation especially in case of curved surfaces (e.g., a wheel rolling on a flat plate) due to the artificial increase of the contact surface and the contact time (a point contact is replaced by a distributed contact). Possible solutions are: the reduction of the influence region just for sliding friction; an alternative gradual activation of sliding friction as if the solids were deformable.

6.4. Solid-solid normal reactions under impingement

This section relies on Amicarelli et al. (2015, [4]), Amicarelli et al. (2020, [5]), and Amicarelli et al. (2022, [7]), whose reading is suggested for further details.

The solid-solid interaction term in (6.5) -$\underline{P}_s$- represents body-body and body-boundary impingement forces, whose time and spatial evolution, in the continuum, is theoretically proportional to Dirac's delta. The numerical model needs to discretize $\underline{P}_s$, as explained hereafter.

The "boundary force particle" method of Monaghan (2005, [57]) defines repulsive forces to represent a conservative full elastic impingement between two SPH interacting particles (of any medium). In particular, the acceleration $\underline{a}_{bfp,jk}$ of particle "j", due to the impingement with particle "k", is aligned with the inter-particle distance $\underline{r}$ and inversely proportional to its absolute value r:

$$\underline{a}_{bfp,jk} = \frac{f_{bfp}}{r_{jk}} \frac{m_k}{m_j + m_k} \underline{r}_{jk} \tag{6.20}$$

The analytic function f_{bfp} is symmetric with respect to the impact point. The dependence of (6.20) on the particle masses allows conserving both global momentum $\left(m_j \underline{a}_{bfp,jk} = -m_k \underline{a}_{bfp,kj}\right)$ and kinetic energy (one may notice that $\underline{r}_{jk} = -\underline{r}_{jk}$ and $f_{jk} = f_{kj}$). The formulation works for inter-particle high velocity impacts.

This formulation is here implemented and extended to whole solid bodies (not only particle impingements), even at low velocities, as well as body-frontier interactions.

Consider the overall force $\underline{P}_s$, which represents the impingements between a generic computational body ("B") and all its neighbouring bodies ("K") and frontiers ("$K*$").

$\underline{P}_s$ is decomposed in elementary 2-body ($\underline{P}_{BK}$) and body-frontier ($\underline{P}_{BK*}$) interactions:

$$\underline{P}_S = \sum_K \underline{P}_{s,BK} + \frac{\sum_{K*} \underline{P}_{s,BK*}}{N_{BK*}} \tag{6.21}$$

where N_{BK*} is the number of neighbouring boundary frontiers.

Adopting the same principles of the boundary force particle method, $\underline{P}_{BK}$ involves interactions between all the body particles "j" of the computational body "B" and their neighbour body particles "k", belonging to the neighbouring body "K":

$$\underline{P}_{BK} = -\alpha_I \sum_j \sum_k \frac{2u^2_{\perp,jk}}{r_{per,jk}} \frac{m_j m_k}{m_j + m_k} \Gamma_{jk} \left(1 - \frac{r_{par,jk}}{dx_s}\right) \underline{n}_k \tag{6.22}$$

The components of the inter-particle relative distance, $\underline{r}_{par}$ and $\underline{r}_{per}$, are parallel and perpendicular to the neighbour normal, respectively. The term within brackets in (6.22) deforms the kernel support of the body particles "j", so that it mainly develops along the direction aligned with the normal of the neighbouring particle (dx_s is the size of the body particles). The weighting function Γ is expressed according to Monaghan (2005, [57]) and depends on $q = r_{jk}/h$:

$$\Gamma_{jk} = \begin{cases} \frac{2}{3}, & 0 \le q < \frac{2}{3} \\ \left(2q - \frac{3}{2}q^2\right), & \frac{2}{3} \le q < 1 \\ \frac{1}{2}(2-q)^2, & 1 \le q < 2 \\ 0, & 2 \le q \end{cases} \tag{6.23}$$

The present model introduces two modifications for body-body interactions, with respect to the original formulation of the boundary force particles. The first one concerns the impact velocity $u_{\perp,jk}$, which replaces the term ($0.1c$) in the formulation of Monaghan (2005, [57]) and properly deals with low velocity impacts. It avoids too strong or too weak impingement forces. For each body-body interaction, the impact velocity has a unique value for all the particle-particle interactions during the on-going time step. This velocity is computed as the maximum of the absolute values of the inter-particle relative velocity (projected over the normal of the neighbouring particle). For this purpose, the model considers all the inter-particle interactions recorded while the 2 bodies are approaching. The expression for the impact velocity reads:

$$u_{\perp,jk}(t) = \max_{j,k,t^*}\left\{\left|\left(\underline{u}_j - \underline{u}_k\right)\cdot \underline{n}_k\right|\right\}, \qquad t_0 \le t^* \le t \tag{6.24}$$

where t_0 refers to the beginning of the approaching phase. When other forces (e.g. pressure and gravity forces) are taken into account, the impact velocity can eventually increase in the inter-body impact zone, causing a potential and partial penetration of a solid into another body. In this case, and only during the approaching phase, (6.24) allows increasing the magnitude of the impingement force, depending on the actual impact velocity (instead of the undisturbed impact velocity). This modification avoids mass penetrations in case of complex impingements.

Further, (6.22) introduces the coefficient α_I. This normalizing parameter corrects discretization errors and better preserves the global momentum and kinetic energy of the body-body system during the impingement. If one omitted α_I, (6.22) would drastically under-estimate the impingement forces if the whole mass of the bodies did not lie within the impact zone (of depth $2h$). To avoid this shortcoming, a formulation for α_I is presented hereafter. Consider the absolute value of the impingement force P_s as a function of the global parameters of the bodies, instead of the particle values. This second formulation for P_{BK} is denoted as follows:

$$P_{BK}' \equiv \frac{2u_{\perp,BK}^2}{r_{per,BK}}\frac{m_B m_K}{m_B + m_K}\Gamma_{BK}, \quad r_{per,BK} = \min_{B,K}\left\{r_{per,jk}\right\}, \quad u_{\perp,BK}^2 = \max_{B,K}\left\{u_{\perp,jk}^2\right\} \tag{6.25}$$

The inter-body velocity impact $u_{\perp,BK}$ is now defined as the highest among the particle impact velocities, while the relative inter-body distance is considered as the minimum among the corresponding inter-particle distances. In practise, $u_{\perp,BK}$ can be roughly, but more efficiently, estimated as the sum of the absolute values of the two body particles, whose interaction shows the highest relative velocity in the system.

One may now derive a proper definition for α_I, by equalling P_{BK} to P_{BK}':

$$\alpha_I = \frac{\dfrac{u_{\perp,BK}^2}{r_{per,BK}}\dfrac{m_B m_K}{m_B + m_K}\Gamma_{BK}}{\sum_j\sum_k\left[\dfrac{u_{\perp,jk}^2}{r_{per,jk}}\dfrac{m_j m_k}{m_j + m_k}\Gamma_{jk}\left(1 - \dfrac{r_{par,jk}}{dx_s}\right)\right]} \tag{6.26}$$

In practise, the model prefers using the following approximated formulation to speed-up the simulations:

$$\alpha_I = \frac{\dfrac{1}{r_{per,BK}}\dfrac{m_B m_K}{m_B + m_K}\Gamma_{BK}}{\sum_j\sum_k\left[\dfrac{1}{r_{per,jk}}\dfrac{m_j m_k}{m_j + m_k}\Gamma_{jk}\left(1 - \dfrac{r_{par,jk}}{dx_s}\right)\right]} \tag{6.27}$$

This is equivalent to considering the body impact velocity as a weighted average of the particle impact velocities. At a first approximation, the normalizing factor α_I roughly represents the inverse of the fraction of the system mass which lies into the impingement zone. This mass should numerically represent the 2-body system during the impact. On the other hand, one cannot use (6.26) to model a body-body impact. In this case, for example, a definition for the direction of P_s*'* is required, but the direction of the relative distance between the two bodies does not avoid mass penetration. This would happen, for example, if two cubic bodies, very close to each other and with null barycentre velocities, began to rotate.

The model represents body-boundary interactions. A generic boundary is simulated as a body with infinite mass and discretization tending to zero (the semi-analytical approach, used to model frontiers, is an integral method). The interaction force assumes the following expression (here the subscript "$_{K*}$" refers to a generic neighbouring frontier):

$$\underline{P}_{BK*} = -\alpha_I \sum_j \frac{2u^2_{\perp,jK*}}{r_{per,jK*}} m_j \Gamma_{jK*} \underline{n}_{K*}, \quad \alpha_I = \frac{\frac{m_B}{r_{per,BK*}} \Gamma_{BK*}}{\sum_j \left(\frac{m_j}{r_{per,jK*}} \Gamma_{jK*} \right)} \tag{6.28}$$

The body-boundary forces are normalized by the number of neighbouring frontiers for a given body. This number is approximated by the maximum number of neighbouring frontiers of a single body particle belonging to the body.

6.5. Normal restitution coefficient

The simulated normal restitution coefficient (R_n) is equal to the unity if all the following conditions are satisfied: homogeneous velocity of the solid particles belonging to the impinging body, impingement with a single frontier element, isolated impingement, body axes aligned with the frontier axes. Otherwise (real cases), the scheme for body-boundary force is dissipative and represents R_n<1. The equivalent value of the restitution coefficient might be estimated a posteriori.

6.6. Sliding friction force

The present section refers to Amicarelli et al. (2022, [7]).
The formulation for the sliding friction force refers to Amontons' law applied to a single body interacting with multiple frontiers. It might be revised for a single body interacting with multiple bodies. The current code version assumes that a unique friction angle applies to all the body-frontier interactions. The overall normal of the neighbouring frontiers is the unit vector aligned with the vector sum of the neighbouring normals. In case of need, this sum might be weighted in a proper manner.

6.6.1. Aerial stage (non-negative value for the input friction angle and body-frontier interactions)

The present section refers to Amicarelli et al. (2022, [7]).
The simulated and exact formulation (*) for the sliding friction force are equal:

$$\underline{T}^*_S = -\underline{G} \cdot \underline{n}_w \mu_{sf} \underline{s}_w = -\underline{G} \cdot \underline{n}_w \tan(\varphi_{dry}) \underline{s}_w \tag{6.29}$$

where φ is the sliding friction angle, the subscript "$_{dry}$" refers to dry conditions, $\underline{s}_w$ is the unit vector parallel to the local frontiers and with the same direction as the velocity vector of the body barycentre (projected on the local DEM).
The coefficient of sliding friction (μ_{sf}) reads:

$$\mu_{sf} = \tan(\varphi_{dry}) \tag{6.30}$$

The absolute value for the exact formulation reads:

$$\left| \underline{T}^*_S \right| = mg \cos(\alpha_{DTM}) \tan(\varphi_{dry}) \tag{6.31}$$

where the cosine of the slope angle is obtained by means of a dot product:

$$\cos \alpha_{DTM} = \left| \underline{n}_w \cdot \underline{k} \right| \tag{6.32}$$

The direction of the sliding friction force is the opposite to the velocity direction of the centre of mass of the computational body.
The following limiter applies to the sliding friction force:

$$\left| \underline{T}^*_S \right|_{\max} = \frac{\left| \underline{u}_{\tan} \right| m}{dt} \tag{6.33}$$

where $\underline{u}_{\tan}$ is the maximum particle velocity (all over the computational body) tangential to the interacting local frontier elements.

6.6.2. Aerial stage (negative value for the input friction angle or body-body interactions)

The present section refers to Amicarelli et al. (2022, [7]).
The sliding friction force ($\underline{T}s$: drag force on a body under body-boundary interactions) is approximately represented by means of a tangential force, which balances the tangential component of gravity:

$$\underline{T_S} = -\left[\underline{G} - \left(\underline{G} \cdot \underline{n_w}\right)\underline{n_w}\right] \tag{6.34}$$

Its absolute value is:

$$\left|\underline{T_S}\right| = mg \sin\left(\alpha_{DTM}\right) \tag{6.35}$$

where α_{DTM} is the slope angle and $\underline{k}$ the unit vector aligned with the vertical axis:

$$\sin \alpha_{DTM} = \left|\underline{n_w} \times \underline{k}\right| \tag{6.36}$$

The removal of the gravity force component which is parallel to the bottom is equivalent to introducing an approximated sliding friction force, where the slope angle approximates the sliding friction angle.

$$\underline{T_S} = \underline{T_S^*} \Rightarrow mg \sin \alpha_{DTM} = mg \cos\left(\alpha_{DTM}\right) \tan\left(\phi_{dry}\right) \Rightarrow \phi_{dry} \cong \alpha_{DTM} \tag{6.37}$$

This approximation might be acceptable, especially in case the sliding friction angle is unavailable.
Under this configuration, normal reaction does not provide any torque (nevertheless the limiter for the sliding friction force depends on the velocity of the solid particles interacting with the frontiers).

6.6.3. Submerged stage

The present section refers to Amicarelli et al. (2022, [7]).
The sliding friction provided by a solid boundary to a wet body assumes the following form:

$$\underline{T}_S = -\min\left\{T_{S,\max}, T_{S,\lim}\right\}\underline{s_w} \tag{6.38}$$

where $\underline{s}_w$ is the unit vector parallel to the local frontiers and with direction opposite to the velocity vector of the body barycentre (projected on the local frontier surface). If this velocity is null, then the friction force is zero. The maximum absolute value for the sliding friction reads:

$$T_{S,\max} = \left|\left(\underline{G} + \underline{P_F}\right) \cdot \underline{n_w}\right| \tan(\delta) \tag{6.39}$$

where δ is here the sliding friction angle (the current code version assumes a unique value for all the body-frontier interactions). In order to prevent a dissipative force to increase the energy of the system, the following approximated limiter applies:

$$T_{S,\lim} = 2\frac{\left|\underline{u}_{\tan}\right| m}{dt} \tag{6.40}$$

where $\underline{u}_{\tan}$ is the maximum particle velocity (all over the portion of the computational body within the boundary-body interaction region) tangential to the interacting local frontier elements. This limiter is excessive as it avoids any surface velocity reversal.
The sliding friction force contributes to the global torque. The application point of the friction force for any solid body is computed as the average position of the body particles involved in the "body particle - frontier" interactions. Each interaction has the same weight in this averaging process (every particle might be considered more than once).
The following hypotheses are assumed: all the surface body particles in contact with wet boundaries have a defined pressure; all the body-boundary interfaces are hydraulically connected with the fluid; normal reaction forces (with bodies and boundaries, included the current boundaries providing friction) are not considered to assess the sliding force.

6.7. Body-boundary normal reaction force under sliding (at null normal velocity)

6.7.1. Aerial stage

The present section refers to Amicarelli et al. (2022, [7]). The exact formulation is correctly represented:

$$\underline{P}_S = \underline{P}_S^* = -\left(\underline{G}\cdot\underline{n}_w\right)\underline{n}_w, \quad \underline{u}\cdot\underline{n}_w = 0 \tag{6.41}$$

This term is added to the body-boundary normal force of (6.32). The overall normal of the neighbouring frontiers is the unit vector aligned with the vector sum of the neighbouring normals.

6.7.2. Submerged stage

The present section refers to Amicarelli et al. (2022, [7]). The normal reaction is formally null.

$$\underline{P}_S = \underline{0}, \quad \underline{u}\cdot\underline{n}_w = 0 \tag{6.42}$$

According to Monaghan (2005, [57]), the normal forces of the boundary force particles (6.32) dynamically restore the normal reaction force (body-boundary interactions), despite some body spurious oscillations (normal to the frontier) in the interaction zone (noise amplitude comparable with the spatial resolution).

$$\underline{P}_S^* = -\left(\underline{G}+\underline{P}_F\right)\cdot\underline{n}_w, \quad \underline{u}\cdot\underline{n}_w = 0 \tag{6.43}$$

6.8. Fluid-body coupling terms

This section relies on Amicarelli et al. (2015, [4]), Amicarelli et al. (2020, [5]), the RSE developments in Paggi et al. (2021, [56]) and Amicarelli et al. (2022, [7]), whose reading is suggested for further details.

The fluid-body interaction terms rely on the boundary technique introduced by Adami et al. (2012, [16]), implemented and adapted for free-slip conditions (Amicarelli et al., 2015, [4]). If boundary is fixed, this method can be interpreted as a discretization of the semi-analytical approach used to treat fluid-boundary interactions (Sec.5.1). The outer domain of (5.1) is here represented by all the body particles inside the kernel support of the computational fluid particle. Further, Adami et al. (2012, [16]) introduce a new term, related to the acceleration of the fluid-solid interface, which influences the estimation of body particle pressure. The implementation and our modifications of this technique is hereafter described.

The fluid-body interaction term in the Continuity Equation reads (free-slip conditions):

$$2\rho_0\sum_s\left[\left(\underline{u}_s-\underline{u}_0\right)\cdot\underline{n}_s\right]W_s^{'}\omega_s \tag{6.44}$$

Modelling the shear stress gradient term in the Navier-Stokes equation requires an alternative formulation to (6.44). The discretization of this term, which is coherent with the definition of the inter-particle velocity under no-slip conditions, assumes the following form:

$$2\rho_0\sum_s\left(\underline{u}_s-\underline{u}_0\right)\cdot\underline{\nabla}W_s\omega_s \tag{6.45}$$

Analogously, the fluid-body interaction term in the Momentum Equation assumes the form:

$$\rho_0\sum_s\left(\frac{p_s+p_0}{\rho_0^2}\right)W_s^{'}\omega_s - 2\nu_0\sum_s\frac{\left(\underline{u}_s-\underline{u}_0\right)}{r_{0s}}\left.\frac{\partial W}{\partial r}\right|_s\omega_s \tag{6.46}$$

where the subscript "s_0" denotes a generic solid-fluid inter-particle interaction. The first term in the RHS of (6.46) is reported by Adami et al. (2012, [16]) whereas the last term will be discussed later on this section. The pressure value of the generic neighbouring surface body particle "$_s$" is derived as follows. Consider a generic point at a generic fluid-body interface. In case of free-slip conditions, the normal projection of the acceleration on the fluid side ("$_f$") and on the solid side ("$_w$") are equal (in-built motion in the direction normal to the interface):

$$\left(\frac{d\underline{u}_f}{dt}\right)\cdot\underline{n}_w = \left(-\frac{1}{\rho_f}\underline{\nabla}p_f+\underline{g}\right)\cdot\underline{n}_w = \underline{a}_w\cdot\underline{n}_w \tag{6.47}$$

The “wall” acceleration at the position of a generic body particle can then be derived by linearizing (6.47). This depends on the particular computational fluid particle “*0*” we are considering, so that we can refer to the interaction subscript “*s,0*”:

$$\begin{gathered}\int_0^s \left(-\frac{1}{\rho_f} \underline{\nabla} p_f \right) \cdot \underline{n}_w \underline{dl_n} = \int_0^s \left(\underline{a}_w - \underline{g} \right) \cdot \underline{n}_w \underline{dl_n} \Rightarrow \\ \Rightarrow p_{s0} \approx p_0 + \rho_0 \left[\left(\underline{g} - \underline{a}_s \right) \cdot \underline{n}_w \right] \left[\left(\underline{x}_s - \underline{x}_0 \right) \cdot \underline{n}_w \right]\end{gathered} \qquad (6.48)$$

where $\underline{dl_n}$ is a vectorial length element along the centreline of the two particles, projected along the wall element normal.

One applies a SPH interpolation over all the pressure values estimated according to Adami et al. (2012, [16]) to derive a unique pressure value for a body particle:

$$p_s = \frac{\sum_0 p_{s,0} W_{0s} \left(\frac{m_0}{\rho_0} \right)}{\sum_0 W_{0s} \left(\frac{m_0}{\rho_0} \right)} = \frac{\sum_0 \left\{ p_0 + \rho_0 \left[\left(\underline{g} - \underline{a}_s \right) \cdot \underline{n}_w \right] \left[\left(\underline{x}_s - \underline{x}_{'0} \right) \cdot \underline{n}_w \right] \right\} W_{0s} \left(\frac{m_0}{\rho_0} \right)}{\sum_0 W_{0s} \left(\frac{m_0}{\rho_0} \right)} \qquad (6.49)$$

In case of no-slip conditions, in-built motion interests all the acceleration components and the shear stress gradient term is involved; (6.47) is replaced by the following expression:

$$\left(\frac{d\underline{u}_f}{dt} \right) = \left(-\frac{1}{\rho_f} \underline{\nabla} p_f + \underline{g} + \nu \nabla^2 \underline{u} \right) = \underline{a}_w \qquad (6.50)$$

Analogously to (6.48), one obtains:

$$\int_0^s \left(-\frac{1}{\rho_f} \underline{\nabla} p_f \right) \underline{dl_n} = \int_0^s \left(\underline{a}_w - \underline{g} + \nu \nabla^2 \underline{u} \right) \underline{dl_n} \Rightarrow p_{s0} \approx p_0 + \rho_0 \left[\left(\underline{g} - \underline{a}_s - \nu_0 \left\langle \nabla^2 \underline{u} \right\rangle_0 \right) \right] \left[\left(\underline{x}_s - \underline{x}_{'0} \right) \right] \qquad (6.51)$$

In case of no-slip conditions, (6.49) is replaced by the following expression:

$$p_s = \frac{\sum_0 p_{s0} W_{s0} \left(\frac{m_0}{\rho_0} \right)}{\sum_0 W_{s0} \left(\frac{m_0}{\rho_0} \right)} = \frac{\sum_0 \left[p_0 + \rho_0 \left(\underline{g} - \underline{a}_s - \nu_0 \left\langle \nabla^2 \underline{u} \right\rangle_0 \right) \cdot \underline{r}_{0s} \right] W_{s0} \left(\frac{m_0}{\rho_0} \right)}{\sum_0 W_{s0} \left(\frac{m_0}{\rho_0} \right)} \qquad (6.52)$$

which is at the moment approximated by this formula:

$$p_s = \frac{\sum_0 p_{s0} W_{s0} \left(\frac{m_0}{\rho_0} \right)}{\sum_0 W_{s0} \left(\frac{m_0}{\rho_0} \right)} = \frac{\sum_0 \left[p_0 + \rho_0 \left(\underline{g} - \underline{a}_s \right) \cdot \underline{r}_{0s} \right] W_{s0} \left(\frac{m_0}{\rho_0} \right)}{\sum_0 W_{s0} \left(\frac{m_0}{\rho_0} \right)} \qquad (6.53)$$

The pressure value of (6.49) or (6.53) is finally used in (6.46).

Only a minority of the body particles represents the body surface, but we also need many inner body particles to estimate p_s. Thus, the model defines the normal vectors for the neighbouring body particles lying inside the bodies, as described by the following algorithm.

For any fluid-body particle interaction “*s0*”, the most representative surface body particle is searched to define $\underline{n}_s$ in (6.52). If the on-going body particle “*s*” belongs to the body surface and satisfies a visibility criterion, then it is immediately considered as representative. Otherwise, the fluid particle “*0*” isolates its visible neighbouring surface body particles. Visibility is assessed considering the sign of the projection of the inter-particle distance on the body particle normal. The visible neighbour, which is the closest to the joining segment of particles “*0*” and “*s*”, is then selected. This particle provides the normal “$\underline{n}_s$” for the fluid-solid particle interaction “*s0*” in (6.52).

The assumption (6.47) relies on the fact that all the involved variables are differentiable in time. This means that this equation cannot properly deal with impulses (infinite accelerations). However, the

numerical accelerations of our model are always finite and the solid particle accelerations can be easily used in (6.52). Nevertheless, we prefer defining a maximum threshold for $|\underline{a}_s|$, here $10g$.
The search for the fluid-body interaction normals is only carried out in case of free-slip conditions.
An improvement on the searching algorithm for the fluid-body interaction normals provides more accurate results and a computational time reduction with respect to Amicarelli et al. (2015, [4]), as described in the following. In that paper, the searching volume V_s was defined as the parallelepiped described by the coordinates of the interacting particles:

$$V_s \equiv V_{s1} : x_i \in \left[x_{0,i}, x_{s,i} \right] \tag{6.54}$$

This caused some issues (i.e. searching area close to zero or no accurate result for the searching algorithm) under the following configurations: interacting particles having almost the same value for one coordinate; surface solid particle being one of the interacting particles. To solve these issues, the new searching algorithm extends the searching region, which is here defined as the intersection between the two identical spheres centred at the interacting particle positions and radius equal to the inter-particle distance (Figure 6.1). The new definition of the searching volume reads:

$$V_s = \left(V_{s1} + V_{s2} + V_{s3} \right) : \begin{cases} d\left(\underline{x}_0, \underline{x}\right) \leq d\left(\underline{x}_0, \underline{x}_s\right) \\ d\left(\underline{x}_s, \underline{x}\right) \leq d\left(\underline{x}_0, \underline{x}_s\right) \end{cases} \tag{6.55}$$

where $d(\ldots)$ (m) is here the distance between two points.

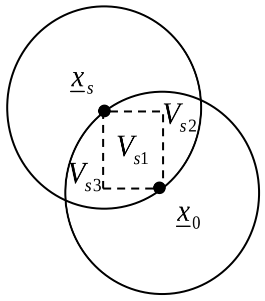


Figure 6.1. Searching volume in fluid-body interactions to define the interaction normal to the fluid-body interface.

The search for the fluid-body interaction normals now explicitly reports possible fluid-solid mass penetrations (just in case of free-slip conditions). In case of penetration (allowed for rough spatial resolutions), the supplementary search for a formal normal now only involves surface body particles. The modified algorithm extends the searching region, which is defined as the intersection between the two identical spheres centred at the interacting particle positions and with radius equal to the inter-particle distance.
Two pressure limiters can be activated to treat the fluid-solid interface:

$$p_{\min} = 0, \quad p_{\max,s,body} = \gamma \left(z_{FS,\max} - z_{s,\min,body} + 2h \right) + \left(1.05 \rho_{ref} \right) c_{ref} \left| \underline{u}_{s,\max,body} - \underline{u}_{\max} \right| \tag{6.56}$$

The "negative value pressure limiter" provides a minimum threshold; the "positive value pressure limiter" acts as a LPRS and provides a maximum pressure threshold: the maximum and minimum operators apply to the whole domain.
Modelling the shear stress gradient term in the Navier-Stokes equation requires the introduction of an additional fluid-body coupling term in the RHS of the fluid momentum equation. It is the last term of (6.46). The discretization of this coupling term, which considers the shear stress exchanged at the fluid-body interface, is derived in coherence with the SPH particle approximation in the inner domain.
The solid particle volume is computed as follows:

$$\omega_s = \frac{\omega_0}{\left(\dfrac{\omega_0}{\omega_s} \right)} = \frac{\left(\dfrac{m_0}{\rho_{ref}} \right)}{\left(\dfrac{dx}{dx_s} \right)^D} \tag{6.57}$$

The inter-particle velocity u_{s0} represents the field of the fluid velocity virtually reconstructed within the portion of the kernel support, which is truncated by the solid body.
The component of the inter-particle velocity, which is normal to the interface, guarantees no mass penetration (symmetric conditions) at the interface:

$$\underline{u}_{s0} = \underline{u}_{s0,n} + \underline{u}_{s0,T} = \left[\left(2\underline{u}_s - \underline{u}_0\right)\cdot \underline{n}_s\right]\underline{n}_s + \left[\left(2\underline{u}_s - \underline{u}_0\right)\cdot \underline{t}_s\right]\underline{t}_s = 2\underline{u}_s - \underline{u}_0 \tag{6.58}$$

Under no-slip conditions, the component of the inter-particle velocity, which is tangential (subscript "$_T$") to the interface, guarantees a uniform velocity gradient around the interface (if its position is assumed to be the average of the positions of the interacting particles):

$$\underline{u}_{s0,T} = \underline{u}_{s,T} - \left(\underline{u}_{0,T} - \underline{u}_{s,T}\right) = 2\underline{u}_{s,T} - \underline{u}_{0,T} = \left[\left(2\underline{u}_s - \underline{u}_0\right)\cdot \underline{t}_s\right]\underline{t}_s \tag{6.59}$$

where the unit vector $\underline{t}$ is tangential to the interface.
Under no-slip conditions, the difference between the inter-particle velocity and the fluid particle velocity in Eq. (51) is expressed as follows:

$$\underline{u}_{SA} - \underline{u}_0 = 2\left(\underline{u}_w - \underline{u}_0\right) \tag{6.60}$$

6.9. Recent improvements to the scheme for body transport

The present section is reported as a preprint of Amicarelli et al. (2022, [14]).
Several improvements to the scheme for body transport have been recently introduced: the procedures to simulate complex solid geometries of any shape; the functionalities of the "proxy body quantities" and a visibility criterion to improve the spatial reconstruction for the fluid-body interactions; the pre-processor directives for the scheme of body transport. Some of them are synthesized in the following.
The body-transport scheme of SPHERA was featured by solid geometries where each body was a Boolean operation over a set of parallelepipeds. This solution is useful to simplify complex geometries (e.g., the electricity pylon in Amicarelli et al., 2022, [7]) but has a limited scope and cannot not be adopted for applications such as marine energy. The scheme is improved to generalize the geometry of the solid bodies which can now assume any 3D complex shape.
The optimum SPH particle distribution to minimize the SPH truncation error is a regular and isotropic Cartesian distribution with particle volumes implicitly equal to cubes (Amicarelli et al., 2011, [15]). Among each polyhedron type, the equilateral polyhedron is the best choice in terms of anisotropy index. The more the volume is isotropic, the smaller is the local SPH truncation error. According to this principle, the best choice for the SPH particle volume would be a sphere and then, in descending order, all the equilateral polyhedrons with decreasing number of faces until the equilateral tetrahedron. However, a finite compact volume cannot be partitioned in non-overlapping spheres. Analogously, all the other equilateral polyhedrons are discarded, except for equilateral tetrahedra and cubes (equilateral parallelepipeds). Cubes have a more isotropic shape, but equilateral tetrahedra represent a better general choice as cubes cannot properly fill the fluid volume under general configurations, e.g., close to boundaries. Furthermore, tetrahedra are easier to be managed when imposing initial conditions. The above discussion has motivated the adoption of tetrahedra as the volumes of the SPH solid body particles when dealing with complex solid bodies produced by Computer-Aided Engineering (CAE) software tools in the numerical chain of SPHERA.
SnappyHexMesh (OpenFOAM v.2.3.0, OpenCFD Ltd, 2022, [58]) was introduced in SPHERA chain to generate the positioning surface meshes for a boundary treatment scheme not used in this study (Amicarelli et al., 2020, [5]). Here several program units are written in SPHERA to elaborate the hexahedral volume meshes produced by SnapphyHexMesh (2022, [58]), as converted in tetrahedral meshes by means of Paraview (2022, [54]). Its tetrahedralization is executed without Steiner's points. These meshes are input elements to elaborate the initial positions, volumes, areas and normals of the body particles which partition the CAE-made solid bodies. Normals are recomputed in SPHERA (2022, [1]) as Paraview (2022, [54]) cannot produce them properly. Further, an automatic correction to the vertex order is applied by SPHERA which fixes many normal anomalies. However, not all the errors can be automatically solved. Thus, a dedicated point-to-point normal check is mandatory. For

each solid body, SPHERA requires in input three lists of particle indexes to flip the residual normal components bad oriented.
The formulations for the normal and the area of the surface body particles of CAE-made solid bodies are hereafter derived. A solid body is partitioned in tetrahedral mesh elements. Every element is replaced by a SPH particle which keeps the same volume and is located at its barycentre. The resulting distribution of the solid particles is not optimized for SPH applications but is a fair discretization of the solid body. For a body particle locally describing the surface of the body (i.e., a SPH surface body particle as defined in Amicarelli et al., 2015, [4]), a sole equivalent value for the normal vector $\underline{n}_s$ and the area A_s (m^2) replaces the values of the tetrahedron faces lying on the solid surface. One considers the hydrodynamic thrust $\underline{S}$ (kg×m×s^{-2}), a major fluid-body coupling quantity, over the surface faces of a generic tetrahedral element whose intersection with the body surface is A (m^2):

$$\underline{S} = \int_A p\underline{n}dA \tag{6.61}$$

Assuming pressure p (Pa) as approximately constant at the sub-particle scale, one obtains:

$$\underline{S} = p\int_A \underline{n}dA = p\underline{A}, \qquad \underline{A} \equiv \left(A_x, A_y, A_z\right) \equiv \int_A \underline{n}dA \simeq \sum_f \underline{n}_{sf} A_{sf} \tag{6.62}$$

where $\underline{A}$ (m^2) is the surface integral of the normal. Its absolute value over an open surface is almost equal to the maximum projectable area delimited by the edge line of the open surface. It is null for closed surfaces. If the edge line links coplanar points, then $|\underline{A}|$ is the area of the surface delimited by the edge line. Imposing the replacement of the surface faces ("$_{sf}$") of the tetrahedron with the associated SPH body particle "$_s$" implies that the particle normal is the normalized average of the face normals weighted by the face areas and the particle area is $|\underline{A}|$:

$$\sum_{sf} \underline{n}_{sf} A_{sf} = \underline{n}_s A_s \Rightarrow \underline{n}_s \equiv \left(\frac{\sum_{sf} \underline{n}_{sf} A_{sf}}{\sum_{sf} A_{sf}} \right) \Bigg/ \left| \frac{\sum_{sf} \underline{n}_{sf} A_{sf}}{\sum_{sf} A_{sf}} \right| \Rightarrow \underline{n}_s = \frac{\sum_{sf} \underline{n}_{sf} A_{sf}}{A_s}, \quad A_s = \left| \sum_{sf} \underline{n}_{sf} A_{sf} \right| \tag{6.63}$$

SnapphyHexMesh produces the mesh-based field of a variable which detects the external surface of a solid body. This field, elaborated by SPHERA, is not always available or cannot cover the whole solid surface, depending on the body. Thus, a complementary surface-detection algorithm is introduced in SPHERA to detect the surface faces of the CAE-made solid bodies. The overall treatment of CAE-made solid bodies in SPHERA is available only in 3D.
Other code improvements concern the "proxy body quantities" and a visibility criterion to improve the spatial reconstruction for the fluid-body interactions, as reported hereafter.
An algorithm for the searching of the proxy normals was available for free-slip conditions before this study to assign a normal to the inner body particles considering for each fluid-body inter-particle interaction the most proper surface body particle. The algorithm is optimized so that the elapsed time of the simulations with solid bodies under free-slip conditions is reduced of 8.4 times. The algorithm is extended to other variables, also under no-slip conditions. In the body boundary contribution to the pressure-gradient term, the wall acceleration is now taken from the surface body particle representative of the fluid-body inter-particle interaction (instead of the interaction body particle). Analogously, the wall velocity for the velocity divergence is taken from the proxy surface particle.
All the internal and surface body particles in the kernel support of a computational fluid particle "$_0$" contribute to its fluid Momentum Equation and to the sub-particle term in the Continuity Equation. Instead, only the surface body particles provide a contribution to the body Momentum Equation. This is based on a reconstruction scheme where the equivalent pressure p_s of a surface body particle depends on all the inter-particle pressure values p_{0s} of its fluid-body interactions. Here a visibility criterion is introduced so that only the p0s values associated with fluid particles visible by the surface body particle are selected for the assessment of ps. This visibility criterion prevents the non-visible

particles to alter the pressure over the solid bodies. Only rare false positives could be detected, if a solid surface is covered by two solid surfaces on the visibility path to a SPH fluid particle.
The surface detection algorithm for solid body particles assumes the two following conditions, the latter including the first:

a) each point geometrically belonging to a cell is explicitly involved in the definition of the faces of the same cell; this imposes to use only one mesh level (no multi-connectivity level); the mesh boundary layer is not an issue as all the points are involved in the definition of the cells they belong to (Fig.5.15 of the guide of OpenFOAM); tetrahedralization (even without face congruence) does not violate this condition; the introduction of Steiner's points does not alter this condition;
b) face congruence: any internal face should represent a whole face for the connected cell; in other words, a face cannot change if viewed from the opposite side (from the connected cell); SnappyHexMesh (included its boundary layer) has no issues in this regard; any proper tetrahedralization should explicitly check for the respect of this condition; a simple polyhedron decomposition is not sufficient: face congruence has to be satisfied.

The number of possible fluid particles intruding the solid domain is monitored. These possible particles are made non-influential and removed from the domain. No matter about the actual presence of such particles, this option ("remove_fluid_from_bodies") is mandatory to assign the proper proxy surface body quantities in the spatial reconstruction scheme for body particles.

6.10. Improving initial conditions in the presence of solid bodies

A pre-processing procedure focuses on removing the SPH fluid particles which lie within the solid sub-domain only to set the proper initial conditions. A fluid particle is removed if its distance from the closest SPH body particle is smaller than the threshold $d_{0s,r}$ (m). At the beginning of the current study the threshold was defined as follows:

$$d_{0s,r} = \frac{\sqrt{D}}{2} dx_s \tag{6.64}$$

Under homogeneous size of the body particles, the maximum correct distance between a surface body particle and a close fluid particle should be:

$$d_{0s,1} = \sqrt{(D-1)\left(\frac{dx_s}{2}\right)^2 + \left(\frac{dx}{2} + \frac{dx_s}{2}\right)^2} \tag{6.65}$$

whereas the minimum correct distance between a surface body particle and a close fluid particle reads:

$$d_{0s,2} = \frac{dx}{2} + \frac{dx_s}{2} \tag{6.66}$$

The expression (6.64) is corrected and generalized as follows for any dx/dx_s with c_1 read from input:

$$d_{0s,r} = c_1 \frac{\sqrt{D}}{4} dx \tag{6.67}$$

7. DB-SPH BOUNDARY TREATMENT SCHEME FOR MOBILE SURFACE WALLS

This section describes the "Discrete Boundary" (DB) - SPH method for boundary treatment (Amicarelli et al., 2013, [9]; Amicarelli et al., 2020, [5]; Amicarelli et al., 2022, [7]). The activation of the DB-SPH method also alters the balance equations in the internal domain, as described in the following sub-sections: DB-SPH particle approximation and modifications of the balance equations (Sec.7.1); 1D Linearized Partial Riemann Solver (Sec. 7.2); semi-particle volume (Sec.7.3); DB-SPH inlet and outlet sections (Sec.7.4); shear stress boundary terms (Sec.7.5).

7.1. DB-SPH particle approximation and modifications of the balance equations

According to the DB-SPH method, the first derivative of a generic function (*f*) is approximated by means of the following SPH particle approximation:

$$\left\langle \frac{\partial f}{\partial x_i} \bigg| \right\rangle_{\underline{x}_0} = \sum_a f_a W_a n_{i,a} \omega_a - \sum_b f_b W_b^{'} \omega_b, \qquad W_b^{'} \equiv \left. \frac{\partial W}{\partial x_i} \right|_b \tag{7.1}$$

In (7.1), the volume integral in (2.2) is replaced with a summation over the fluid particles within the kernel support. The surface integral of the same equation is replaced with a summation over the wall surface elements "*a*" intercepted by the kernel support volume (V_h). Eq.(7.1) is normalized by the integral Shepard coefficient (γ) to obtain this further definition:

$$\left\langle \frac{\partial f}{\partial x_i} \bigg| \right\rangle_{\underline{x}_0} \equiv \sum_a \left(f_A \right) \frac{W_A}{\gamma_0} n_{i,a} \omega_a - \sum_b \left(f_b \right) \frac{W_b^{'}}{\gamma_0} \omega_, \quad \gamma \equiv \int_{V_h} W dx^3 \tag{7.2}$$

γ varies as function of the involved computational particle "*0*". Provided fixed time and position, γ represents a constant for a particle equation system because it does not depend on the neighbouring particles. Thus, the normalization of the kernel derivative is simply obtained dividing by γ. This normalization allows considering the truncated kernel support as if it were entire (in the continuum), but with non-spherical shape.

Eq.(7.2) is used to approximate the pressure gradient term of Euler momentum equation (Sec.5.1). In the absence of the semi-particles, defined by Ferrand et al. (2013, [20]) in 2D, the boundary terms of (7.2) seem too modest to avoid the penetration of fluid particles trough the solid frontiers, once (7.2) is applied to the fluid dynamics balance equations. This limit seems due to the characteristics of the kernel function and its derivative (SPH truncation errors). Thus, the present model adopts semi-particles, whose 3D definition is slightly different from the edge particles (semi-particles) of Ferrand et al. (2013, [20]).

The "semi-particles" represent special fluid particles, which are smallest than the (inner) fluid particles. Each semi-particle is associated to a surface wall element. Semi-particle positions are formally located at the solid frontiers of the fluid domain, but the volumes of the semi-particles completely lie in the inner domain and touch the solid boundaries. The union of the semi-particle volumes represents a thin film of fluid, which is a buffer zone between the inner domain (filled with computational particles) and the wall frontiers. The film depth is smaller than the characteristic length of the fluid particles (*dx*).

Surface elements and semi-particles share the same values of their parameters. Every surface element is defined by its position, velocity, area (length in 2D) and normal vector. Semi-particles additionally require the mass.

Every discrete surface element represents a portion of frontier with area dx_w^2 (3D) or length dx_w (in 2D). At the same position, a fluid semi-particle is located. The semi-particle volumes are smaller than the fluid particle volumes not to alter the spatial resolution. The semi-particle position is located on one side of the physical volume of the semi-particle. However, this position should be representative of the entire semi-particle volume. This implies that the maximum distance between any edge of the semi-particle and its position should be smaller than ${}^{dx_w}\!/_{2}$. Provided this constraint, the semi-particle depth coefficient should be high enough to improve the model accuracy.

Normally, SPH models do not consider the free surface as a frontier of the fluid domain as the atmospheric pressure is usually null in the gaseous sub-domain and on the free surface itself. Here, the DB-SPH approximation (7.2) introduces the parameter $p_0 \neq 0$ in the surface terms of the momentum equation. Formally, one should explicitly model the free surface by means of surface elements over which summing the pressure gradient boundary terms of (7.2). In any case, this complication does not seem necessary if pressure gradients keep small enough at the free surface. This shortcoming is common in SPH mono-phase modelling (using other boundary treatments), and its effects are normally considered negligible (even because pressure gradients are generally zero at the very free surface).

When activating the DB-SPH boundary treatment, density in the inner domain is estimated by means of a SPH particle approximation, which replaces the continuity equation (Ferrand et al., 2013, [20]):

$$\left\langle \rho \right\rangle_0 = \frac{\sum_{bs} \rho_{bs} W_{bs} \omega_{bs}}{\tilde{\gamma}_0} \tag{7.3}$$

where the kernel is normalized by a corrected estimation of the integral Shepard coefficient.
The following correction of γ avoids excessive SPH truncation errors at the free surface:

$$\tilde{\gamma} \equiv \begin{Bmatrix} \sigma, & \gamma \geq \sigma + \varepsilon \\ \gamma, & \gamma < \sigma + \varepsilon \end{Bmatrix}, \quad \sigma \equiv \sum_{bs} W_{bs} \omega_{bs} = \sum_{bs} W_{bs} \frac{m_{bs}}{\rho_{bs}} \tag{7.4}$$

The integral Shepard coefficient is replaced with the discrete Shepard coefficient at the free surface, which is numerically defined where $\gamma \geq \sigma + \varepsilon$. ε can be set equal to 0.05 or chosen as an input parameter to better detect the free surface, depending on the test case and the spatial resolution.
A direct estimation of γ would imply the expensive estimation of 3D analytical integrals. Instead, the present model follows the procedure of Ferrand et al. (2013, [20]), as synthesized by (7.5) and (7.6).
Consider the Lagrangian derivative of γ:

$$\frac{d\gamma}{dt} \cong \sum_a W_a \underline{n}_a \cdot \left(\underline{u}_a - \underline{u}_0 \right) \omega_a, \quad \frac{\partial W}{\partial t} = 0 \tag{7.5}$$

The initial values of γ are approximately provided by the associated values of σ, as the model exactly assigned the initial values of the fluid particle volumes:

$$\gamma_0 \left(t = 0 \right) \cong \begin{Bmatrix} 1, & \min \left\{ r_{0a} \right\} \geq 2h \\ \sigma_0 \left(t = 0 \right), & \min \left\{ r_{0a} \right\} < 2h \end{Bmatrix} \tag{7.6}$$

The integral Shepard coefficient γ is initialized, according to the following procedure.

1) Some fictitious fluid particles are inserted in the computational domain to cover all the truncated parts of the kernel supports in the fluid domain (e.g., the gaseous sub-domain in mono-phase simulations of free surface flows). The density of the fictitious particles is negligible with respect to the computed fluid densities. The fictitious particles are neighbours of the computational particles, close to the free surface. The "fictitious neighbouring particles" define several air volumes, which are provided as input "fictitious fluid volumes".
2) The model computes the initial values of γ by means of the approximated values provided by the estimation of the discrete Shepard coefficient. Thanks to the fictitious particles (having the same characteristic length of the computational particles), the estimation of σ (and then of γ) is sufficiently accurate, as the kernel supports are never truncated by the free surface.
3) The "fictitious air particles" can be removed at the end of γ initialization.

7.2. 1D Linearized Partial Riemann Solver

At boundaries, the fluid velocity component, which is perpendicular to the wall frontier, is equal to the same component of the frontier velocity (non-penetration condition). The model adopts a 1D LPRS (Linearized Partial Riemann Solver) to impose boundary conditions at the wall elements and semi-particles. The 1D LPRS is an up-wind scheme, also used in SPH-ALE modelling (Marongiu et al., 2010, [59]), which allows wall pressure being approximately compatible with the 3D pressure and velocity fields in the inner domain (constrained to the frontier kinematics).
The definition of the initial conditions ("^{L}", "Left") of the 1D LPRS are described by means of a first order spatial reconstruction scheme.
For each interaction ("*0a*") between a surface element ("*a*") and a fluid particle ("*0*"), the LPRS initial conditions are defined at the position of the wall element. Here the model estimates density and the velocity components, by means of a first-order spatial reconstruction scheme around he computational particle (f alternatively refers to density and every velocity component):

$$f_{0a}^L \cong f_0 + \left\langle \underline{\nabla f} \right\rangle_0 \cdot \left\langle \underline{x}_a - \underline{x}_0 \right\rangle, \quad u_{n,0a} = \underline{u}_{0a}^L \cdot \underline{n}_a \tag{7.7}$$

The velocity vector is projected along the normal of the surface wall element to obtain $\underline{u}_n$.
The solution (*) of the LPRS (at the wall element position) provides a reconstructed density value, whereas the associated pressure comes from the EOS (mono-phase formulation):

$$\rho^*_{0a} = \rho^L_{0a} + \left(u^L_{n,0a} - u_{n,a}\right)\frac{\rho^L_{0a}}{c^L_{0a}}, \quad p_{0a} = c^2\left(\rho^*_{0a} - \rho_0\right) \tag{7.8}$$

So far, the model has estimated several values of pressure, at each wall element. The following SPH approximation of these values (summation over all the neighbouring fluid particles) provides a unique pressure value for the surface element:

$$p_a = \frac{\sum_a p_{0a}(W\omega)_a}{\sum_a (W\omega)_a} \tag{7.9}$$

7.3. Semi-particle volume

The volume of a semi-particle ω_s is defined for complex and generic geometries:

$$\omega_s = k_w k_d dx_w \omega_a \tag{7.10}$$

where k_d is the semi-particle shape coefficient, k_w is the semi-particle depth coefficient and Δx_w (m) is the size of the surface wall elements.
The exact assessment of the shape coefficient is not an easy task. However, some exact solutions for noticeable cases are evaluated, both in 2D and 3D, based on the hypothesis of uniform angles (in the same configuration) with the number of adjacent faces equal to the number D of the spatial dimensions. From those exact values, the following interpolating formula is obtained:

$$k_d = \begin{cases} \dfrac{\sum_i \alpha_i}{D\frac{\pi}{2}} \dfrac{D}{n_{af}} - 1 = \dfrac{\sum_i \alpha_i}{\frac{\pi}{2} n_{af}} - 1, & \sum_i \alpha_i > \frac{\pi}{2} n_{af} \\ 0, \quad \sum_i \alpha_i \le \frac{\pi}{2} n_{af} \end{cases} \tag{7.11}$$

where n_{af} represents the number of adjacent elements detected and the subscript "$_i$" here represents the generic adjacent element. The angles α_i (rad) lie between a generic surface element and each of the adjacent elements. According to the adopted formalism, the model needs to add π at the original assessment if the angle between the element normal vectors varies between -π/2 and π/2. The reference formula for α_i reads:

$$\alpha_i = \begin{cases} \pi + ar\cos\left(\underline{\hat{n}}_s \cdot \underline{\hat{n}}_i\right) sign\left[\underline{\hat{n}}_s \cdot \underline{d}_{si}\right], & \underline{\hat{n}}_s \cdot \underline{d}_{si} \ne 0 \\ \pi, \quad \underline{\hat{n}}_s \cdot \underline{d}_{si} = 0 \end{cases} \tag{7.12}$$

Further details are available in [7].

7.4. DB-SPH inlet and outlet sections

The inlet and outlet sections are represented by special surface elements, which are characterized by the following parameters: position, normal vector, null area (or length), pressure. Inlet and outlet surface elements allow detecting the computational particles, which are selected to impose inlet and outlet boundary conditions. The model searches these particles within an influence sphere of characteristic length $L_c/\sqrt{2}$, where L_c represents the size of the inlet/outlet section. This search is very fast but approximated: the accuracy of this simplified procedure depends on the test case. Once the interested computational particles are found, Dirichlet boundary conditions are assigned in terms of pressure and/or velocity components.
The inlet section is also interested by the following procedure, which reduces the SPH truncation errors. The free surface in the inlet region is made wavy to optimize the distribution of the fluid particles. The characteristic wavelength is $dx/2$. The displacements are always perpendicular to the inlet normal. Two pattern regularly alternate. A white noise, with amplitude of $dx/10$, is finally added to the particle positions.

7.5. Shear stress boundary terms for mobile wall frontiers

The shear stress boundary terms of Ferrand et al. (2013, [20]) are adapted to treat mobile frontiers and integrated in the boundary treatment scheme of Amicarelli et al. (2013, [9]).

The boundary terms above appear in the Right-Hand Side of the momentum equation and are due to both surface wall elements and semi-particles. In the first case, one obtains:

$$-\frac{1}{\gamma_0\rho_0}\sum_a \left|\underline{\nabla}\cdot\gamma_{0a}\right|\left[\mu_0\left(\underline{\nabla}u_i\right)\big|_0+\mu_a\left(\underline{\nabla}u_i\right)\big|_a\right]\cdot\underline{n}_a=-\frac{1}{\gamma_0\rho_0}\sum_a W_a\omega_a\left[\mu_0\left(\underline{\nabla}u_i\right)\big|_0+\mu_a\left(\underline{\nabla}u_i\right)\big|_a\right]\cdot\underline{n}_a, \underline{\nabla}\cdot\gamma_{0a}=W_a\omega_a\underline{n}_a \quad (7.13)$$

where $\underline{u}\equiv(u,v,w)$ (m/s) is the velocity vector, $\underline{\nabla}$ is the gradient operator, ρ (kg/m^3) is density, W (m^{-3}) is the kernel function, γ is the integral Shepard coefficient, μ (Pa×s) is the dynamic viscosity, $\underline{n}$ is the normal of the surface boundary element with area ω_a (m^2) and the subscript "*0*" represents the computational particle whose momentum equation is under assessment.

Applying the derivation chain rule and assuming that the normal slowly varies with time, it follows:

$$\frac{\partial u_i}{\partial x_k}n_k=\frac{\partial u_i}{\partial x_k}\frac{\partial x_k}{\partial n_k}\cong\frac{\partial u_i}{\partial \underline{n}}\Rightarrow\left(\mu_0\left(\underline{\nabla}u_i\right)\big|_0+\mu_a\left(\underline{\nabla}u_i\right)\big|_a\right)\cdot\underline{n}_a\cong\left(\mu_0\left.\frac{\partial u_i}{\partial\underline{n}}\right|_0+\mu_a\left.\frac{\partial u_i}{\partial\underline{n}}\right|_a\right) \quad (7.14)$$

where $\underline{x}\equiv(x,y,z)$ (m) is the Cartesian position and Einstein's notation applies to the subscript "*k*". Once provided the similarity law of the viscous sub-layer, aligned the friction velocity vector with fluid velocity close to wall and assumed mobile frontiers, one can write:

$$\left.\frac{\partial u_i}{\partial\underline{n}}\right|_0=\frac{\left(u_{i,0}-u_{i,a}\right)}{d_{0a}} \quad (7.15)$$

where d (m) is the inter-element distance. Assuming a continuous shear stress, the last term of (7.14) is expressed as a SPH approximation normalized by the discrete Shepard coefficient (σ):

$$\mu_a\frac{\partial u_{i,a}}{\partial\underline{n}}=\frac{1}{\sigma_a}\sum_0\mu_0\frac{\left(u_{i,0}-u_{i,a}\right)}{d_0}W_0\omega_0 \quad (7.16)$$

Thus, the shear stress boundary term due to the surface wall elements is:

$$-\frac{1}{\gamma_0\rho_0}\sum_a W_a\omega_a\left[\mu_0\frac{\left(u_{i,0}-u_{i,a}\right)}{d_{0a}}+\frac{1}{\sigma_a}\sum_0\mu_0\frac{\left(u_{i,0}-u_{i,a}\right)}{d_0}W_0\omega_0\right]\cdot\underline{n}_a \quad (7.17)$$

The analogous term due to the semi-particles "*s*" (i.e., fluid elements along the mobile boundaries associated with the surface boundary elements "*a*", Amicarelli et al., 2013, [9]), reads:

$$2\nu_s\sum_s\frac{m_s}{\rho_0 d_{0s}}\left(\underline{u}_s-\underline{u}_0\right)\left.\frac{\partial W}{\partial d_0}\right|_s, \quad \nu_s\equiv\nu_a=\frac{\sum_0\nu_0 W_0\omega_0}{\sum_0 W_0\omega_0} \quad (7.18)$$

where ν (m^2/s) is the kinematic viscosity, *m* (kg) is the SPH particle mass and *r* (m) is the distance from the barycentre of the SPH kernel support.

8. THE SUBSTATION-FLOODING DAMAGE SCHEME

A substation-flooding damage model is presented and integrated in SPHERA (Amicarelli et al., 2021, [60]). The model distinguishes the damage due to power outages from the damage to the components of the electrical substations.

Considering the power outages (blackout events), a vulnerability and "proxy" damage (in time units, not in monetary units) model is presented. This assumes, as a simplifying hypothesis of no redundancy of the electric grid, that the failure of an electrical substation triggers a blackout event in a grid branch (no matter about its length and connections). The assessment of the overall vulnerability of the electric grid and the global damage (in monetary units) due to flood-related blackout events is out of the targets of this code version and needs the following elements: the coupling of the present model with a power grid model; the maps of the exposed population, the values of the public (also environmental) and private goods, the activities affected by the blackout and their flood-related vulnerability curves. On the other hand, considering the components of the electrical substations, a complete (direct and tangible) damage model is presented.

The substation-flooding damage model estimates the following physical quantities, at every electrical substation: the Probability of a power Outage Start (*POS*) as function of the maximum substation water depth; the Expected power Outage Status (*EOS*), at the level of the single electrical substations (in the absence of power grid redundancy); Expected Outage Time/duration (*EOT*, s); the flood-related vulnerability and damage (euros) limited to the components of the electrical substations.

8.1. Mathematical models

The mathematical models of the substation-flooding damage scheme separately considers both blackout events (Sec.8.1.1) and the components of the electrical substations (Sec.8.1.2).

8.1.1. Proxy damage and vulnerability to flood-induced out-of-service events

A proxy damage quantifies in time units (not in monetary units) the damage due to blackout events triggered by substation flooding. One assumes that the malfunction of a substation determines a blackout event in a branch of the power grid (simplifying hypothesis of no redundancy).
The breakdown in the electrical energy supply is analysed only considering the power outage events and neglecting the damage due to brownouts (voltage drops).
The Probability of an Outage Start (*POS*) (i.e., the probability of triggering a blackout event) at a fixed time is smaller than (or equal to) the probability of occurrence of a blackout event at the same time (*EOS*, "Expected Outage Status") because the latter also depends on the blackout events which might start previously.
The procedure of HAZUS-MH (2011, [61]) considers a threshold water depth of Y_{th}=1.2m for blackout events associated with the flooding of electrical substation. However, the above procedure does not mention neither data nor the method to elaborate them. Holmes (2015, [62]) explains that the above threshold value is overestimated. In particular, the majority of the UK electrical substations has shown a threshold of Y_{th}=0.30m (Crawford & Seidel, 2013, [63]). Considering this value and a maximum threshold of $Y_{th,max}$=0.50m, Holmes (2015, [62]) provides a linear relationship between the probability of an Outage Start and water depth, here assumed as spatial average over the territory belonging to the electrical substation (Y_{sub}, m):

$$POS\big|_{t=t^*} = \begin{cases} 1, & Y_{sub}\big|_{t=t^*} \geq 0.52 \\ 1.92Y, & 0 < Y_{sub}\big|_{t=t^*} < 0.52 \\ 0, & Y_{sub}\big|_{t=t^*} = 0 \end{cases} \tag{8.1}$$

The above relationship has been validated on other open-field data (Holmes, 2015, [62]).
The "Expected Outage Status" is defined as a binary variable which represents either the expected presence (*EOS*=1) or the expected absence (*EOS*=0) of a blackout event at a fixed time, at the level of the single electrical substation:

$$EOS\big|_{t=t^*} = \begin{cases} 1, & \sum_i \max\left\{\left(POS\big|_{t=t_i} - 0.49\right), 0\right\} > 0, \quad t^* - t_{rei,e} \leq t_i \leq t^* \\ 0, & otherwise \end{cases} \tag{8.2}$$

EOS is unity if at least once, during the period before the on-going time (*t*=*t**) lasting the restoration time of the electrical infrastructures ($t_{rei,e}$), *POS* is greater than 0.5.
The values of $t_{rei,e}$ are commonly greater for smaller substations with less redundancy. This characteristic time has been quantified by several authors: Chow et al. (1996, [64]) report the interval t_{rei}=0'-500'; Maliszewski & Perrings (2012, [65]) suggest an average value of $t_{rei,e}$=99' and a maximum value of $t_{rei,max}$=330'. Both the above references consider a typical value of t_{rei} smaller than two hours. However, their analyses only consider blackout events under normal (non-extreme) environmental conditions. Reed (2008, [66]) proposes to use a particular realization of the "gamma" probability density function, for t_{rei} on blackouts induced by intense meteorological and flood events. Considering a null minimum value and a maximum value (under the worst hypothesis) of 22 hours (95th percentile of the "gamma" distribution of Reed -2008, [66]-), the present model assumes an

expected restoration time of 11 hours ($t_{rei,e}$=39'600s), in the absence of further data and in favour of safety.
The system (8.1)-(8.2) represents a formulation of the vulnerability of the single electrical substation to the flood-induced blackout events, in the absence of redundancy.
The "Expected Outage Time" (*EOT*, s) is here defined as the expected cumulated duration of the sub-periods of blackout:

$$EOT\big|_{t=t^*} = \sum_i EOS_{t=t_i}\,\Delta t\big|_{t=t_i}, \quad t_0 \le t_i \le t_f \tag{8.3}$$

where Δt is the model time step duration. The subscripts "$_0$", "$_f$" and "$_i$" represent initial conditions, final conditions and a generic time step, respectively. A blackout event can involve several interruptions and the following relationship is assumed: $EOT \geq t_{rei,e}$. Eq.(8.3) represents a formulation of the "proxy" damage (quantified in time units) due to flood-induced blackout events, in the absence of redundancy(at the level of the single substations).

8.1.2. Flood-induced damage to the components of the electrical substations

This mathematical model assesses the (direct and tangible) damage induced by floods to the components of the electrical substations (D_{sub}). This damage model is complete as it quantifies both vulnerability and damage (the latter in monetary units):

$$D_{sub}\big|_{t=t^*} = V_{u,sub}\big|_{t=t^*} \cdot V_{a,sub} \tag{8.4}$$

The value of the electrical substations ($V_{a,sub}$) can be expressed in US dollars (HAZUS-MH, 2011, [61]):

$$V_{a,sub} = \begin{cases} 50'000, & High-Voltage-Substation \\ 20'000, & Medium-Voltage-Substation \\ 10'000, & Low-Voltage-Substation \end{cases} \tag{8.5}$$

but SPHERA works in euros (currency exchange rate of February 2018):

$$V_{a,sub} = \begin{cases} 41'000, & High-Voltage-Substation \\ 16'400, & Medium-Voltage-Substation \\ 8'200, & Low-Voltage-Substation \end{cases} \tag{8.6}$$

$V_{u,sub}$ is the vulnerability of an electrical substation with respect to the (direct and tangible) flood-induced damage involving the substation components. This model provides a regression curve (6th degree polynomial function) for $V_{u,sub}$, after elaboration of the point values reported by HAZUS-MH (2011, [61]):

$$V_{u.sub}\big|_{t=t^*} = \begin{cases} 0.15, & \left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right) \ge 10 \\ \left(\begin{aligned} &\mathrm{a}\left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right)^6 + b\left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right)^5 + c\left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right)^4 + \\ &+\mathrm{d}\left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right)^3 + e\left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right)^2 + \mathrm{f}\left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right) \end{aligned} \right), & 0 < \left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right) < 10 \\ 0, & \left(\mathrm{Y}_{\mathrm{sub,max}}\big|_{\mathrm{t\le t^*}}\right) = 0 \end{cases} \tag{8.7}$$

where the maximum value of the substation water depth $Y_{sub,max}$ refers to the period simulated until the on-going time ($t=t^*$). The regression procedure provides the following constants: a=-1.22877·10^{-6}m^{-6}, b=1.92478·10^{-5}m^{-5}, c=8.4216·10^{-6}m^{-4}, d=-0.00119121m^{-3}, e=0.00390726m^{-2}, f=0.0170243m^{-1}.

8.2. The numerical model

The numerical model of the substation-flooding damage scheme discretize the mathematical models of Sec.8.1. At the beginning of the simulation the following procedures are executed:

- ✓ identification of the DEM vertices internal to the polygons (representing the electrical substations in plan view);
- ✓ assessment of the areas of the above polygons (represented by triangles, quadrilaterals, pentagons and hexagons);

- ✓ initializations of the variables of the electrical substations.

At the end of each output writing time step of the damage model, the following procedures are executed for every electrical substation:

- ✓ assessment of the variable Y_{sub} as spatial average of the water depth values at the DEM vertices within the polygon of the electrical substation;
- ✓ update of the variable $Y_{sub,max}$;
- ✓ assessment of the variables *POS* and *EOS*;
- ✓ update of the variables *EOT*, D_{sub} and $V_{u,sub}$;
- ✓ writing of the output file for the vulnerability and damage variables.

The water depths at the DEM points are saved and stored until the following time step.

As an example, the map of the Italian electrical stations, substations and transmission lines is available at MATTM (2018, [67]). It is useful to activate the layer of OpenStreetMap and the "aerial view" of Bing to detect the electrical stations (squares), substations (polygons) and transmission lines (lines), as well as the electricity pylons (nodes/points). Contrarily, the feature "Atlarete Linee" seems more imprecise (some symbols of the electrical stations would represent electricity pylons; the electrical substations do not seem to represent the nodes of the transmission power grid).

9. INSTALLATION, INPUT-FILE TEMPLATE AND TUTORIALS

The installation of SPHERA (2022, [1]) is straightforward (Sec.9.1). The github repository of SPHERA (2022, [1]) contains a sequence of input files, whose associated test cases are either reported on International Journal papers or represent their analogous simplifications. Please refer to SPHERA (2022, [1]) main references (Sec.1), its numerical schemes (Secs.2-8) and the commented templates for SPHERA (2022, [1]) input files (Sec.9.2). The tutorials are discussed in Sec.9.3.

9.1. Installation

SPHERA (2022, [1]) is distributed for Linux OS on a dedicated Git (2022, [68]) repository on github.com. The submission of the command line "make" from a Linux terminal under the folder "src" compiles the source code. The Makefile of SPHERA (2022, [1]) permits to generate 32 different executables depending on the user choice regarding 5 Makefile variables. They are associated with 3 active preprocessor macros (Table 9.1). The only mandatory argument in the command line to launch an executable of SPHERA is the test-case name defined by the user.

Makefile variable	**Suggested value**	**Executable name: SPHERA_j_abcdef**	**Description**
VERSION	V10_UserInitials_CommitData	j=VERSION	Master subversion compiled by the user
COMPILER	ifort	a=1	Compiler
	gfortran	a=2	
EXECUTION	optimized	b=1	Execution mode
	debug	b=2	
PARALLELIZATION	OMP	c=1	parallel simulations
	NO	c=2	sequential simulations
PD_SPACE	-DSPACE_3D	d=3	3D (macro)
	-DSPACE_2D	d=2	2D (macro)
PD_KTGF	-DKTGF	e=1	KTGF scheme (dense granular flows) (inactive macro)
	(blank space)	e=0	no KTGF scheme (inactive macro)
PD_SOLID_BODIES	-DSOLID_BODIES	f=1	Body transport (macro)
	(blank space)	f=0	No body transport

Table 9.1. Makefile variables; macros for preprocessor directives; executable names (Makefile of SPHERA, 2022, [1]).

9.2. Commented templates of the input files of SPHERA v.10.0.0

The commented templates of the input files of SPHERA (2022, [1]) are reported in the code repository under the folder “input/template”. They are apportioned among the folders of Table 9.2. The comments within each template define the input quantities, describe the meaning of their possible values, indicate suggested and default values and explain how the input values have to fill the template. If the name of an input-file template contains the string “xx” or “0x” then it must be copied and pasted in case of multiple elements (e.g., more monitoring lines).

Input file folder	**Input file folder**	**Input file folder**
01_title	08_KTGF	15_vtu_output_timing
02_domain	09_media	16_monitoring_points
03_vertices	10_body_dynamics	17_monitoring_lines
04_lines	11_time_integration_and_numerical_data	18_monitoring_sections
05_faces	12_gravity_and_reference_pressure	19_electrical_substations
06_boundaries_and_fluid_bodies	13_restart	20_CLC
07_DBSPH	14_output_timing	

Table 9.2. Folders of the input files of SPHERA v.10.0.0 (2022, [1]) containing the commented input-file templates.

9.3. Tutorials

SPHERA (2022, [1]) was validated or applied to 47 available tutorials, each one having possible variants. Some of the test cases are published on International Journals. Other minor tutorials only represent very simple configurations. Tutorials are divided in three groups. The first one (folder “input/tutorials/release_v_9_0_0”) reports 45 tutorials associated with the release v.9.0.0, with 3 new test cases following the distribution v.9.0.0. The second group (folder “input/tutorials/subversions_between_v9_v10”) contains 4 updates to the tutorials of v.9.0.0 and 4 new tutorials. They are explicitly associated with a code subversion v.9.x.y expressed in terms of “git commit code”. The third group (folder “input/tutorials/release_v_10_0_0”) reports 4 new tutorials for v.10.0.0 plus one tutorial update. The following sub-sections briefly recall the publications containing the description of the tutorials.

9.3.1. “body_body_impacts”

This tutorial is completely described in Amicarelli et al. (2015, [4]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.2. “body_boundary_impacts”

This tutorial is completely described in Amicarelli et al. (2015, [4]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.3. “dam_breach_ICOLD_trunks”

This tutorial is described in Amicarelli & Agate (2014, [69]), Amicarelli & Agate (2015, [70]) and Amicarelli et al. (2020, [5]). Here, the tutorial is represented by means of the mixture model for dense granular flows of Amicarelli et al. (2017, [48]).

9.3.4. “db_2bodies”

This tutorial is completely described in Amicarelli et al. (2015, [4]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.5. “db_2D_demo”

This is a very simple and very fast tutorial representing a 2D dam break (over a flat bottom) at a rough spatial resolution. Amicarelli et al. (2020, [5]) completely describes this test case. This project report is Open-Access and also includes a synthetic English version.

9.3.6. “db_Alpe_Gera”

This tutorial is completely described in Amicarelli & Paggi (2019, [71]) and Amicarelli et al. (2020, [5]).

9.3.7. “db_Alpe_Gera_Lanzada_substations”

This tutorial is completely described in Amicarelli (2021, [60]). This project report is Open-Access and also includes a synthetic English version.

9.3.8. “db_body_exp_UniBas”

This tutorial is completely described in Amicarelli et al. (2015, [4]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.9. “db_ICOLD”

This tutorial is described in Amicarelli & Agate (2014, [69]) and Amicarelli et al. (2020, [5]).

9.3.10. “db_multi_body”

This tutorial is completely described in Amicarelli et al. (2015, [4]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.11. “db_squat_obstacle”

This tutorial is described in Amicarelli et al. (2013, [9]). The paper version available on ResearchGate might help in case the published version is unavailable. Here, the surface mesh is generated by SnappyHexMesh (OpenCFD Ltd, 2018, [58]) and SPHERA source code improvements for the DB-SPH treatment (Amicarelli & Agate, 2015, [70]) apply.

9.3.12. “db_tall_obstacle”

This tutorial is described in Amicarelli et al. (2013, [9]). The paper version available on ResearchGate might help in case the published version is unavailable. Here, the surface mesh is generated by SnappyHexMesh (OpenCFD Ltd, 2018, [58]) and SPHERA source code improvements for the DB-SPH treatment (Amicarelli & Agate, 2015, [70]) apply.

9.3.13. “dike_breach_2D_expSchHag12JHR_ID22”

This tutorial is completely described in Amicarelli et al. (2016, [46]) and Amicarelli et al. (2020, [5]).

9.3.14. “edb_2D_demo”

This tutorial of SPHERA v.10.0.0 (2022, [1]) was used as a reference test case to carry out a sensitivity analysis in Manenti et al. (2018, [3]). This paper is Open-Access and also describes in detail the present test case.

9.3.15. “edb_2D_FraCap02”

This tutorial is described in Amicarelli et al. (2017, [48]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.16. “edb_2D_FraCap02_Taipei”

This tutorial is described in Amicarelli et al. (2017, [48]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.17. “edb_2D_Spi05”

This tutorial is described in Amicarelli et al. (2017, [48]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.18. “edb_ICOLD”

This tutorial is described in Amicarelli et al. (2017, [48]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.19. "edb_KarlSand"

This tutorial is described in Amicarelli et al. (2017, [48]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.20. "edb_Pon10"

This tutorial is described in Amicarelli et al. (2017, [48]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.21. "floating_cube_stability"

This is a very simple and demonstrative tutorial: a solid cube is leaned on still water (rough resolution). Amicarelli & Agate (2014, [69]) and Amicarelli et al. (2020, [5]) describe this test case.

9.3.22. "flushing_2D"

This tutorial is described in Amicarelli & Agate (2014, [69]) and Amicarelli et al. (2020, [5]). Here, the tutorial is represented by means of the mixture model for dense granular flows of Amicarelli et al. (2017, [48]).

9.3.23. "flushing_3D"

As a 3D version of the previous tutorial, this test case is available on Amicarelli et al. (2014, [72]) and Amicarelli et al. (2020, [5]). The first publication is an Open-Access project report and includes a synthetic English version.

9.3.24. "jet_plate"

The 7 variants of this tutorial are described in Amicarelli et al. (2013, [9]) and Amicarelli et al. (2015, [4]). Here the DBSPH boundary scheme is treated with automated procedures (Amicarelli & Agate, 2015, [70]) and the scheme for body transport is corrected in terms of no-slip conditions for fluid-body interactions (RSE developments reported in Paggi et al., 2021, [56]).

9.3.25. "rectangular_side_weir_Fr_0_491"

This tutorial is described in Amicarelli (2018, [73]) and Amicarelli et al. (2020, [5]). The first publication is an Open-Access project report and includes a synthetic English version.

9.3.26. "San_Fernando_Lower_van_Norman_dam_liquefaction"

This tutorial is described in Amicarelli (2016, [46]) and Amicarelli et al. (2020, [5]). Here, the tutorial is represented by means of the mixture model for dense granular flows of Amicarelli et al. (2017, [48]).

9.3.27. "sloshing_tank_TbyTn_0_78"

This tutorial is described in Amicarelli et al. (2013, [9]). The paper version available on ResearchGate might help in case the published version is unavailable. Here, the surface mesh is generated by SnappyHexMesh (OpenCFD Ltd, 2018, [58]) and SPHERA source code improvements for the DB-SPH treatment (Amicarelli & Agate, 2015, [70]) apply.

9.3.28. "sloshing_tank_TbyTn_1_07"

This tutorial is described in Amicarelli et al. (2013, [9]). The paper version available on ResearchGate might help in case the published version is unavailable. Here, the surface mesh is generated by SnappyHexMesh (OpenCFD Ltd, 2018, [58]) and SPHERA source code improvements for the DB-SPH treatment (Amicarelli & Agate, 2015, [70]) apply.

9.3.29. “spherical_Couette_flows”
This tutorial is described in Amicarelli et al. (2022, [7]).

9.3.30. “SPH_udb_exp_Kim2015HYDROL”
This tutorial is described in Amicarelli (2021, [60]).

9.3.31. “submerged_landslide”
This tutorial is described in Amicarelli & Agate (2014, [69]) and Amicarelli et al. (2020, [5]). Here, the tutorial is represented by means of the mixture model for dense granular flows of Amicarelli et al. (2017, [48]).

9.3.32. “still_water_tank”
This tutorial is a very simple and rough-resolution 2D test case on hydrostatic conditions.

9.3.33. “wave_motion_for_WaveSAX”
This tutorial is described in Amicarelli et al. (2022, [14]).

9.3.34. “wedges_falls_on_still_water”
This tutorial is described in Amicarelli et al. (2015, [4]). The paper version available on ResearchGate might help in case the published version is unavailable.

9.3.35. “2D_Vajont_experiment”
The reference to this tutorial is Manenti et al. (2018, [3]).

9.3.36. “spillway_withpiers”
The reference to this tutorial is Agate & Guandalini (2010, [74]).

9.3.37. “tsunami_benchmark1_ISEC”
The reference to this tutorial is Guandalini et al. (2014, [75]).

9.3.38. “dyke_failure_Kagerplassen_2015”
The reference to this tutorial is Amicarelli et al. (2022, [7]).

9.3.39. “stream_flat_bottom”
The reference to this tutorial is SPHERA (2022, [1]) github repository.

9.3.40. “douf_Vajont”
The reference to this tutorial is Amicarelli et al. (2022, [36]).

9.3.41. “uf_es_Calenzano”
The reference to this tutorial is SPHERA (2022, [1]) github repository.

9.3.42. “dam_break_3D_demo”
The reference to this tutorial is SPHERA (2022, [1]) github repository.

9.3.43. “Couette_sf”
The reference to this tutorial is Amicarelli et al. (2022, [14]).

9.3.44. “water_shipping”
The reference to this tutorial is Amicarelli et al. (2022, [14]).

9.3.45. "WSI_WAVESAX"

The reference to this tutorial is Amicarelli et al. (2022, [14]).

10. OVERVIEW OF THE PROGRAM UNITS

The present section reports a synthetic description of the program units (Sec.10.1-Sec.10.40) and some non-mandatory rules for the style format (Sec.10.41).

10.1. Folders of the program units

The folders and sub-folders of SPHERA source code are briefly described in Table 10.1.

Folder	**Sub-folder**	**Synthetic Description**
BC		Generic Boundary Conditions
BE_mass		Mass-related balance equations
BE_momentum		Momentum Equation
Body_Transport		Scheme for the transport of solid bodies
	BE_body_momentum	Solid Momentum Equation
	BE_mass	Contributions of solid bodies to the Mass-related balance equations
	BE_momentum	Contributions of solid bodies to the Momentum Equation
	IC	Initial conditions for solid bodies
	Neighbouring_Search	SPH neighbouring search for body particles
	Post_processing	Post-processing for solid bodies
	Time_Integration	Time integration for solid bodies
CLC		CLC land-use classes
DB_SPH		Scheme for mobile surface walls
EoS		Equation of State
Erosion_Criterion		Erosion criterion
Geometry		Geometry procedures
IC		Initial Conditions
KTGF_tools		Scheme for dense granular flows
Main_algorithm		Main algorithm
Memory_I_O		Memory Input/Output management
	allocate_de_derived_types_a allocate_de_derived_types_b	Allocations/deallocations of derived types
	allocate_de_preset_types	Allocations/deallocations of preset types
Modules		Fortran modules
Neighbouring_Search		SPH neighbouring search
Post_processing		Post-processing
	2D_fields	Post-processing 2D synthetic fields
	elapsed_time	Post-processing the elapsed time
	Monitors	Post-processing at SPH monitors
Pre_processing		Pre-processing
	reading_inp	Reading the input files
SA_SPH		Scheme for fixed surface walls
	BE_mass	Contributions of the fixed surface walls to the Mass-related balance equations
	BE_momentum	Contributions of the fixed surface walls to the Momentum Equation

	IC	Initial Conditions for the fixed surface walls
	integrals_IC	Assessment of the SASPH integrals at the initial time
	integrals_time_steps	Interpolations of SASPH integrals during the time steps
	Neighbouring_Search	SPH neighbouring search for fixed surface walls
	Time_integration	Contribution to the stability criteria from fixed surface walls
	Turbulence	Wall function for fixed surface walls
Strings		String management
Time_Integration		Time integration

Table 10.1. Folders hosting the program units of SPHERA (2022, [1]) under “src”.

10.2. Program units for Generic Boundary Conditions

The program units for Generic Boundary Conditions are briefly described in Table 10.2.

Program unit	**Synthetic Description**
BC_zmax_anyt.f90	Dirichlet's BC for the maximum fluid height. The selected zone is continuously filled with SPH particles from the non-stationary free surface height to the given input height "zmax", under hydrostatic or homogeneous pressure conditions. Only active in 3D with SASPH. No need to impose pressure to the particles already in the "z_max" zone (ie., the particles not emitted). Each "zmax" zone needs at least 5 adjacent open sections in the input files in order to set BCs on the fluid depth. However, a "zmax" zone can be used for a more generic purpose. The associated DEM-DTM or bottom has to be locally a uniform and isotropic Cartesian grid.
BC_zmax_t0.f90	Selection of the zone vertices at the beginning of the simulation for Dirichlet's BCs for the maximum fluid height. Only active in 3D with SASPH boundary treatment. The associated DEM-DTM or bottom has to be locally a uniform and isotropic Cartesian grid (in plan view). Assessment of the minimum local bottom height (influential in the presence of dry bottom).
CancelOutgoneParticles_2D.f90	To count and delete the outgoing particles on boundaries of type "leve", "flow", "velo", "crit", "open".
CancelOutgoneParticles_3D.f90	To count and delete the outgoing particles on boundaries of type "leve", "flow", "velo", "crit", "open". Deletion occurs in 2 different ways: a) If the particle belongs to a particle zone (maxzone) with the highest index (the only zone where both particle number reduction and increase are allowed), then the outgoing particle (npi) is replaced by the last particle (nag) in the particle array pg, and the total number of particles becomes nag=nag-1; simultaneously, the index of the last particle of the zone is changed (Partz(maxzone)%limit(2)). b) Otherwise, simply pg(npi)%cella = 0 (particle out of the domain boundaries).
domain_edges.f90	to find the domain edges (x, y and z maximum and minimum values)
FindFrame.f90	To find the rectangular frame of a boundary
FindLine.f90	To find the rectangular frame of a line
fluid_particle_imposed_kinematics.f90	Possible imposed velocities for the fluid particles
GenerateSourceParticles.f90	Time-dependent generation of new particles at the inlet sections. The actual position of the generated particles depends on the ratio between the residual time since the last emission step and the particle size (for null residual time the particle barycentre is located upstream the inlet section of dx/2; the maximum residual is associated with particle barycentres located dx/2 downstream the inlet section).

inlet_sections.f90	Elaboration of the quantities of the inlet sections. The first call to this program unit is mainly relevant for the SASPH boundary integrals and the neighbouring search algorithm. They are not time-dependent and refer to the maximum inlet fluid depth, depending on the initial vertex positions, not on the inlet time series. The following calls are time-dependent and relevant to the SPH inlet particles.
NormFix.f90	Minor program unit
VelLaw.f90	To impose an input kinematics to particles
z_FS_max_9p_stencil.f90	To assess the maximum free surface height in the 9-point stencil around the current vertex around which the extrusion takes place.

Table 10.2. Program units for Generic Boundary Conditions (SPHERA, 2022, [1]).

10.3. Program units for Mass-related balance equations

The program units for Mass-related balance equations are briefly described in Table 10.3.

Program unit	**Synthetic Description**
B_ren_divu_inversion.f90	Inversion of the renormalization matrix of an input fluid particle for the approximation of the velocity-divergence term
Continuity_Equation.f90	Continuity Equation RHS, included explicit ALE1 and ALE2 terms. Computation of velocity gradients and the second invariant of the strain-rate tensor (with 2D renormalization).
PressureSmoothing_2D.f90	Partial smoothing for pressure (Di Monaco et al., 2011), also with DB-SPH boundary treatment scheme. This subroutine is not just the formal 2D extension of "PressureSmoothing_3D.f90":
PressureSmoothing_3D.f90	Partial smoothing for pressure (Di Monaco et al., 2011), also with DB-SPH boundary treatment scheme. This subroutine is not just the formal 3D extension of PressureSmoothing_2D: differences are appreciable.

Table 10.3. Program units for Mass-related balance equations (SPHERA, 2022, [1]).

10.4. Program units for Momentum Equation

The program units for Momentum Equation are briefly described in Table 10.4.

Program unit	**Synthetic Description**
ALE3_term_momentum.f90	ALE3 term: computation and contribution to the Momentum Equation
B_ren_gradp_inversion.f90	Inversion of the renormalization matrix of an input fluid particle for the approximation of the pressure-gradient term
inter_EqMoto.f90	Computation of the momentum equation RHS (with DB-SPH boundary treatment scheme, Shepard's coefficient and gravity are added at a later stage) merged with the control-volume velocity equation, included contributions to the ALE1 velocity increment.
RHS_momentum_equation.f90	Right Hand Side of the momentum equation merged with the control-volume velocity equation, included contributions to the ALE1 velocity increment.
velocity_smoothing_2.f90	Partial velocity smoothing (part 1). Full velocity smoothing within the "zmax" zones (part 1).
velocity_smoothing.f90	Partial smoothing of the fluid velocity field (part 2). Full velocity smoothing within the "zmax" zones (part 2).
velocity_smoothing_SA_SPH_2D.f90	SASPH 2D contribution to the partial velocity smoothing
velocity_smoothing_SA_SPH_3D.f90	SASPH 3D contribution to the partial velocity smoothing
viscomon.f90	Monaghan (2005) artificial viscosity term. It is also active for separating particles. Volume viscosity term is neglected in the momentum equation.
viscomorris.f90	To compute the volume inter-particle contributions to the shear

	viscosity term in the momentum equation (Morris, 1997, JCP). Both interactions "fluid particle - fluid particle" and "fluid particle - semi-particle" (DBSPH) are considered.

Table 10.4. Program units for Momentum Equation (SPHERA, 2022, [1]).

10.5. Program units for the Solid Momentum Equation

The program units for Solid Momentum Equation are briefly described in Table 10.5.

Program unit	**Synthetic Description**
body_boundary_for_sliding_friction_normal_reaction.f90	Body-boundary interactions. Intermediate computations for the sliding friction force and the normal reaction force under sliding. Generic "body particle - frontier" interaction.
Gamma_boun.f90	Interpolative function defined by Monaghan (2005) for boundary force particles (Amicarelli et al.,2015,CAF).
RHS_body_dynamics.f90	To estimate the RHS of the body dynamics equations (Amicarelli et al., 2015, CAF; Amicarelli et al., 2020, CPC; Amicarelli et al., 2022, IJCFD)

Table 10.5. Program units for the Solid Momentum Equation (SPHERA, 2022, [1]).

10.6. Program units for the contributions of solid bodies to the Mass-related balance equations

The program units for the contributions of solid bodies to the Mass-related balance equations the are briefly described in Table 10.6.

Program unit	**Synthetic Description**
body_particles_to_continuity.f90	Contributions of the body particles to the continuity equation, included explicit BODY BC ALE terms and ALE implicit terms in CE.
body_to_smoothing_pres.f90	Contributions of body particles to pressure partial smoothing and update of the mirror body-particle pressure (Amicarelli et al., 2015, CAF; Amicarelli et al., 2020, CPC; Amicarelli et al., 2022, IJCFD)

Table 10.6. Program units for the contributions of solid bodies to Mass-related balance equations (SPHERA, 2022, [1]).

10.7. Program units for the contributions of solid bodies to the Momentum Equation

The program units for the contributions of solid bodies to the Momentum Equation are briefly described in Table 10.7.

Program unit	**Synthetic Description**
body_p_max_limiter.f90	Limiter for the body-particle maximum pressure (Amicarelli et al., 2015, CAF).
body_pressure_mirror_interaction.f90	Computation of the mirror pressure for a generic fluid-body inter-particle interaction (Amicarelli et al., 2015, CAF; Amicarelli et al., 2020, CPC; Amicarelli et al., 2022, IJCFD).
body_to_smoothing_vel.f90	Contributions of body particles to velocity partial smoothing (Amicarelli et al., 2015, CAF).

Table 10.7. Program units for the contributions of solid bodies to the Momentum Equation (SPHERA, 2022, [1]).

10.8. Program units for Initial conditions for solid bodies

The program units for Initial conditions for solid bodies are briefly described in Table 10.8.

Program unit	**Synthetic Description**
IC_CAE_bodies.f90	Initial Conditions for the CAE-made solid bodies
IC_CAE_body_particles.f90	Initial Conditions for the CAE-made solid body particles
IC_solid_bodies.f90	Initial Conditions for solid bodies. Both handmade and CAE-made

	bodies are considered.
initial_fluid_removal_in_solid_bodies.f90	Initial removal of possible SPH fluid particles within solid bodies (due to design) at the beginning of the simulation (initial conditions for fluid particle positions in case of Fluid - Structure interactions).

Table 10.8. Program units for Initial conditions for solid bodies (SPHERA, 2022, [1]).

10.9. Program units for SPH neighbouring search for body particles

The program units for SPH neighbouring search for body particles are briefly described in Table 10.9.

Program unit	**Synthetic Description**
proxy_normals_for_body_particles.f90	Computation of the array «proxy_normal_bp_f» of the indices of the neighbouring surface body particles providing the normal to the "body-particle - fluid-particle" interactions (only for no-slip conditions) and the surface boundary velocity and acceleration (any slip condition). Body-particle mirror pressure (only at the first time step "it_start" of the current simulation).

Table 10.9. Program units for SPH neighbouring search for body particles (SPHERA, 2022, [1]).

10.10. Program units for post-processing for solid bodies

The program units for post-processing for solid bodies are briefly described in Table 10.10.

Program unit	**Synthetic Description**
Body_dynamics_output.f90	.txt output files for body transport in fluid flows

Table 10.10. Program units for post-processing for solid bodies (SPHERA, 2022, [1]).

10.11. Program units for Time integration for solid bodies

The program units for Time integration for solid bodies are briefly described in Table 10.11.

Program unit	**Synthetic Description**
time_integration_body_dynamics.f90	Time integration for body transport in fluid flows

Table 10.11. Program units for Time integration for solid bodies (SPHERA, 2022, [1]).

10.12. Program units for CLC land-use classes

The program units for CLC land-use classes are briefly described in Table 10.12.

Program unit	**Synthetic Description**
CLC_pre_processing.f90	Pre-processsing of the CLC (Corine Land Cover) input data
neighbouring_hcell_CLCp.f90	Contribution of the current polygon to the neighbouring search linking the cells of the horizontal background grid with the neighbouring CLC polygons
vertex_usage_code.f90	Computation of the "vertex usage code" of a triangular face, provided the occurrences of its 3 vertices in the triangulated polygon the current face belongs to.
z0_CLC.f90	Assessment of the 2D field of the CLC class. Assessment of the 2D field of the roughness characteristic length (z0) as function of the CLC class. Writing of a unique 2D output file on the horizontal background grid with the following quantities: x, y, CLC class, z0.
z0_CLC_table.f90	Assignment of the z0 value of the current horizontal background cell as function of the CLC class

Table 10.12. Program units for CLC land-use classes (SPHERA, 2022, [1]).

10.13. Program units for the Scheme for mobile surface walls

The program units for the Scheme for mobile surface walls are briefly described in Table 10.13.

Program unit	**Synthetic Description**
adjacent_faces_isolated_points.f90	Provided 2 adjacent triangular/quadrilateral faces, it finds at least 2 vertices not in common, at least one per face. They are ID_face1_iso and ID_face2_iso. In case the faces are not adjacent, then false_hyp=.true.
BC_wall_elements.f90	Wall element density and pressure (Amicarelli et al., 2013, IJNME). Shepard correction for the kinematic viscosity of the semi-particles. Shepard correction for the velocity gradient (times shear viscosity) of the semi-particles in the Viscous Sub-Layer of the Surface Neutral Boundary Layer.
DBSPH_BC_shear_viscosity_term.f90	Computation of the contributions to the numerator of the boundary shear viscosity term in DB-SPH-NS.
DBSPH_find_close_faces.f90	Finding the adjacent surface elements of a given surface element, both using 3D -triangular elements- and 2D -quadrilateral raw elements- configurations (DB-SPH).
DBSPH_IC_surface_elements.f90	Initialization of wall surface elements (Amicarelli et al., 2013, IJNME).
DBSPH_inlet_outlet.f90	Impose boundary conditions at the inlet and outlet sections (DB-SPH boundary treatment scheme).
DBSPH_kinematics.f90	Imposing input kinematics for the DB-SPH elements (linear interpolation of input data).
DBSPH_velocity_gradients_VSL_SNBL.f90	Computation of the velocity gradients in the Viscous Sub-Layer of the Surface Neutral Boundary Layer. The gradients are used in the DB-SPH BC shear viscosity term (DB-SPH-NS). For wall elements, the numerator and the denominator (wall element Shepard coefficient without contributions from semi-particles) are updated independently. Their ratio is computed in "DBSPH_BC_shear_viscosity_term": here are summed their contributions. To compute the kinematic viscosity of the semi-particles (before Shepard correction). Contributions to the discrete Shepard coefficient of wall elements depending on fluid particles (not on semi-particles).
Gradients_to_MUSCL_boundary.f90	0th-order consistency estimation of velocity and density gradients for the MUSCL reconstruction (to feed the Partial Linearized Riemann Solver; Amicarelli et al., 2013, IJNME).
Gradients_to_MUSCL.f90	Estimation of the boundary terms for the MUSCL reconstruction scheme (DB-SPH), in case they are required in input.
Import_ply_surface_meshes.f90	To import the surface meshes (generated by SnappyHexMesh -OpenFOAM-), as converted by Paraview into .ply files. This subroutine is mandatory and activated only for the DB-SPH boundary treatment scheme. In 3D, SPHERA (DBSPH) works with triangular faces, in 2D with quadrilateral faces. Input .ply files must be triangular/quadrilateral/pentagonal/hexagonal meshes in 3D or square meshes in 2D.
removing_DBSPH_fictitious_reservoirs.f90	Removing the fictitious reservoirs used for DB-SPH initialization (Amicarelli et al., 2013, CAF)
semi_particle_volumes.f90	To compute the semi-particle shape coefficients and volumes
wall_elements_pp.f90	Smoothing wall element values for post-processing. Post-processing the wall surface element values (provided a selected region). Post-processing the hydrodynamic normal force on DBSPH surface elements (provided a selected region). Post-processing the wall surface element values (provided selected element IDs).
wavy_inlet.f90	To provide a very slightly wavy flow at the inlet section. Each particle layer is staggered by 0.5dx with respect to the previous and the following ones, which are instead aligned each other. This numerical feature reduces the SPH truncation errors at the DB-SPH inlet sections. A white noise is also added. (Amicarelli

	et al., 2013, IJNME).

Table 10.13. Program units for the Scheme for mobile surface walls (SPHERA, 2022, [1]).

10.14. Program units for the Equation of State

The program units for the Equation of State are briefly described in Table 10.14.

Program unit	**Synthetic Description**
EoS_barotropic_linear.f90	Barotropic linear Equation of State (EoS) for pressure estimation and its inverse for density estimation
EoS_fluid_loop.f90	Loop over the fluid particles to apply the Equation of State (EoS) to assess pressure

Table 10.14. Program units for the Equation of State (SPHERA, 2022, [1]).

10.15. Program units for the Erosion criterion

The program units for the Erosion criterion are briefly described in Table 10.15.

Program unit	**Synthetic Description**
compute_k_BetaGamma.f90	To compute k_BetaGamma=teta_c/teta_c,00. k_BetaGamma is the ratio between Shields critical non-dimensional stress for a generic 3D slope (teta_c) and its analogous value defined by Shields diagram (teta_c,00) on flat bed.
fixed_bed_slope_limited.f90	Forced deposition (or no erosion) for particles at least 2h below the fixed bed (as it is defined in the associated column) during the same time step: i.e., the maximum slope of the fixed bed is 2h/2h. This avoids eventual too fast propagation of erosion along the vertical (erosion is an interface phenomenon).
Shields.f90	3D erosion criterion based on the formulation of both Shields-van Rijn 2D criterion and Seminara et al. (2002) 3D criterion. 2D Shields erosion criterion based on pure fluid - fixed bed interactions (Manenti et al., 2012, JHE). Extension for bed-load transport layer - fixed bed interactions (Amicarelli & Agate, 2014, SPHERIC). Extension to the third dimension (Amicarelli & Agate, 2014, SPHERIC). k=3d_90 (Manenti et al., 2012, JHE; Amicarelli & Agate, 2014, SPHERIC). Shields threshold for low Re* (Amicarelli & Agate, 2014, SPHERIC).

Table 10.15. Program units for the Erosion criterion (SPHERA, 2022, [1]).

10.16. Program units for the geometry procedures

The program units for the geometry procedures are briefly described in Table 10.16.
Some details on the program unit “distance_point_line_3D” are reported in the following.

Program unit	**Synthetic Description**
area_hexagon.f90	Computation of the area of a generic hexagon from the coordinates of its vertices.
area_pentagon.f90	Computation of the area of a generic pentagon from the coordinates of its vertices
area_quadrilateral.f90	Computation of the area of a generic quadrilateral from the coordinates of its vertices.
area_triangle.f90	Computation of the area of a generic triangle, provided the coordinates of its vertices
dis_point_plane.f90	Computation of the distance between a point and a plane
distance_point_line_2D.f90	Computation of the distance between a point and a plane
distance_point_line_3D.f90	Computation of the distance between a point and a line in 3D
face_shared_by_tet_cells.f90	Check if 2 tetrahedrcal cells have 1 face in common. “2 cells with 3 points in common” is equivalent to “2 cells with 1 face in common”.

IsPointInternal.f90	Checking wheather a point with local normal coordinates csi(1:3) is internal to a given face, whose type code is fk (=1 triangle, =2 quadrilateral), or not. This procedure is based on the subroutine "LocalNormalCoordinatesGiven". They will be replaced by the subroutine "point_inout_convex_non_degenerate_polygon" (already available in SPHERA), which is more effective and can be applied to any polygon (it already works for triangles and quadrilaterals). Only in 3D.
line_plane_intersection.f90	Computation of the intersection point, if unique, between a line and a plane.
LocalNormalCoordinates.f90	Given the local Cartesian coordinates PX(1:2) of a point P laying on the plane of the boundary face nf, this procedure assigns to csi(1:3) the non-Cartesian coordinates in a different local reference systems, whose axis are aligned with the face sides. Scaling provides coordinate values to go from 0 to 1 for internal points. This procedure applies for triangular faces and simplier works for rectangular faces (not for quadrilateral faces).
Matrix_Inversion_3x3.f90	Computation of the inverse of a 3x3 matrix. It is also called in 2D.
MatrixProduct.f90	Returning in CC the product between matrices AA and BB. nr: number of rows of AA and CC nc: number of columns of BB and CC nrc: number of columns of AA = number of rows of BB
MatrixTransposition.f90	Matrix transposition
point_inout_convex_non_degenerate_polygon.f90	Test to evaluate if a point lies inside or strictly outside a polygon. A point is internal to the polygon if its distances from the lines passing for the polygon sides (no matter about the number of sides, but they must be taken in either a clockwise or an anti-clockwise order), have all the same sign of a generic polygon vertex not belonging to the selected side -a null distance is always a positive test for internal points-). The maximum number of polygon sides is now equal to 6 (triangles, quadrilaterals, pentagons and hexagons can be treated). Polygons must be convex and non-degenerate (a n-side polygon should have n vertices, not more).
point_inout_hexagon.f90	Test to evaluate if a point lies inside or strictly outside a generic hexagon. The hexagon is partitioned into 4 triangles (P1P2P6,P2P5P6,P2P3P5,P3P4P5). A point is internal to the hexagon if it is internal to one of its triangles.
point_inout_pentagon.f90	Test to evaluate if a point lies inside or strictly outside a generic pentagon. The pentagon is partitioned into 3 triangles (P1P2P5,P2P3P5,P3P4P5). A point is internal to the pentagon if it is internal to one of its triangles.
point_inout_quadrilateral.f90	Test to evaluate if a point lies inside or strictly outside a generic quadrilateral. The quadrilateral is partitioned into 2 triangles (P1P2P3,P1P3P4). A point is internal to the quadrilateral if it is internal to one of the triangles.
quadratic_equation.f90	To solve a quadratic equation
reference_system_change.f90	Transformation of coordinates, expressed in a new reference system.
three_plane_intersection.f90	Computation of the intersection of 3 planes. It is also called in 2D
Vector_Product.f90	Cross product of two vectors
vector_rotation_axis_angle.f90	Provided 2 vectors, this subroutine computes the rotation axis and the rotation angle which allow rotating from the unit vector aligned with the first vector to the unit vector aligned with the second vector.
vector_rotation_Euler_angles.f90	3D rotation of a given vector, provided the vector of Euler's

	angles (3D). It is als ocalled in 2D.
vector_rotation_Rodrigues.f90	3D rotation of a given vector, provided the rotation axis and the rotation angle, based on Rodrigues formula. It is also called in 2D

Table 10.16. Program units for the geometry procedures (SPHERA, 2022, [1]).

10.17. Program units for the Initial Conditions

The program units for the Initial Conditions are briefly described in Table 10.17.

Program unit	**Synthetic Description**
defcolpartzero.f90	Particle colours for visualization purposes
GeneratePart.f90	Particle positions (initial conditions)
IsParticleInternal2D.f90	To check whether a particle is internal to the 2D domain
IsParticleInternal3D.f90	To check whether a particle is internal to a 3D volume (domain/zone) or not. It checks if point "Px" is internal to the perimeter "mib". It returns ".true." (positive check) or ".false.". The perimeter can be both convex or concave. The input point is internal to the zone if the number of both "faces intercepted by the vertical (passing for the input point) above the input point" and "faces intercepted by the vertical (passing for the input point) below the input point" are odd.
particle_position_extrusion.f90	Particle positions extruded from DEM-DTM or any 3D solid bottom
particles_in_out_dams.f90	To test if the SPH particles (preliminarily designed at this stage) lie out of (test_dam=0) or within (test_dam=1) the volume of a dam (initial conditions).
pos_plus_white_noise.f90	Add a white noise to a particle position (initial conditions)
SetParticleParameters.f90	Setting initial particle parameters
SetParticles.f90	Particle coordinates (initial conditions)
SubCalcPreIdro.f90	To initialize pressure and density fields. Hydrostatic pressure profiles or uniform pressure or radial pressure as initial conditions. Hydrostatic profiles are formally correct only in the presence of maximum 2 fluid media along the vertical. Update/initialization of the particle volume
z_min_max_DEM_DTM_9p_stencil.f90	To assess the minimum/maximum height at the 9 vertices of the DEM-DTM (or any 3D solid bottom) around the given DEM-DTM vertex (included). The minimum local bottom height is useful for the fluid body extrusion from topography (IC). The maximum local bottom height is useful for the BCs of the "zmax" zones (where the fluid height is imposed).

Table 10.17. Program units for the Initial Conditions (SPHERA, 2022, [1]).

10.18. Program units for the Scheme for dense granular flows

The program units for the Scheme for dense granular flows are briefly described in Table 10.18.

Program unit	**Synthetic Description**
initialization_fixed_granular_particle.f90	To initialize the most of the fixed SPH mixture particles (bed-load transport).
KTGF_update.f90	To update some quantities associated with dense granular flows (Kinetic Theory of Granular Flow under the packing limit, Amicarelli et al., 2017, IJCFD)
mixture_viscosity.f90	To compute the frictional viscosity and the mixture viscosity for dense granular flows (KTGF packing limit). (Amicarelli et al., 2017, IJCFD)

Table 10.18. Program units for the Scheme for dense granular flows (SPHERA, 2022, [1]).

10.19. Program units for the Main algorithm

The program units for the Main algorithm are briefly described in Table 10.19.
Table 10.20 lists the program units called by the program unit “time_step_loop”, when the time integration scheme Leapfrog is activated (SPHERA). The call order is highlighted.

Program unit	**Synthetic Description**
Gest_Trans.f90	Introductory procedure for the main algorithm. Writing of the ".vtk" geometry file.
sphera.f90	Main program unit
time_step_loop.f90	loop over the simulation time steps

Table 10.19. Program units for the Main algorithm (SPHERA, 2022, [1]).

Program unit	**Program unit (direct calls)**	**Program unit (1st order indirect calls)**	**Program unit (2nd order ind. calls)**
time_step_loop			
	time_step_duration		
	fluid_particle_imposed_kinematics		
	RHS_momentum_equation		
		inter_EqMoto	
			viscomon
			viscomorris
		AddBoundaryContributions_to_ME3D	
			wall_function_for_SASPH
		AddElasticBoundaryReaction_3D	
	RHS_body_dynamics		
		body_pressure_mirror	
	Leapfrog_momentum		
	time_integration_body_dynamics		
	velocity_smoothing		
		body_to_smoothing_vel	
		velocity_smoothing_SA_SPH_3D	
			wall_function_for_SASPH
	velocity_smoothing_2		
	Leapfrog_trajectories		
	CancelOutgoneParticles_3D		
	GenerateSourceParticles		
	BC_zmax_anyt		
		z_FS_max_9p_stencil	
	OrdGrid1		
	NormFix		
	CalcVarLength		
	liquid_particle_ID_array		
	velocity_smoothing		
	velocity_smoothing_2		
	ComputeBoundaryDataTab_3D		
		FindCloseBoundaryFaces3D	
		ComputeBoundaryVolumeIntegrals_P0	
	KTGF_update		
		initialization_fixed_granular_particle	
		Shields	
			fixed_bed_slope_limited
			compute_k_BetaGamma
	liquid_particle_ID_array		
	Continuity_Equation		
		body_particles_to_continuity	
	SASPH_continuity		
		AddBoundaryContribution_to_CE3D	
	Leapfrog_continuity		
	CalcPre		
	body_pressure_mirror		
	PressureSmoothing_3D		
		body_to_smoothing_pres	
	body_pressure_postpro		
	mixture_viscosity		

	time_step_post_processing		

Table 10.20. Program units called by the program unit "time_step_loop", when the time integration scheme Leapfrog is activated (SPHERA).

10.20. Program units for the Memory Input/Output management

The program units for the Memory Input/Output management are briefly described in Table 10.21.

Program unit	**Synthetic Description**
check_max_file_unit_ID.f90	Check on the machine/OS-dependent number of existing file units
deallocation_sequence.f90	Sequence of deallocations
diagnostic.f90	Diagnostic (error) messages
open_close_file.f90	File opening or closing

Table 10.21. Program units for the Memory Input/Output management (SPHERA, 2022, [1]).

10.21. Program units for the Allocations/deallocations of derived types

The program units for the Allocations/deallocations of derived types are briefly described in Table 10.22.

Program unit	**Synthetic Description**
allocate_de_Bod_elem_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type "body_element" and range (number of dimensions) 1
allocate_de_BodPar_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type body_particle and range (number of dimensions) 1
allocate_de_Bod_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type body and range (number of dimensions) 1
allocate_de_BouConEdg_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyBoundaryConvexEdge and range (number of dimensions) 1
allocate_de_BouDat_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyBoundaryData and range (number of dimensions) 1
allocate_de_BouFac_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyBoundaryFace and range (number of dimensions) 1
allocate_de_BouSid_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyBoundarySide and range (number of dimensions) 1
allocate_de_BouStr_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyBoundaryStretch and range (number of dimensions) 1
allocate_de_CLCp_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type CLC_polygon_der_type and range (number of dimensions) 1
allocate_de_CtlLin_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyCtlLine and range (number of dimensions) 1
allocate_de_CtlPoi_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyCtlPoint and range (number of dimensions) 1
allocate_de_face_r1.f90	Allocation/Deallocation of a generic allocatable array of type "face_derived_type" and range (number of dimensions) 1
allocate_de_Med_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyZone and range (number of dimensions) 1
allocate_de_Par_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyParticle and range (number of dimensions) 1
allocate_de_QSec_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type tyQ_section_array and range (number of dimensions) 1
allocate_de_Sub_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type type_substation and range (number of dimensions) 1
allocate_de_TimSta_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type Tytime_stage and range (number of dimensions) 1
allocate_de_vertex_r1.f90	Allocation/Deallocation of a generic allocatable array of type "vertex_derived_type" and range (number of dimensions) 1
allocate_de_vtu_grid_r1.f90	Allocation/Deallocation of a generic allocatable array of type "vtu_grid_derived_type" and range (number of dimensions) 1

allocate_de_Zon_r1.f90	Allocation/Deallocation of a generic allocatable array of derived type TyZone and range (number of dimensions) 1

Table 10.22. Program units for the Allocations/deallocations of derived types (SPHERA, 2022, [1]).

10.22. Program units for the Allocations/deallocations of preset types

The program units for the Allocations/deallocations of preset types are briefly described in Table 10.23.

Program unit	**Synthetic Description**
allocate_de_ch100_r1.f90	Allocation/deallocation of a generic allocatable array of type character(100) and range (number of dimensions) 1.
allocate_de_dp_r1.f90	Allocation/Deallocation of a generic allocatable array of type "double precision" and range (number of dimensions) 1
allocate_de_dp_r2.f90	Allocation/Deallocation of a generic allocatable array of type "double precision" and range (number of dimensions) 2
allocate_de_dp_r3.f90	Allocation/Deallocation of a generic allocatable array of type "double precision" and range (number of dimensions) 3
allocate_de_dp_r4.f90	Allocation/Deallocation of a generic allocatable array of type "double precision" and range (number of dimensions) 4
allocate_de_int4_r1.f90	Allocation/Deallocation of a generic allocatable array of type integer(4) and range (number of dimensions) 1
allocate_de_int4_r2.f90	Allocation/Deallocation of a generic allocatable array of type integer(4) and range (number of dimensions) 2
allocate_de_log_r1.f90	Allocation/Deallocation of a generic allocatable array of type logical and range (number of dimensions) 1
allocate_de_log_r2.f90	Allocation/Deallocation of a generic allocatable array of type logical and range (number of dimensions) 2

Table 10.23. Program units for the Allocations/deallocations of preset types (SPHERA, 2022, [1]).

10.23. Program units for Fortran modules

The program units for Fortran modules are briefly described in Table 10.24.

Program unit	**Synthetic Description**
Dynamic_allocation_module.f90	Module to define dynamically allocated variables
Hybrid_allocation_module.f90	Module to define derived types of both dynamically and statically allocated variables
I_O_file_module.f90	Module for I/O units: 11-max_file_unit_booked
Memory_I_O_interface_module.f90	Interfaces to the program units of the folder Memory_IO
Neighbouring_Search_interface_module.f90	Interfaces to the program units of the folder Neighbouring_Search
SA_SPH_module.f90	Module for the semi-analytic approach (boundary treatment scheme) of Di Monaco et al. (2011, EACFM)
Static_allocation_module.f90	Module to define global (and statically allocated) variables
Time_module.f90	Module for time recording

Table 10.24. Program units for Fortran modules (SPHERA, 2022, [1]).

10.24. Program units for the SPH neighbouring search

The program units for the SPH neighbouring search are briefly described in Table 10.25.

Program unit	**Synthetic Description**
CalcVarLength.f90	Neighbouring search (pre-conditioned dynamic vector), relative positions, kernel functions/derivatives, Shepard's coefficient, position of the fluid-sediment interfaces along each background grid column. Fluid-fluid and fluid-body contributions to the renormalization

	matrices for the pressure-gradient term and the velocity-divergence term.
cell_indices2pos.f90	To return the position of the barycentre of the background cell whose indices are provided as input. So far, it is used only for CLC.
CellIndices.f90	To return the indices (i,j,k) of the cell (nc) in a 3D domain with ni*nj*nk cells and the cell index in the 2D horizontal grid.
CellNumber.f90	To return the ID of the cell of indices (i,j,k). If particle is outside of the grid, it returns 0
CreaGrid.f90	To create the background positioning grid
grad_W_sub.f90	Gradient of the kernel function. kernel_ID: =1 cubic beta-spline cubic kernel as defined in the paper of Monaghan & Lattanzio (1985) =2 anti-cluster cubic kernel as defined in the paper of Gallati & Braschi (2000, "L'Acqua")
InterFix.f90	Minor program unit
liquid_particle_ID_array.f90	To count the liquid particles and update the associated ID array
OrdGrid1.f90	Ordering the numerical elements on the background positioning grid.
ParticleCellNumber.f90	To return the ID of the grid cell where particle np is located
w.f90	Kernel function. Cubic beta-spline function as defined in the paper of Monaghan & Lattanzio (1985)

Table 10.25. Program units for the SPH neighbouring search (SPHERA, 2022, [1]).

10.25. Program units for Post-processing

The program units for Post-processing are briefly described in Table 10.26.
The main output files report the following parameters:

- flow rate hydrographs at the flow rate monitoring sections;
- 2D fields of the maximum values of the specific flow rate and the free surface height;
- time evolution of the interfaces of the bed-load transport model;
- time evolution of the main fluid dynamics variables (pressure and velocity) along the monitoring lines and points;
- hydrographs of the free surface height along the monitoring points;
- application log of SPHERA;
- 3D fields of the main fluid dynamics and SPH variables (".vtu" and ".pvd" file formats) for Paraview (graphic FOSS) visualization;
- frontier geometry for the boundary treatment SA-SPH (".vtk" format for Paraview);
- output files of the boundary treatment scheme DB-SPH (ref.: folder "DB_SPH");
- output files on the transport of solid bodies in free surface flows (ref.: folder "Body_dynamics").

SPHERA returns the time series of the maximum height (file ".plb") and the minimum height (file ".zlft") of the free surface. The latter represents the position of the lowest water-air interface, along a generic vertical monitoring line. This quantity seems proper to assess the water depth, obtained after difference with the height of the underlying solid surface. The algorithm searches in the positioning grid the lowest height of the free surface (from the bottom towards the top of a monitoring line) and stops at the first positioning cell which is empty and has at least one cell filled of water below (along the vertical). The filtering procedure is based on pinpointing air masses between portions of liquid along a vertical monitoring line. This procedure might be improved integrating an additional control on the possible detachment between the lowest portion of the local liquid sub-domain and the underlying solid surface.

Program unit	**Synthetic Description**
cat_post_proc.f90	To concatenate the ".txt" output files and remove the original ones
electrical_substations.f90	output assessment and writing for electrical substations (only in 3D). Output (time series): Probability of an Outage Start

	(POS), Expected Outage Status (EOS), Expected Outage Time (EOT, time series update of its expected scalar value), Damage to the Substation Beyond Outage (Dsub, time series update of its expected scalar value), Substation Vulnerability (Vul). Output depends on Ysub (spatial average of the fluid/mixture depth at the DEM grid points, within the substation polygon). Ysub alternatively depends on two estimations of the water depth: the first (i_fil==1) is a raw estimation; the latter (i_fil==2) filters the atomization and the wave breaking effects.
final_post_processing.f90	Closure of the log file
fluid_global_quantities.f90	Record for the output time series of the fluid global quantities: mass, linear momentum, angular momentum (with respect to the domain origin), volume.
Memo_Results.f90	To write detailed results for restart
Print_Results.f90	Post-processing for the log file
result_converter.f90	Post-processing for .vtu (fluid dynamics parameters) for Paraview. Subroutine call for the concatenation of the .txt temporary output files
time_step_post_processing.f90	Post processing each time step

Table 10.26. Program units for Post-processing (SPHERA, 2022, [1]).

10.26. Program units for Post-processing the 2D synthetic fields

The program units for Post-processing the 2D synthetic fields are briefly described in Table 10.27.

Program unit	**Synthetic Description**
Update_Zmax_at_grid_vert_columns.f90	Updating the 2D arrays of the maximum values of the fluid particle height for each grid column (only in 3D). Printing the current 2D fields of the free-surface height (filtered and unfiltered), water depth (filtered and unfiltered), specific flow rate components (filtered) and height-averaged velocity (filtered), according to the output frequency chosen in the input file (only in 3D). The filter zeroes the effects associated with the presence of multiple fluid depths along the same vertical (atomization, breaking) and the liquid detachment from the topographic surface.
write_h_max.f90	To compute and write the 2D fields of the maximum values of the water depth, at the nodes of the Cartesian topography, provided as input data (only in 3D; two variants -hu, hf- based either on the unfiltered -Zu- or filtered -Zf- free surface height). Same task for the 2D field of the maximum (over time) specific flow rates.

Table 10.27. Program units for Post-processing the 2D synthetic fields (SPHERA, 2022, [1]).

10.27. Program units for Post-processing the elapsed time

The program units for Post-processing the elapsed time are briefly described in Table 10.28.

Program unit	**Synthetic Description**
s_ctime.f90	Minor program unit
start_and_stop.f90	Time recording
time_elapsed_IC.f90	To assess the elapsed time at the end of the initial conditions

Table 10.28. Program units for Post-processing the elapsed time (SPHERA, 2022, [1]).

10.28. Program units for Post-processing at SPH monitors

The program units for Post-processing at SPH monitors are briefly described in Table 10.29.

Program unit	Synthetic Description
calc_pelo.f90	Post-processing to write the time evolution of the free surface height at the monitoring lines. Filtering atomization of the fluid domain. The "lower fluid top height" ("z_lower_fluid_top") is suitable to estimate the liquid depth as a filter of the "maximum liquid height" ("pelolib") to cut off atomization and wave breaking effects. Filter criterion: during the upward search (starting from the bottom) for SPH particles along the monitoring line, stop at the first empty background cell, if at least one cell below is filled with the searched fluid.
CalcVarp.f90	To calculate physical quantities at a monitoring point
interface_post_processing.f90	Post-processing the interfaces for bed-load transport phenomena
Memo_Ctl.f90	Post-processing for monitoring lines and points
SPH_approximations_at_monitors.f90	SPH approximations of density, pressure and velocity vector at a monitoring element. Density is not obtained via EOS to treat also dense granular flows which are multi-phase.
sub_Q_sections.f90	Writing flow rate at monitoring sections provided in input for the flow rate (only in 3D).
write_Granular_flows_interfaces.f90	To print the interfaces for bed-load transport phenomena

Table 10.29. Program units for Post-processing at SPH monitors (SPHERA, 2022, [1]).

10.29. Program units for Pre-processing

The program units for Pre-processing are briefly described in Table 10.30.

Program unit	Synthetic Description
Gest_Input.f90	Input check and management
Init_Arrays.f90	Array initializations
ply_headings.f90	Reading the headings of a generic ".ply" file to obtain the number of vertices and faces
ReadCheck.f90	Check on a generic reading
ReadRestartFile.f90	To read the restart file
vtu_file_reading.f90	Decoding of a ".vtu" file. Only tetrahedral cells are admitted (".vtu" cell type: 10).
vtu_variable_reading.f90	Decoding of a ".vtu" variable within a ".vtu" file.

Table 10.30. Program units for Pre-processing (SPHERA, 2022, [1]).

10.30. Program units for Reading the input files

The program units for Reading the input files are briefly described in Table 10.31.

Program unit	Synthetic Description
ReadBedLoadTransport.f90	Reading input data for bed-load transport
ReadBodyDynamics.f90	Reading input data for body transport in fluid flows (Amicarelli et al., 2015, CAF).
ReadDBSPH.f90	Reading input data for the DB-SPH boundary treatment scheme (Amicarelli et al., 2013, IJNME).
ReadInputBoundaries.f90	Reading input data
ReadInputControlLines.f90	Reading input data for the boundary treatment scheme SA-SPH (Semi-Analytical approach; Di Monaco et al., 2011, EACFM).
ReadInputControlPoints.f90	Reading monitoring lines
ReadInputDomain.f90	Reading monitoring points
ReadInputDrawOptions.f90	Reading input data to define the numerical domain
ReadInputExternalFile.f90	Reading the output time step for the 3D fields

ReadInput.f90	Reading external files on vertices and faces
ReadInputFaces.f90	Reading input faces
ReadInputGeneralPhysical.f90	Reading gravity and background pressure
ReadInputLines.f90	Reading input lines
ReadInputMedium.f90	to read the input data from the section "medium" of SPHERA main input file.
ReadInputOutputRegulation.f90	Reading the output time steps
ReadInputParticlesData.f90	Reading the particle data from the section "BOUNDARIES" of the input files
ReadInputRestart.f90	To read the restart parameters from the main input file
ReadInputRunParameters.f90	Reading time-integration and other numerical quantities
ReadInputTitle.f90	Reading the test-case title
ReadInputVertices.f90	Reading the input vertices
ReadSectionFlowRate.f90	Input management for the flow rate monitoring sections
ReadSubstations.f90	Input management for monitoring the electrical substations

Table 10.31. Program units for Reading the input files (SPHERA, 2022, [1]).

10.31. Program units for the SASPH contributions to the Mass-related balance equations

The program units for the SASPH contributions to the Mass-related balance equations are briefly described in Table 10.32.

Program unit	**Synthetic Description**
AddBoundaryContribution_to_CE2D.f90	Computation of the SASPH terms for the 2D continuity equation of a generic "npi" SPH particle (Di Monaco et al., 2011, EACFM). SASPH contributions to the renormalization matrix for the 2D velocity-divergence term (only for the first step). ALE3-LC: 2D auxiliary vectors for the explicit ALE1 SASPH term of CE; ALE implicit terms in CE.
AddBoundaryContribution_to_CE3D.f90	Computation of the SASPH boundary terms for the 3D continuity equation (Di Monaco et al., 2011, EACFM). SASPH contributions to the renormalization matrix for the 3D velocity-divergence term (only for the first step). ALE3-LC: 3D auxiliary vectors for the explicit ALE1 SASPH term of CE; ALE implicit terms in CE.
SASPH_continuity.f90	SASPH boundary term for the continuity equation. Inversion of the renormalization matrices for the 3D and 2D velocity-divergence term and pressure-gradient term (also in the absence of SASPH neighbouring frontiers). Renormalization of the SASPH velocity-divergence term of the continuity equation. ALE3-LC: explicit ALE1 and ALE2 SASPH terms of CE (both C0 and C1 consistency).

Table 10.32. Program units for the SASPH contributions to the Mass-related balance equations (SPHERA, 2022, [1]).

10.32. Program units for the SASPH contributions to the Momentum Equation

The program units for the SASPH contributions to the Momentum Equation are briefly described in Table 10.33.

Program unit	**Synthetic Description**
AddBoundaryContributions_to_ME2D.f90	To compute boundary terms for the 2D momentum equation (gradPsuro,ViscoF). Equations refer to particle npi. In case of a neighbouring inlet section, the particle velocity is assigned (Di Monaco et al., 2011, EACFM).

	SASPH contributions to the renormalization matrix for the 2D pressure-gradient term (only for the first time step). Inversion of the renormalization matrix in 2D, even in the absence of SASPH neighbours (only for the first time step). 2D SASPH contributions to the ALE1 velocity increment.
AddBoundaryContributions_to_ME3D.f90	3D SASPH boundary terms for the momentum equation of any computational particle (Di Monaco et al., 2011, EACFM). SASPH contributions to the renormalization matrix for the 3D pressure-gradient term (only for the first time step). Inversion of the renormalization matrix in 3D, even in the absence of SASPH neighbours (only for the first time step). 3D SASPH contributions to the ALE1 velocity increment and its associated term in the momentum equation.
AddElasticBoundaryReaction_2D.f90	To add supplementary normal boundary reaction to support possible insufficient pressure gradient boundary term (2D). (Di Monaco et al., 2011, EACFM).
AddElasticBoundaryReaction_3D.f90	To add supplementary normal boundary reaction to support possible insufficient pressure gradient boundary term (3D). (Di Monaco et al., 2011, EACFM).

Table 10.33. Program units for the SASPH contributions to the Momentum Equation (SPHERA, 2022, [1]).

10.33. Program units for the Initial Conditions for the fixed surface walls

The program units for the Initial Conditions for the fixed surface walls are briefly described in Table 10.34.

Program unit	**Synthetic Description**
BoundaryMassForceMatrix2D.f90	Generation of the generalized boundary mass force matrix in 2D (Di Monaco et al., 2011, EACFM)
BoundaryMassForceMatrix3D.f90	Generation of the generalized boundary mass force matrix in 3D (Di Monaco et al., 2011, EACFM)
BoundaryReflectionMatrix2D.f90	Generation of the generalized reflection matrix in 2D (Di Monaco et al., 2011, EACFM)
BoundaryVolumeIntegrals2D.f90	To compute the boundary volume integrals IntWdV (Di Monaco et al., 2011, EACFM)
CompleteBoundaries3D.f90	Complete SASPH 3D boundaries (Di Monaco et al., 2011, EACFM)
DefineBoundaryFaceGeometry3D.f90	To define boundary faces from 3D geometry (Di Monaco et al., 2011, EACFM)
DefineBoundarySideGeometry2D.f90	Definition of the boundary sides. (Di Monaco et al., 2011, EACFM)
DefineBoundarySideRelativeAngles2D.f90	Detection of the previous adjacent side and associated relative angle (for each boundary side). (Di Monaco et al., 2011, EACFM)
GridCellBoundaryFacesIntersections3D.f90	To find the boundary faces intercepted by each frame cell of the grid (Di Monaco et al., 2011, EACFM)
main_wall_info.f90	To count the number of vertices of the main wall (i.e., the solid ("fixe") boundary with the minimum zone ID).
ModifyFaces.f90	To generate triangles from quadrilaterals (partitioning along the shortest diagonal)

Table 10.34. Program units for the Initial Conditions for the fixed surface walls (SPHERA, 2022, [1]).

10.34. Program units for the assessment of the SASPH integrals at the initial time

The program units for the assessment of the SASPH integrals at the initial time are briefly described in Table 10.35.

Program unit	**Synthetic Description**
ComputeBoundaryIntegralTab.f90	Partitioning the truncated portion of the kernel support for the SASPH scheme (Di Monaco et al., 2011, EACFM)
ComputeKernelTable.f90	Kernel functions and gradients for SASPH scheme (Di Monaco et al., 2011, EACFM)

IWro2dro.f90	To compute the definite integral IWro2dro (Di Monaco et al., 2011, EACFM)
J2Wro2.f90	To compute the integral J2Wro2 (Di Monaco et al., 2011, EACFM)
JdWsRn.f90	SA-SPH intermediate integrals (Di Monaco et al., 2011, EACFM)
WIntegr.f90	Computing the definite integral WIntegr (Di Monaco et al., 2011, EACFM). The constant "KERNELCONST2D" refers to the beta-spline cubic kernel, which always applies to $J_{3,w}$ also for the 2D pressure-gradient term.

Table 10.35. Program units for the assessment of the SASPH integrals at the initial time (SPHERA, 2022, [1]).

10.35. Program units for the interpolations of SASPH integrals during the time steps

The program units for the interpolations of SASPH integrals during the time steps are briefly described in Table 10.36.

Program unit	**Synthetic Description**
ComputeBoundaryDataTab_2D.f90	To calculate the array to store close boundaries and integrals in 2D (Di Monaco et al., 2011, EACFM)
ComputeBoundaryDataTab_3D.f90	To calculate the array to store close boundaries and integrals in 3D (Di Monaco et al., 2011, EACFM)
ComputeBoundaryVolumeIntegrals_P0.f90	Computation of boundary volume integrals in 3D (Di Monaco et al., 2011, EACFM)
ComputeSurfaceIntegral_WdS2D.f90	Computing the surface integral of kernel W along the segments intercepted by the kernel support (Di Monaco et al., 2011, EACFM)
ComputeVolumeIntegral_WdV2D.f90	Computing the integral of WdV extented to the volume delimited by the kernel support (Di Monaco et al., 2011, EACFM)
InterpolateBoundaryIntegrals2D.f90	Interpolation of 2D SASPH integrals (Di Monaco et al., 2011, EACFM)
InterpolateTable.f90	Interpolation of 3D SASPH integrals (Di Monaco et al., 2011, EACFM)

Table 10.36. Program units for the interpolations of SASPH integrals during the time steps (SPHERA, 2022, [1]).

10.36. Program units for the SPH neighbouring search for fixed surface walls

The program units for the SPH neighbouring search for fixed surface walls are briefly described in Table 10.37.

Program unit	**Synthetic Description**
DefineLocalSystemVersors.f90	To define the directional cosines of the local reference system (Di Monaco et al., 2011, EACFM). Further modifications take into account pentagon and hexagon faces (only for complex "perimeter" zones / fluid reservoirs, not for SASPH frontiers).
FindBoundaryConvexEdges3D.f90	To look for possible edges with an associated convex geometry. Their geometrical data are saved in BoundaryConvexEdge as TyBoundaryConvexEdge. (Di Monaco et al., 2011, EACFM)
FindBoundaryIntersection2D.f90	To find the intersection segment between the kernel support and the 2D boundary (Di Monaco et al., 2011, EACFM)
FindCloseBoundaryFaces3D.f90	To find the neighbouring SASPH boundary faces (Di Monaco et al., 2011, EACFM)
FindCloseBoundarySides2D.f90	To find the neighbouring SASPH boundary sides (Di Monaco et al., 2011, EACFM)
SelectCloseBoundarySides2D.f90	Completion of the neighbouring search for SASPH boundary sides (Di Monaco et al., 2011, EACFM)

Table 10.37. Program units for the SPH neighbouring search for fixed surface walls (SPHERA, 2022, [1]).

10.37. Program units for the contribution to the stability criteria from fixed surface walls

The program units for the contribution to the stability criteria from fixed surface walls are briefly described in Table 10.38.

Program unit	Synthetic Description
EvaluateBER_TimeStep.f90	Stability criterion for the SASPH elastic reaction (Di Monaco et al., 2011, EACFM)

Table 10.38. Program units for the contribution to the stability criteria from fixed surface walls (SPHERA, 2022, [1]).

10.38. Program units for the wall function for fixed surface walls

The program units for the wall function for fixed surface walls are briefly described in Table 10.39.

Program unit	Synthetic Description
wall_function_for_SASPH.f90	Assessment of the mixing-length turbulent viscosity and the slip coefficient only for the SASPH boundary terms of the momentum equation, depending on the wall function of the Surface Neutral Boundary Layer (SNBL) for rough walls. The slip coefficient also affects the partial smoothing of the velocity field.

Table 10.39. Program units for the wall function for fixed surface walls (SPHERA, 2022, [1]).

10.39. Program units for the string management

The program units for the string management are briefly described in Table 10.40.

Program unit	Synthetic Description
GetToken.f90	To extract the "itok"-th token from the input string "ainp"
lcase.f90	To convert upper case in lower case letters for a string whose blank spaces are all located at the end of the string
ReadRiga.f90	Line reading

Table 10.40. Program units for the string management (SPHERA, 2022, [1]).

10.40. Program units for Time integration

The program units for Time integration are briefly described in Table 10.41.

Program unit	Synthetic Description
Euler.f90	Explicit RK1 time integration scheme (Euler scheme)
Heun.f90	Heun scheme: explicit RK2 time integration scheme
Leapfrog_continuity.f90	Leapfrog time integration scheme for mass-related balance equations.
Leapfrog_momentum.f90	Leapfrog time integration scheme (momentum equation). Detection of the frozen-mass particles (ALE3).
Leapfrog_trajectories.f90	Leapfrog time integration scheme (fluid particle trajectories)
stoptime.f90	Assessment of the final simulation time for each medium depending on the motion type. The input motion type "law" is also converted in "fix".
time_integration.f90	Explicit Runge-Kutta time integration schemes
time_step_duration.f90	Computation of the time step duration (dt) according to stability constraints (CFL condition, viscosity term stability criterion). Plus, a special treatment for Monaghan artificial viscosity term and management of low-velocity SPH mixture particles for bed-load transport phenomena.

Table 10.41. Program units for Time integration (SPHERA, 2022, [1]).

10.41. Style formatting

The following non-mandatory rules have been followed in formatting SPHERA program units:

1) Please use the subroutine labels at the beginning of each program unit (title and description) and of sub-section (“modules”, “declarations”, “explicit interfaces”, “allocations”, “initializations”, “statements”, “deallocations”).
2) Please use Fortran 95 standard and portable procedures to be compiled with both gfortran and ifort.
3) A generic program unit has to be named as the associated file (without file extension) to have simpler dependencies in the makefile. As a consequence, one file per program unit is allowed and vice versa.
4) Please write since the first column of each line.
5) Please use 3 blank spaces for indentation.
6) Please use 1 blank space only before and after any mathematical operator in the Right Hand Side of each assignment and when a blank space is clearly convenient in terms of readability. Otherwise, blank spaces are used only for indentation (and within comments). For example, “endif” and “enddo” better replace “end if” and “end do”. Further, no blank space is present between a procedure and its arguments (e.g. write(*,*)).
7) For readability and printability, do not write beyond column 80. Here the symbol “&” is put for a new line.
8) Please follow this variable order for declarations: parameters, “inout” variables, local variables, external functions. For each of the previous variable set, please following the following sub-order: scalars, 1D arrays, …, nD arrays. Provided the same dimensionality, variable declarations follow this “sub-sub-order”: “logical”, “integer”, “double precision”, “character”, derived types.
9) A comment begins with “! <capitol letter>“ (there is a blank space after “!”).
10) Any logical expression is written within brackets (e.g., “(a==b).and.(c==d)”).
11) Automatic indentation is allowed only with blank spaces instead of tabs (but the makefile).
12) No multiple statements on a line (do not use “;” as a statement separator).
13) Do not go to a new line with “&” under the section “declarations”.
14) Keywords are written in lower case letters (e.g.: do,if,…).
15) Comments are meant to be written in English.

11. THE NUMERICAL CHAIN OF SPHERA

An overview of the numerical modelling chain of SPHERA is reported in Figure 11.1.
The RSE tools of the present numerical chain (i.e. SPHERA, DEM2xyz, ply2SPHERA_perimeter, Grid Interpolator) are developed by means of a series of numerical tools: gedit (GNOME Foundation, 2021, [76], text editor), gfortran (GNU, Free Software Foundation, 2021, [77]; Fortran compiler), gmake (GNU, Free Software Foundation, 2021, [78]; for the Makefile execution), gdb (GNU, Free Software Foundation, 2021, [79]; debugger), gprof (GNU, Free Software Foundation, 2021, [80]; profiler for scalar executions), Valgrind (Valgrind Developers, 2021, [81]; memory management tool), Scalasca (Forschungszentrum Jülich & TU Darmstadt, 2021, [82]; code profiler for OMP executions). The RSE tools are developed and distributed with no charge by means of Git (Torvalds et al., 2021, [68]; main “Distributed Version Control System” -DVCS-) and GitHub (GitHub Inc., 2021, [83]; main platform for Git-managed software).
The numerical modelling chain is based on free tools: FOSS, freeware or OpenData.
FOSS tools are defined as “Free/Libre and Open-Source Software” by the Free Software Foundation (2021, [84]). The FOSS tools of the modelling chain are: gedit, gfortran, gmake, gdb, gprof, Valgrind, Scalasca, Git, GDAL, Paraview, DEM2xyz, SnappyHexMesh, ply2SPHERA_perimeter, SPHERA, Grid Interpolator, Image Magick, Gnuplot, Engauge Digitizer, Virtual Dub.
“Freeware” tools are simply free software. Two freeware tools are used in the reference modelling chain: GitHub (for public repositories) and GSView.
“Open-Data” tools are databases available upon public and free access like SRTM3, which belongs to the modelling chain. The FOSS tools gfortran, gprof and Scalasca can be possibly replaced with more effective codes such as: ifort (Intel Corporation, 2021, [85]; Fortran compiler); idb (Intel

Corporation, 2021, [86]; Fortran debugger); cpuinfo (Intel Corporation, 2021, [87]; code profiler for OMP executions); ITAC, Trace Analyzer and VTUNE (Intel Corporation, 2021, [88]; code profiler for OMP/MPI executions); Advisor (Intel Corporation, 2021, [89]; profiler for executions with code vectorization); TotalView (Rogue Wave Software, 2018, [90]; code debugger for parallel executions). Analogously, it is possible to support or replace the other elements of the modelling chain with more effective software, if available (e.g., a DEM with finer spatial resolution).
The possible replacement of a free tool with a proprietary tool (available with charge) should be a reversible procedure which does not alter the functioning of the modelling chain.
The following sub-sections report some examples of use procedures for the software tools of the numerical chain of SPHERA. For further details one considers the guide of a specific tool.

11.1. Engauge Digitizer

Some input data need a digitalization procedure. Further, the profiles which will be simulated will need to be compared with the analogous experimental (or numerical) profiles available from experimental images or scientific literature (indexed journals; Open-Data archives).
In all the above cases, it is normally admitted the digitization of experimental and numerical profiles from published sources by means of Engauge Digitizer (Mitchell et al., 2021, [91]), with proper citation of the source.

11.2. Site overview

During the site setting one could be interested in the following procedure:

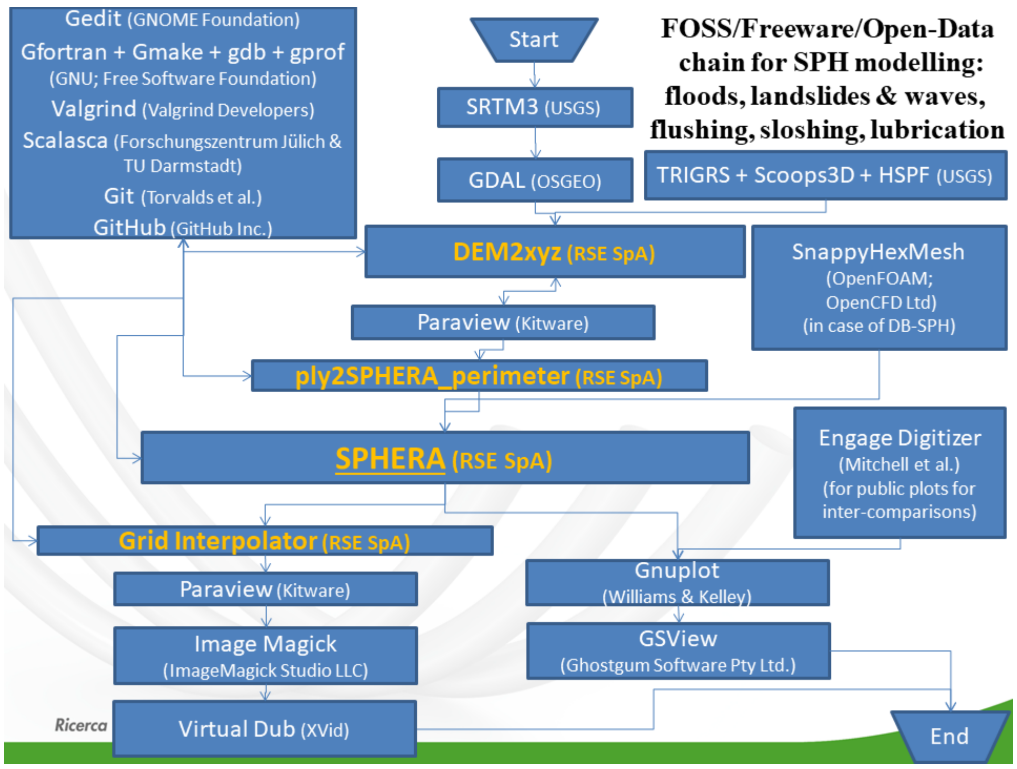
1. Define the site boundaries.
2. Provide the reference geographic and cartographic coordinates of a representative point within the domain.
3. Preliminary assessment of the elapsed time t_e (s) as function of the fluid and topography spatial resolution (dx and $dx_{DEM,DTM}$). The elapsed time roughly depends on the average number of SPH particles N_{avg}, the number of boundary elements $N_{BC,elements}$ and the number of time steps N_{step} ($t_e \propto N_{avg} \cdot N_{BC,elements} \cdot N_{step}$). Define the main variables influencing N_{avg}, $N_{BC,elements}$ and N_{step}. They are test-case dependent. Compare them to an analogous test case with known t_e.

11.3. The SRTM3 dataset and further DEM-DTM datasets

Published at the end of 2014, the dataset SRTM3 (USGS, 2014, [92]) provides a repository of DEM (“Digital Elevation Models”) with an almost global cover and spatial resolution of 1” (in terms of geographical coordinates) or ca.31m (maximum/coarser spatial resolution in terms of cartographic coordinates). With respect to the former free DEM datasets (finest spatial resolution of 3”), SRTM3 has permitted a relevant improvement in using Open-Access DEM. The SRTM3 products are available in the format “.tif”. Considering its global coverage, SRTM3 provides a root-mean-square error on heights of ca.6m, even though the error kurtosis is very high (Rexer & Hirt, 2014, [93]). However, these estimations refer to almost the whole terrestrial surface, included the high-latitude regions where errors are larger. Any additional DEM and DTM files might replace/integrate the SRTM3 DEM files for the current test case. For Italian sites, the web sites of the regions and the Ministry MiTE might represent the main references.
The effective spatial resolution of the DEM/DTM is $\frac{1}{\sqrt{2}}$ times its formal spatial resolution as Delaunay’s triangulation applies each rectangular boundary element is halved.

11.4. GDAL

The software tool GDAL (OSGEO, 2021, [53]), the main QGIS library, can be used as an independent code. In the frame of the present modelling chain, GDAL allows to convert the DEM file format “.tif” in the alternative format “.asc”.


Gedit (GNOME Foundation)
Gfortran + Gmake + gdb + gprof (GNU; Free Software Foundation)
Valgrind (Valgrind Developers)
Scalasca (Forschungszentrum Jülich & TU Darmstadt)
Git (Torvalds et al.)
GitHub (GitHub Inc.)
Start
FOSS/Freeware/Open-Data chain for SPH modelling: floods, landslides & waves, flushing, sloshing, lubrication
SRTM3 (USGS)
GDAL (OSGEO)
TRIGRS + Scoops3D + HSPF (USGS)
DEM2xyz (RSE SpA)
SnappyHexMesh (OpenFOAM; OpenCFD Ltd) (in case of DB-SPH)
Paraview (Kitware)
ply2SPHERA_perimeter (RSE SpA)
SPHERA (RSE SpA)
Engage Digitizer (Mitchell et al.) (for public plots for inter-comparisons)
Grid Interpolator (RSE SpA)
Gnuplot (Williams & Kelley)
Paraview (Kitware)
GSView (Ghostgum Software Pty Ltd.)
Image Magick (ImageMagick Studio LLC)
Ricerca
Virtual Dub (XVid)
End
Figure 11.1. FOSS / Freeware / Open-Data tools of the modelling chain for SPHERA.

11.5. DEM2xyz: preliminary elaboration of the raw DEM and DTM files

"DEM2xyz v.3.0" (RSE SpA) reads a "DEM" file and writes the associated DEM in a corresponding "xyz" file, possibly changing the spatial resolution (as requested by the user). In case the absolute value of the mean latitude is provided with a non-negative value, the following conversion takes place "(*lon*, *lat*) in (°) to (*X*, *Y*) in (m)". In this case, an interpolation (weighted on the inverse of the distance square) is carried out to provide a regular cartographic output grid in (*X*, *Y*). The height of the DEM points which belong to the digging/filling regions (provided in input) is modified. After this treatment, each digging/filling region has null slope. Bathymetry is possibly extruded from the heights of the most upstream and downstream coastline points. The bathymetry/reservoir extrusion is corrected in case the volume reservoir is provided as an input parameter. Multiple reservoirs are admitted. Digging regions cannot overlap each other unless they are set to the same height. Reservoir/bathymetry regions cannot overlap each other. In case a digging region overlaps a reservoir region, the latter holds the priority. In the presence of a volume correction, two reference shapes are available: "reservoir" and "volcanic lake". DEM2xyz v.3.0 is compatible with SPHERA v.10.0.0 (RSE SpA).
The current procedure has to be repeated for each dataset (multiple datasets).
The origin of the reference system of the DEM2xyz output is the lower-left vertex of the DEM2xyz input (relative reference system).
Use the same relative reference system for all the DTMs: at this stage the origin of the relative reference system is the lower-left vertex of the lower-left DEM-DTM of the current dataset.

11.6. Paraview: post-processing of the preliminary DEM and DTM files

The output files of DEM2xyz (".txt" or ".xyz") can be post-processed by means of Paraview (Kitware, 2021, [94]), by executing the following procedure (to be repeated for each DTM and DEM file):

1. open the .txt/.csv file; change the "Field Delimiter Character" to a blank space by typing one blank space; choose the option "Merge consecutive limiters";
2. apply the filter "TabletoPoints";
3. eventually cut the proper domain especially if the virtual memory is insufficient: halve the DTM step by step by halving the maximum among 0.5 · $n_{columns}$ (*x* values; "0.5" is due to the threshold anisotropy in Paraview when building "Delaunay2D" tessellations) and n_{rows} (*y* values). This

procedure will efficiently respect both the maximum number of elements, columns and rows for the filter "Delaunay2D" of Paraview.

4. apply the filter "Delaunay2D"; build a Delaunay tessellation at this stage that the DTMs are not merged to avoid crashes in case of big DEM-DTM files;

5. remove the spurious parts of the output of Delaunay2D;

6. "Save Data>.ply".

7. Save data as ASCII .ply and binary .ply; check both on PV.

8. Save data as .csv:

- **8.1.** use scientific notation; select more than 5 significant decimal digits only if necessary;
- **8.2.** save;
- **8.3.** check the .csv with a text editor and compare with the precisions to be defined;
- **8.4.** reorder the columns by means of Linux OS "dos2unix", "yes", "cut" and "paste" commands (their combination and use is test-case dependant): first column (x-values), second column (y-values), third column (null values), fourth column (z-values);
- **8.5.** check on Paraview.

11.7. Elaboration of a unique DEM-DTM integrated surface

Hereafter is reported a use procedure for integrating several DEM (Digital Elevation Model) files with the associated DTM (Digital Terrain Model) files in a unique wall boundary for SPH simulations, by means of the following software tools: Paraview (Kitware, 2020, [94]); DEM2xyz (2020, [95]); ply2SPHERA_perimeter (2020, [96]); Grid Interpolator (2020, [97]). Specific treatments concern the following items: power lines and electricity pylons, trees, sub-grid roughness, bridges, DTM and DEM intrinsic bugs.

Paraview also allows to cut the numerical domain (the cuts have to be far enough from the water bodies not to disable the procedures to extrude the water bodies from the DEM), circumscribes the water bodies, draws the possible filling/digging regions, detects the dam toe and the most upstream point over the coastline of the water bodies.

The current procedure is composed by the following steps and has to be repeated for each dataset (multiple datasets):

1. integrate the elaborated DTM and DEM 3D surfaces:

- **1.1.** assess the 2D field of the height difference (z_{diff} -m-) between the DEM and the DTM and save it with the ".csv" format ("Save Data"; in order to optimize the Virtual Memory in the following steps, select more than 5 significant decimal digits only if necessary);
- **1.2.** ad-hoc treatment of the intrinsic errors due to the definition of the DEM (i.e., a surface described by a 3D function, a single quote being associated with any couple of horizontal coordinates) by means of Paraview (local treatment of the field of z_{diff}, obstacle by obstacle, where necessary): select the horizontal coordinates of the "digging regions" to feed DEM2xyz (one can use the "Source Splines" for design removing the exceeding decimal digits, provided the precision requested by DEM2xyz):
  - **1.2.1.** fix the elevated permeable obstacles:
    - **1.2.1.1.** remove the power lines, the electricity pylons and the other truss structures as they are extruded to the ground as impermeable obstacles;
    - **1.2.1.2.** the tree foliage can be left, except for those trees too close each other whose extrusion to the ground locally form a fictitious "tree impermeable wall";
    - **1.2.1.3.** removal of bridges as they are extruded to the ground as impermeable obstacles, whereas their depth is limited;
  - **1.2.2.** remove the clearly fictitious obstacles (i.e., the DEM bugs) from the riverbeds;
    - **1.2.2.1.** convert the z_{diff} file from the ".csv" to the ".xyz" format for Grid Interpolator (one can use the commands "cut" and "paste" from Linux terminal);
    - **1.2.2.2.** convert the z_{diff} file from the ".xyz" to the ".asc" format by means of Grid Interpolator;

1.2.2.3. apply the "digging regions" by means of DEM2xyz to zero the DEM/DTM differences at those nodes where the DEM bugs are selected, and convert the resulting file (variable $z_{diff,digs}$) from the ".asc" to the ".xyz" format;

1.2.2.4. upload in Paraview the field of $z_{diff,digs}$, apply the filter h_f (m) and add the DTM heights to obtain the DEM field of $z_{DEM,digs,filter}$, filtered with respect to the sub-grid roughness and DEM bugs;

The Paraview use procedure for the DEM/DTM elaboration needs a filter for the local height difference between the DEM and the DTM. Hereafter the formulation of a generic filter in case one only uses the integer rounding function ("*rint*") as discontinuous function. This is the case when using Paraview without Python scripts (Paraview filter "Python Caculator"; no "*if*" construct is available, no other discontinuous function is avilable such as the Heaviside step).

The definition of the DEM height filtered ($z_{DEM,filt}$, -m-) depends on the DEM height (z_{DEM}, m), the DTM height (z_{DTM}, m), the filter threshold on the height difference ($z_{diff,thr}$ m) and the height precision z_{pr} (m):

$$z_{DEM,filt} = z_{DTM} + z_{diff} K, \quad z_{diff,thr} = z_{DEM} - z_{DTM}, \quad K = A_{filt} / \left(A_{filt} + z_{pr} \right) \cong \left\{ \begin{matrix} 1, & \mathrm{z_{diff}} \geq \mathrm{z_{diff,thr}} \\ 0, & \mathrm{z_{diff}} < \mathrm{z_{diff,thr}} \end{matrix} \right\}$$
$$A_{filt} = \mathrm{rint} \left\{ \frac{\left(\mathrm{z_{diff}} - \min\{\mathrm{z_{diff}}\} \right)}{\left[2 \cdot \left(\mathrm{z_{diff,thr}} - \min\{\mathrm{z_{diff}}\} \right) \right]} \right\} \tag{11.1}$$

The filter h_f for the DEM/DTM roughness is applied for the following reasons:

a) not to explicitly simulate possible impermeable obstacles on the riverbed reported on the DEM, but not on the DTM (within the waterbody, water flows on the DTM not on the DEM);
b) not to explicitly simulate the roughness elements smaller than the sub-grid scale, already considered by the roughness length z_0 (m) in the wall function for the slip coefficient;
c) to keep h_f compatible with the chosen value of z_0 (obtained for example from Davenport's scale; Plate, 1995, [98]), considering that d_{50}=10z_0 (NSBL under rough-wall regime) and that the equivalent diameter of the sub-grid roughness elements is d_{50}=2H, where H (m) is the average height of the same elements;
d) to reduce of the ad-hoc treatments associated with the DEM bugs;
e) to ease the imposition of a homogeneous field of z_0 not to distinguish the riverbed from the floodplain (in terms of sub-rid roughness); the last choice being feasible provided that the DEM-DTM surface is partitioned accordingly;
f) to keep the height precision compatible with the minimum scale of the DEM-DTM explicit roughness (they can be obtained by approximated assessments in Paraview, after comparing the DEM and the DTM).

1.3. analogously repeat the previous steps to fix the intrinsic DTM errors (instead of the DEM errors; consider $z_{DEM,digs,filter}$ instead of z_{diff} and do not apply any filter);

1.4. visualize the fixed obstacles removed during the previous steps by means of Paraview as passive targets which do not affect the SPH simulations;

1.5. if necessary, merge two (or more) DEM/DTM adjacent files by means of Paraview: execute the following procedure only if it can be easily carried out (e.g., 2 files with coincident overlapping nodes), otherwise skip it:

1.5.1. remove the overlapping region in one of the two files;

1.5.2. remove from one file the redundant nodes;

1.5.3. locally seam the boundary region between the two files (filter "Delaunay 2D");

1.5.4. merge the two resulting files and the boundary region (filter "Append Datasets").

1.6. Alternative merging procedure via Grid Interpolator: merging as removing overlapping parts with shared boundaries; the aim is a unique DEM-DTM surface well connected, not a unique DEM-DTM file ($n_{x,max}$=1'000 and $n_{y,max}$=1'000 each output file not to have issues with Paraview "Delaunay2D" filter; check identical shared boundaries and coherence with raw DEM-DTM files). The advantages of this merging are expressed as follows: no overlapping regions; clean merge (shared boundary points have the same height); no ad-hoc seals between Paraview files; it is not necessary to merge all the DTM files on Paraview (issues avoided with huge DTMs); the unique regular Cartesian grid (far from boundaries) avoids issues on SPHERA extrusions (Dirichlet's boundary conditions for the water depth; initial conditions for water bodies extruded from DEM-DTM). Interpolation introduces some errors, thus use the minimum influence radius.

1.7. Repeat the Paraview procedure 11.6.

11.8. DTM/DEM dataset merging

1. Merge all the datasets (each featured by $dx_{DTM,DEM}$) with possible roto-translation (in case of different cartographic systems). This is alternative to work since the beginning with the same cartographic system (to avoid possible complicated procedures with GDAL). The origin of the relative reference system at this final stage is the lower-left vertex of the lower-left DEM-DTM of the lower-left dataset;
2. Digging zones in DEM2xyz (for negative heights it gives NaN, e.g., z=-999.m) to avoid dataset overlapping (start from the finest);
3. Dataset sealing: extract the edge points of the finest dataset on Paraview and apply "Delaunay2D" with the other dataset; unique regular Cartesian grid (except for boundaries) with possible local refinement;
4. Possible interpolation to obtain a regular Cartesian grid, if requested (e.g., fluid extrusions, "BC_{zmax}" zones).

11.9. DEM2xyz: further modifications to the DEM-DTM improvements

DEM2xyz is executed again on each DEM-DTM file to:

1. possibly reconstruct the bathymetry below the water bodies;
2. possibly assign other digging/filling regions (also to fix errors);
3. cut the DEM-DTM files to reduce the computational time;
4. check the normal vectors to the surface elements of the DEM-DTM;
5. save the Paraview state;
6. generate the ".ply" files (both binary and ASCII) for ply2SPHERA_perimeter.
7. In case of multi-spatial-resolution DTM, if possible, try to build a unique ".ply" file from the DTM points elaborated so far to avoid minor discrepancies at those edges where dx_{DTM} changes.

11.10. Paraview: post-processing of the final output of DEM2xyz

At this point, Paraview is used again to draw those geometrical figures which are necessary to initialize some variables in the main input file of SPHERA, in order to detect water bodies, earth-filled dams and monitoring elements.

11.10.1. Simple use procedure to design a weir

Hereafter is reported a simple and approximated use procedure to elaborate the DEM/DTM surface with Paraview in order to design a weir:

1. select a DEM-DTM zone extending $2h$ downstream the section considered and whose width is just sufficient to contain the riverbed (feature "select elements with polygone");
2. vertically extrude the selection above up to the domain maximum height (filter "linear Extrusion");
3. execute "Delaunay 3D" triangulation to obtain a tetrahedric grid within the extrusion;
4. extract from the extrusion volume the weir front wall (multiple use of the feature "clip", even to remove the volume exceeding the free-surface height to be imposed, and the filters "Extract Surface" and "Delaunay 3D") to obtain walls at least $2h$ thick;

5. analogously to the previous step, extract the weir lateral walls and remove the volume exceeding a proper height (larger than the free-surface height to be imposed and sufficient to avoid any lateral overtopping);
6. merge the lateral and front walls (filter “Append Datasets”);
7. remove the faces useless in the SPH simulation (feature “select elements with polygone” with selection inversion; filter “Extract Selection”);
8. check and possibly flip the face normals (filter “Generate Surface Normals” with “Flip Normals” and “splitting”);
9. visually check the weir design and save it in “.ply” format for ply2SPHERA_perimeter.

11.11. Modification of the spatial resolution of the DEM-DTM

The following steps can be finally executed to possibly modify the spatial resolution of the DEM-DTM already obtained:

1. execute again the tool DEM2xyz with a different spatial resolution factor considering the DEM-DTM already elaborated as an input file;
2. treat with Paraview the DEM2xyz “.xyz” output file;
  2.1. erase the points with *z*=-999m;
  2.2. apply the filter “Delaunay2D”;
  2.3. erase manually (by means of the Graphic User Interface) the few faces linking points unavailable in the input DEM-DTM;
  2.4. translate the DEM-DTM according to the origin of the Cartographic System;
  2.5. check the DEM-DTM normals;
  2.6. save the DEM-DTM at the new spatial resolution as a “.ply” output file;
3. elaborate the “.ply” DEM-DTM file with ply2SPHERA_perimeter;
4. add the ply2SPHERA_perimeter output files among the input files of SPHERA (sections “faces” and “vertices”).

In case Delaunay cannot be applied to the whole DEM-DTM, then it is safer to come back to the previous point on DEM2xyz to partition the DEM output instead of cutting it in Paraview. One cannot merge the DEM-DTM files after the interpolation because one would need a unique DEM-DTM input file for DEM2xyz, otherwise the overlapping points would not be equal due to different interpolation results at boundaries.

11.12. SnappyHexMesh

In the presence of CAE-made solid bodies or DBSPH mobile surface walls, SnappyHexMesh (OpenFOAM, OpenCFD Ltd, 2021, [58]) is used as a surface and volume grid generator for the initial positioning grid of the SPH body particles or the DB-SPH elements.
SnappyHexMesh (2021, [58]) uses the “.stl” input geometry as the boundaries of the mesh to be produced. SPHERA elaborates the geometry generated with Paraview (2021, [54]), after its conversion by ply2SPHERA_perimeter (2021, [96]), for the fluid reservoir not violating the fluid boundaries. It follows that using the SnappyHexMesh mesh for the solid and the Paraview-designed geometry for the fluid reservoir, there is no fluid-solid mass penetration, no matter about the test case. At the moment, it does not seem necessary to extract any surface mesh SnappyHexMesh (as “.ply” file) for the “perimeter” of the fluid.

11.13. ply2SPHERA_perimeter

The numerical tool ply2SPHERA_perimeter (RSE SpA, 2022, [96]) converts the DEM “.ply” file in two distinct output files. They have the same format as the sections “VERTICES” and “FACES” of the main input file of SPHERA. It is the vertices and faces of the portion of the DEM within the numerical domain of SPHERA.
“ply2SPHERA_perimeter v.3.0” (RSE SpA) is a minor pre-processing tool of the SPH code SPHERA v.10.0.0 (2022, [1]). It deals with the format conversion from “.ply” to the format of the sections

"VERTICES" and "FACES" of SPHERA main input file to describe the perimeter of a 3D zone (for 3D simulations) or a 2D zone (for 2D simulations).
The ".ply" input file is located in the working directory. Here, there is also the main input file of ply2SPHERA_perimeter v.3.0 (2022, [96]): ply2SPHERA_perimeter.inp. This is composed by 3 lines, with the following structure:
<integer_1> <integer_2> <integer_3> ! perimeter_first_vertex_ID (ID of the first vertex of the perimeter); perimeter_first_face_ID (ID of the first face of the perimeter); perimeter_ID (ID of the perimeter)
<real> ! z_offset (offset of the z-coordinate)
<character> ! ply_file_name (name of the .ply input file)
The tool ply2SPHERA_perimeter (2021, [96]) treats ".ply" files with attribute. Alternatively, some program units of SPHERA (i.e., "Import_ply_meshes", "CLC_preprocessing") manage ".ply" input files without attributes.

11.14. Paraview: procedures to pre-process the CORINE Land Cover (CLC) maps

Paraview v.5.9.0 (or later version) is mandatory to manage the input CLC ".shp" file. The following procedure might be applied:

1. Read the ".shp" file in the original Cartographic reference system;
2. Apply the filter "Transform" to switch to SPHERA reference system (*x*/*y* offset reduction);
3. Apply the filter "Clips" to cut SPHERA domain;
4. Save a screenshot plotting the first CLC category (vector map);
5. Save a screenshot plotting the CLC polygon IDs (vector map);
6. It is useful to save the CLC ".vtu" file only to show the CLC polygon boundaries. However, the colours of the boundaries of the CLC polygons are not representative: for each segment there are actually two overlapping segments (one per adjacent polygon) but only one segment is visible (normally, the one associated with the external polygon) and it seems unfeasible to see the right colour of the internal polygon, even using the opacity options. To extracting the CLC .vtu file, the following information might be useful. Only mandatory data are saved (otherwise the file would be too large to be opened with Paraview): "objectid", "codice_num". The format of the CLC .vtu file reads:
    - ✓ "objectid": CLC polygon ID associated with the segment (each actual segment is represented by two twin segments associated with the inner/outer polygon);
    - ✓ "code_num": CLC code of the polygon associated with the segment;
    - ✓ "offsets": last-point ID for each segment (called "cell");
    - ✓ "connectivity": point-ID list for each segment (called "cell");
    - ✓ "Points": list of vertex coordinates (triplets *xyz*).

   The CLC ".vtu" file file cannot be used to reproduce the ".ply" and ".txt" files for SPHERA without using PV v.5.9.0 to read again the ".shp" file. The CLC ".vtu" file cannot be directly provided to SPHERA because a Delaunay filter is mandatory to partition the CLC polygons. "Dealunay2D" can be applied to ".vtu" files, but the attribute of the CLC code is lost.
7. Apply "Delaunay2D" filter to export many ".ply" files (one per CLC polygon). After, only triangles are present in all the ".ply" files (one per CLC polygon). The Linux tool "tar" is mandatory to easily manage the ".ply" files. Save screenshot plotting the CLC polygons (use the visualization mode "surface without edges" because edges are too many).
8. It seems that Paraview cannot wite ".ply" files with scalar attributes even if it can read them. It is mandatory to add a ".txt" file to provide the CLC code. Use the following command chain: "Merge blocks" -> "Spreadsheet" (visualize only CLC code and CLC polygon ID -Paraview "BlockNumber"-; increasing order to be set on "BlockNumber") -> "Export" the visible spreadsheet. Saving is very slow (up to ca.20' for 1M records). Unfortunately, with Paraview v.5.9.0, "Save Data" does not save the block ID of a multi-block dataset (like a CLC file). It would be interesting to try with Paraview v.4.0.1 to possibly save time in exporting the ".txt" file.

9. It seems there is no way to save the ”.shp” file in other formats (by means of “Save Data) during the intermediate stages of the above steps.

11.15. SPHERA

Once the sections “VERTICES” and “FACES” are obtained from ply2SPHERA_perimeter and the input file for the positioning surface grid is produced by SnappyHexMesh, one can fill all the input files of SPHERA. This 3D CFD-SPH code is executed to simulate the propagation of floods, landslides and water waves. The present documentation file is completely dedicated to this code. The following procedure might be useful to adapt the template labels when starting a new test case:

1. Copy and paste the template input folder;
2. Rename the input files depending on the test case (unique Linux command line using the command “rename”);
3. Replace the input file name within the input files (unique Linux command line using the commands “find” and “sed”);
4. Erase the lines on the test case label within the input files (unique Linux command line using the comands “find” and “sed”);
5. Insert the proper lines on the test case label at the proper line number (unique Linux command line using the comands “find” and “sed”);
6. Erase the lines on the test case description within the input files (unique Linux command line using the comands “find” and “sed”);
7. Insert the proper lines on the test case description within the input files (unique Linux command line using the comands “find” and “sed”).

11.16. Gnuplot

The output files of SPHERA which contain the profiles (1D) of the fluid dynamics variables are visualized by means of Gnuplot (Williams & Kelley, 2021, [99]), which returns the output file in the “.eps” format.

11.17. GSView

These files are read by GSView (Ghostgum Software Pty Ltd, 2021, [100]) and converted in the “.png” format.

11.18. Grid Interpolator: post-processing of the 2D output fields of SPHERA

The output files of SPHERA which contains the synthetic fluid dynamics fields need a following elaboration by means of Grid Interpolator (RSE SpA, 2022, [97]).
The software tool Grid Interpolator (2022, [97]) executes Shepard interpolations to any 3D input field with possible application of a despiking procedure. Grid Interpolator (2022, [97]) reads several DTM input files. Their overlapping areas have different grid points depending on the file. The outline of each input DTM is irregular, and its edges are not aligned with the Cartesian axes. Grid Interpolator (2022, [97]) produces a unique output DTM with variable spatial resolution. This is partitioned in adjacent DTM files with shared boundaries and without overlapping areas. Reference system conversions apply to switch from the local SPH reference system to the cartographic or the geographic reference system. Grid Interpolator (2022, [97]) is also used to post-process the 2D fields of the specific flow rate, the depth-average speed and the water depth.
Grid Interpolator v.3.0 (2022, [97]) reads a 3D field of values from an input grid and interpolates them on an output grid with a different spatial resolution. The input file is a xyz file (with two additional ad-hoc lines at the beginning). The output field is available in both the file formats xyz and DEM. This tool is also useful to post-process the 2D fields of the maximum specific flow rate and the maximum water depth as estimated by SPHERA v.10.0.0 (RSE SpA).
Two additional lines are reported at the beginning of the xyz input file “input_field.prn”, as in the following example:

“

```
n_points_in      x_min       y_min       z_min       x_max       y_max       z_max       dx_out      dy_out
dz_out
       21822  1152.77000    71.07100     0.00000 25779.70000  9926.20000     1.00000     9.47609
9.47609     1.00000
```

“

with the following parameter definition:
n_points_in: number of points in the input file;
x_min: minimum x-coordinate;
y_min: minimum y-coordinate;
z_min: minimum z-coordinate;
x_max: maximum x-coordinate;
y_max: maximum y-coordinate;
z_max: maximum z-coordinate;
dx_out: spatial resolution of the output field along the x-axis direction;
dy_out: spatial resolution of the output field along the y-axis direction;
dz_out: spatial resolution of the output field along the z-axis direction.
The format of the first additional line does not alter the tool execution.
Fortran format specifier of the second additional line is ‘(i12,9(g12.5))’ .
The conversion from cartographic to geographic horizontal coordinates follow the same assumptions reported in DEM2xyz (2022, [95]). Here the linear conversion formula is inverted, with no need to express the latitude and longitude increments (already computed by DEM2xyz, 2022, [95]):

$$\begin{aligned} \frac{(\lambda-\lambda_{\min})}{\Delta\lambda} &= \frac{(X-X_{\min})}{\Delta X} \Rightarrow \lambda = \frac{(X-X_{\min})}{\Delta X}\Delta\lambda + \lambda_{\min}, \quad X_{\min}=0 \\ \frac{(\phi-\phi_{\min})}{\Delta\phi} &= \frac{(Y-Y_{\min})}{\Delta Y} \Rightarrow \phi = \frac{(Y-Y_{\min})}{\Delta Y}\Delta\phi + \phi_{\min}, \quad Y_{\min}=0 \end{aligned} \tag{11.2}$$

11.19. Paraview: post-processing and visualization of the 2D and 3D output fields of SPHERA

Paraview shows the (2D and 3D) fluid dynamics fields produced by SPHERA and returns the associated image files.

11.20. Image Magick

The images above are concatenated in “.gif” animations by means of Image Magick (ImageMagick Studio LLC, 2021, [101]).

11.21. Virtual Dub

The compression of these animations, necessary in case the concatenation involves many files, is carried out by means of Virtual Dub (Avery Lee, 2021, [102]), which returns an “.avi” video output file.

12. GNU FREE DOCUMENTATION LICENSE

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

References

[1] SPHERA (RSE SpA), “https://github.com,” 2022. [Online].

[2] A. Amicarelli, B. Kocak, S. Sibilla and J. Grabe, “A 3D Smoothed Particle Hydrodynamics model for erosional dam-break floods,” *International Journal of Computational Fluid Dynamics,* vol. 31, no. 10, pp. 413-434, DOI: 10.1080/10618562.2017.1422731, 2017.

[3] S. Manenti, A. Amicarelli and S. Todeschini, “WCSPH with Limiting Viscosity for Modeling Landslide Hazard at the Slopes of Artificial Reservoir,” *Water,* vol. 10, no. 4, pp. 515, doi:10.3390/w10040515, 2018.

[4] A. Amicarelli, R. Albano, D. Mirauda, G. Agate, A. Sole and R. Guandalini, “A Smoothed Particle Hydrodynamics model for 3D solid body transport in free surface flows,” *Computers & Fluids,* vol. 116, p. 205–228; DOI 10.1016/j.compfluid.2015.04.018, 2015.

[5] A. Amicarelli, S. Manenti, R. Albano, G. Agate, M. Paggi, L. Longoni, D. Mirauda, L. Ziane, G. Viccione, S. Todeschini, A. Sole, L. Baldini, D. Brambilla, M. Papini, M. Khellaf, B. Tagliafierro, L. Sarno and G. Pirovano, “SPHERA v.9.0.0: a Computational Fluid Dynamics research code, based on the Smoothed Particle Hydrodynamics mesh-less method,” *Computer Physics Communications,* vol. 250, pp. 107157, https://doi.org/10.1016/j.cpc.2020.107157, 2020.

[6] M. Paggi, A. Amicarelli and P. Lenarda, “SPH modelling of hydrodynamic lubrication: laminar fluid flow - structure interaction with no-slip conditions for slider bearings,” *Computational Particle Mechanics,* vol. 8, pp. 665–679; https://doi.org/10.1007/s40571-020-00362-1, 2021.

[7] A. Amicarelli, E. Abbate and A. Frigerio, “SPH modelling of a dike failure with detection of the landslide sliding surface and damage scenarios for an electricity pylon,” *International Journal of Computational Fluid Dynamics,* vol. 36, no. 4, pp. 265-293, DOI 10.1080/10618562.2022.2108020, 2022.

[8] A. Di Monaco, S. Manenti, M. Gallati and S. Sibilla, “SPH modeling of solid boundaries through a semi-analytic approach,” *Engineering Applications of Computational Fluid Mechanics,* vol. 5, no. 1, pp. 1-15, DOI: 10.1080/19942060.2011.11015348, 2011.

[9] A. Amicarelli, G. Agate and R. Guandalini, “A 3D Fully Lagrangian Smoothed Particle Hydrodynamics model with both volume and surface discrete elements,” *International Journal for Numerical Methods in Engineering,* vol. 95, pp. 419–450, DOI: 10.1002/nme.4514, 2013.

[10] A. Amicarelli, S. Manenti and M. Paggi, “SPH modelling of dam-break floods, with damage assessment to electrical substations,” *International Journal of Computational Fluid Dynamics,* vol. 35, no. 1-2, pp. 3-21; DOI 10.1080/10618562.2020.1811240, 2021.

[11] S. Manenti, S. Sibilla, M. Gallati, G. Agate and R. Guandalini, “SPH Simulation of Sediment Flushing Induced by a Rapid Water Flow,” *Journal of Hydraulic Engineering ASCE,* vol. 138, no. 3, pp. 272-284, DOI: 10.1061/(ASCE)HY.1943-7900.0000516, 2012.

[12] J. Monaghan, “Smoothed particle hydrodynamics,” *Rep. Prog. Phys.,* vol. 68, pp. 1703–1759, DOI: 10.1088/0034-4885/68/8/R01, 2005.

[13] T. a. C.-). SPHERIC (SPH scientific and industrial community affiliated to ERCOFTAC -European Research Community On Flow. [Online]. Available: http://spheric-sph.org/ , http://spheric-sph.org/sph-projects-and-codes. [Accessed 2019].

[14] A. Amicarelli and et al., “A Local-form 3-stage Arbitrary-Lagrangian-Eulerian Conservative-Consistent SPH code with applications to water waves and marine energy,” p. preprint, 2022.

[15] A. Amicarelli, J.-C. Marongiu, F. Leboeuf, J. Leduc, M. Neuhauser, L. Fang and J. Caro, “SPH truncation error in estimating a 3D derivative,” *International Journal for Numerical Methods in Engineering,* vol. 87, no. 7, pp. 677-700, DOI: 10.1002/nme.3131, 2011.

[16] S. Adami, X. Hu and N. Adams, “A generalized wall boundary condition for smoothed particle hydrodynamics,” *Journal of Computational Physics,* vol. 231, p. 7057–7075, 2012.

[17] M. R. Hashemi, R. Fatehi and M. Manzari, “A modified SPH method for simulating motion of rigid bodies in Newtonian fluid flows,” *International Journal of Non-Linear Mechanics,* vol. 47, p. 626–638, 2012.

[18] F. Macia, L. Gonzalez, J. Cercos-Pita and A. Souto-Iglesias, “A Boundary Integral SPH Formulation - Consistency and Applications to ISPH and WCSPH-,” *Progress of Theoretical Physics,* vol. 128, no. 3, pp. 439-462, 2012.

[19] A. Mayrhofer, B. Rogers, D. Violeau and M. Ferrand, “Investigation of wall bounded flows using SPH and the uni fied semi-analytical wall boundary conditions,” *Computer Physics Communications,* vol. 184, p. 2515–2527, 2013.

[20] M. Ferrand, D. Laurence, B. Rogers, D. Violeau and C. Kassiotis, “Unified semi-analytical wall boundary conditions for inviscid laminar or turbulent flows in the meshless SPH method,” *International Journal for Numerical Methods in Fluids,* vol. 71, no. 4, pp. 446-472, 2013.

[21] M. Gomez-Gesteira, B. Rogers, R. Dalrymple and A. Crespo, “State-of-the-art of classical SPH for free-surface flows,” *Journal of Hydraulic Research,* vol. 48, no. Extra Issue, pp. 6-27; doi:10.3826/jhr.2010.0012, 2010.

[22] J.-C. Marongiu, F. Leboeuf, J. Caro and E. Parkinson, “Free surface flows simulations in Pelton turbines using an hybrid SPH-ALE method,” *J. Hydraul. Res.,* vol. 47, pp. 40–49, DOI: 10.1080/00221686.2010.9641244, 2010.

[23] D. Price, “Smoothed Particle Hydrodynamics and Magnetohydrodynamics,” *J. Comp. Phys.,* vol. 231, no. 3, pp. 759-794, DOI: 10.1016/j.jcp.2010.12.011, 2012.

[24] D. Le Touzé, A. Colagrossi, G. Colicchio and M. Greco, “A critical investigation of smoothed particle hydrodynamics applied to problems with free-surfaces,” *Int. J. Numer. Meth. Fluids,* vol. 73, pp. 660-691; DOI: 10.1002/fld.3819, 2013.

[25] D. Violeau and B. Rogers, “Smoothed particle hydrodynamics (SPH) for free-surface flows: past, present and future,” *Journal of Hydraulic Research,* vol. 54, no. 1, pp. 1-26, DOI: 10.1080/00221686.2015.1119209, 2016.

[26] H. Gotoh and A. Khayyer, “On the state-of-the-art of particle methods for coastal and ocean engineering,” *Coastal Engineering Journal,* vol. 60, no. 1, pp. 79-103; doi: 10.1080/21664250.2018.1436243, 2018.

[27] E. Saikali, G. Bilotta, A. Hérault and V. Zago, “Accuracy Improvements for Single Precision Implementations of the SPH Method,” *International Journal of Computational Fluid Dynamics,* vol. 34, no. 10, pp. 774-787, DOI: 10.1080/10618562.2020.1836357, 2020.

[28] P. Sun, A. Colagrossi, S. Marrone, M. Antuono and A.-M. Zhang, “A consistent approach to particle shifting in the δ-Plus-SPH model,” *Computer Methods in Applied Mechanics and Engineering,* vol. 348, pp. 912-934, DOI: 10.1016/j.cma.2019.01.045, 2019.

[29] A. Di Mascio, M. Antuono, A. Colagrossi and S. Marrone, “Smoothed particle hydrodynamics method from a large eddy simulation perspective,” *Physics of Fluids,* vol. 29, no. 035102, p. doi: 10.1063/1.4978274, 2017.

[30] A. Khayyer, H. Gotoh, Y. Shimizu, K. Gotoh, H. Falahaty and S. Shao, “Development of a projection-based SPH method for numerical wave flume with porous media of variable porosity,” *Coastal Engineering,* vol. 140, pp. 1-22; DOI: 10.1016/j.coastaleng.2018.05.003, 2018.

[31] A. Khayyer, Y. Shimizu, H. Gotoh and K. Nagashima, “A coupled incompressible SPH-Hamiltonian SPH solver for hydroelastic FSI corresponding to composite structures,” *Applied Mathematical Modelling,* vol. 94, pp. 242-271, DOI: 10.1016/j.apm.2021.01.011, 2021.

[32] M. Basa, N. Quinlan and M. Lastiwka, “Robustness and accuracy of SPH formulations for viscous flow,” *International Journal for Numerical Methods in Fluids,* vol. 60, no. 10, pp. 1127-1148, 2009.

[33] J. Monaghan, “Smoothed Particle Hydrodynamics,” *Annu. Rev. Astron. Astrophys,* vol. 30, pp. 543-74, 1992.

[34] J. Monaghan and J. Lattanzio, “A refined method for astrophysical problems,” *Astronomy and Astrophysics,* vol. 149, pp. 135-143, 1985.

[35] M. Gallati and G. Braschi, “Simulazione lagrangiana di flussi con superficie libera in problemi di idraulica,” in *L’ACQUA 5,* 2000.

[36] A. Amicarelli and et al., “SPH modelling of the Vajont dam-overtopping flood with wall functions adapted to flash floods and weir-like inlet sections,” p. preprint, 2022.

[37] P. Swamee, “Generalized rectangular weir equations,” *Journal of Hydraulic Engineering,* vol. 114, no. 8, pp. 945-949, 1988.

[38] A. Colagrossi and M. Landrini, “Numerical simulation of interfacial flows by smoothed particle hydrodynamics,” *Journal of Computational Physics,* vol. 191, no. 2, pp. 448-475, 2003.

[39] L. Armstrong, S. Gu and K. Luo, “Study of wall-to-bed heat transfer in a bubbling fluidised bed using the kinetic theory of granular flow,” *International Journal of Heat and Mass Transfer,* vol. 53, no. 21-22, pp. 4949-4959, 2010.

[40] D. Schaeffer, “Instability in the Evolution Equations Describing Incompressible Granular Flow,” *Journal of Differential Equations,* vol. 66, pp. 19-50, 1987.

[41] V. Kumaran, “Kinetic theory for sheared granular flows,” *C. R. Physique,* vol. 16, pp. 51-61, 2015.

[42] L. Van Rijn, Principles of sediment transport in rivers, estuaries, and coastal seas, Aqua Publications, 1993.

[43] L. Van Rijn, “The prediction of Bed-Forms and Alluvial Roughness, Report,” Delft Hydraulics laboratory, The Netherlands, 1982.

[44] G. Seminara, L. Solari and G. Parker, “Bed load at low Shields stress on arbitrarily sloping beds: Failure of the Bagnold hypothesis,” *Water Resour. Res.*, vol. 38, no. 11, pp. 1249, doi:10.1029/2001WR000681, 2002.

[45] F. Morrison, An Introduction to Fluid Mechanics, New York: Cambridge University Press, 2013.

[46] A. Amicarelli, G. Agate, B. Stefanova, S. Sibilla and J. Grabe, “A SPH model for dike overtopping and dam liquefaction with bed-load transport, bottom drag and mobile boundaries,” in *11th SPHERIC ERCOFTAC International Workshop SPHERIC 2016 – ERCOFTACT Conference, pp.424-431, Garching (Munich, Germany); ISBN: 9783000533587,* 2016.

[47] J. Vila, “On particle weighted methods and Smooth Particle Hydrodynamics,” *Mathematical Models and Methods in Applied Sciences,* vol. 9, no. 2, pp. 161-209, 1999.

[48] A. Amicarelli, B. Kocak, S. Sibilla and J. Grabe, “A 3D Smoothed Particle Hydrodynamics model for erosional dam-break floods,” *International Journal of Computational Fluid Dynamics,* vol. 31, no. 10, pp. 413-434, 2017.

[49] M. Paggi, A. Amicarelli and P. Lenarda, “SPH modelling of hydrodynamic lubrication: laminar fluid flow - structure interaction with no-slip conditions for slider bearings,” *Computational Particle Mechanics,* vol. 8, pp. 665–679; https://doi.org/10.1007/s40571-020-00362-1, 2021.

[50] “SRTM3/DTED1 (USGS),” 2020. [Online]. Available: http://earthexplorer.usgs.gov/.

[51] B. Kosztra, G. Büttner, G. Hazeu and S. Arnold, “Updated CLC illustrated nomenclature guidelines,” European Environment Agency, European Topic Centre on Urban, land and soil systems; ETC/ULS Service Contract No 3436/R0-Copernicus/EEA.57441, Task 3, D3.1 – Part 1., 2019.

[52] K. Jancewicz and M. Szymanowski, “The Relevance of Surface Roughness Data Qualities in Diagnostic Modeling of Wind Velocity in Complex Terrain: A Case Study from the Snieznik Massif (SW Poland),” *Pure and Applied Geophysics,* vol. 174, no. 2, pp. 569-594, 2017.

[53] “GDAL,” OSGEO, 2021. [Online]. Available: https://github.com/OSGeo/gdal.

[54] Paraview (Kitware). [Online]. Available: https://github.com/Kitware/ParaView. [Accessed 15 July 2021].

[55] CFX (Ansys). [Online]. Available: http://www.ansys.com/products/fluids/ansys-cfx. [Accessed 2020].

[56] M. Paggi, A. Amicarelli and P. Lenarda, "SPH modelling of hydrodynamic lubrication: laminar fluid flow - structure interaction with no-slip conditions for slider bearings," *Computational Particle Mechanics,* vol. 8, pp. 665–679; https://doi.org/10.1007/s40571-020-00362-1, 2021.

[57] J. Monaghan, "Smoothed particle hydrodynamics," *Rep. Prog. Phys.,* vol. 68, p. 1703–1759, 2005.

[58] OpenFOAM (OpenCFD Ltd), "https://github.com/OpenFOAM/OpenFOAM-dev," 2021. [Online]. Available: https://github.com/isoAdvector/isoAdvector.

[59] J.-C. Marongiu, F. Leboeuf, J. Caro and E. Parkinson, "Free surface flows simulations in Pelton turbines using an hybrid SPH-ALE method," *J. Hydraul. Res.,* vol. 47, p. 40–49, 2010.

[60] A. Amicarelli, S. Manenti and M. Paggi, "SPH modelling of dam-break floods, with damage assessment to electrical substations," *International Journal of Computational Fluid Dynamics,* online.

[61] Hazus-MH, "Technical Manual, Multi-hazard Loss Estimation Methodology, Flood Model," Department of Homeland Security, Federal Emergency Management Agency, Mitigation Division, Washington D.C., USA, 2011.

[62] M. Holmes, Water infrastructure vulnerability due to dependency on third party infrastructure sectors, School of Civil Engineering and Geosciences, Newcastle University, May 2015; Doctorate thesis, 2015.

[63] M. Crawford and S. Seidel, "Weathering the storm: building business resilience to climate change," Center for climate and energy solutions, Arlington, VA, USA, 2013.

[64] M. Chow, L. Taylor and M. Chow, "Time of outage restoration analysis in distribution systems," in *IEEE Transactions on Power Delivery, 11(3):1652-1658,* 1996.

[65] P. Maliszewski and C. Perrings, "Factors in the resilience of electrical power distribution infrastructures," *Applied Geography,* vol. 32, pp. 668-679, 2012.

[66] D. Reed, "Electric utility distribution analysis for extreme winds," *Journal of wind engineering and industrial aerodynamics,* vol. 96, no. 1, pp. 123-140, 2008.

[67] MATTM, "Ministero dell'Ambiente e della Tutela del Territorio e del Mare (MATTM); 2018; Infrastrutture elettriche presenti sul territorio italiano," 2020. [Online]. Available: http://sinva.ancitel.it/mapviewer/index.html?collection=http://sinva.ancitel.it/WMC/Collection/VA/D5DB339E-DB14-9144-B043-B141DCABC108&context=http://sinva.ancitel.it/WMC/Context/VA/D5DB339E-DB14-9144-B043-B141DCABC108/963AD4B4-906A-4E53-B7FC-F497C3997138.

[68] "Git," Torvalds et al., 2021. [Online]. Available: https://github.com/git/git.

[69] A. Amicarelli and G. Agate, "SPH modelling of granular flows," in *9th SPHERIC ERCOFTAC International ERCOFTAC Workshop, 17-24,* Paris, 2014.

[70] A. Amicarelli and G. Agate, "A 3D SPH integrated model for granular flows with transport of solid bodies: model couplings and enhanced boundary treatment based on surface wall elements," in *10th SPHERIC ERCOFTAC International Workshop SPHERIC 2015 – ERCOFTACT Conference, 23-30; ISBN: 978-88-7847-487-1,* Parma (Italy), 2015.

[71] A. Amicarelli and M. Paggi, "SPH modelling for fluid-structure interactions with complex 3D surfaces: catastrophic dam-break and flood-control work on a real topography and hydrodynamic lubrication on rough surfaces," in *14th SPHERIC ERCOFTAC International Workshop SPHERIC 2019 - ERCOFTAC Conference, pp.9-16. ISBN: 978-0-902746-44-2,* Exeter University (UK), 2019.

[72] A. Amicarelli and G. Agate, "Modellazione fluidodinamica SPH per inondazioni con trasporto di corpi solidi, dam break per dighe in terra e moti oscillatori in serbatoi: sviluppi numerici e validazioni," project report (Ricerca di Sistema), RSE protocol 14001808, 2014.

[73] A. Amicarelli, "SPH modelling of an experimental urban dam-break flood and a rectangular lateral side weir for flood-control works," in *13th SPHERIC ERCOFTAC International Workshop SPHERIC 2018 - ERCOFTACT Conference, pp.103-109. ISBN: 978-1-908358-59-2,* Galway (Ireland), 2018.

[74] G. Agate and R. Guandalini, "The Use of 3D SPHERA Code to Support Spillway Design and Safety Evaluation of Flood Events," in *Proceedings of the 5th International SPHERIC Workshop, p.267-272,* 2010.

[75] R. Guandalini, G. Agate, A. Amicarelli, S. Manenti, M. Gallati and S. Sibilla, "SPH modelling of a 3D tsunami test case," *Development and applications of Ocean Engineering,* vol. 3, no. 1, pp. 11-21, 2014.

[76] gedit, GNOME Foundation, [Online]. Available: https://wiki.gnome.org/Apps/Gedit.

[77] gfortran (GNU, Free Software Foundation), "http://gcc.gnu.org/fortran/," 2021. [Online].

[78] "GNU Make," GNU, Free Software Foundation, [Online]. Available: https://www.gnu.org/software/make/.

[79] "gdb," GNU, Free Software Foundation, [Online]. Available: https://www.gnu.org/software/gdb/.

[80] "gprof," GNU, Free Software Foundation, 2021. [Online]. Available: http://sourceware.org/binutils/docs/gprof/.

[81] "Valgrind," Valgrind Developers, 2021. [Online]. Available: http://valgrind.org/.

[82] "Scalasca," Forschungszentrum Jülich & TU Darmstadt, 2021. [Online]. Available: http://www.scalasca.org/.

[83] "GitHub," GitHub Inc., 2021. [Online]. Available: www.github.com.

[84] "Free Software Foundation," 2021. [Online]. Available: http://www.fsf.org/.

[85] Intel Fortran Classic (ifort), "https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/fortran-compiler.html#gs.aiojbj," 2021. [Online].

[86] "idb," Intel Corporation, 2021. [Online]. Available: www.intel.com.

[87] "cpuinfo," Intel Corporation, 2021. [Online]. Available: www.intel.com.

[88] "VTUNE," Intel Corporation, 2021. [Online]. Available: www.intel.com.

[89] "Advisor," Intel Corporation, 2021. [Online]. Available: www.intel.com.

[90] TotalView, Rogue Wave Software, 2018. [Online]. Available: https://www.roguewave.com/products-services/totalview.

[91] "Engauge Digitizer (Mitchell et al.)," [Online]. Available: https://github.com/markummitchell/engauge-digitizer. [Accessed 15 July 2019].

[92] "SRTM3/DTED1 (USGS)," 2014. [Online]. Available: http://earthexplorer.usgs.gov/.

[93] M. Rexer and C. Hirt, "Comparison of free high resolution digital elevation data sets (ASTER GDEM2, SRTM v2.1/v4.1) and validation against accurate heights from the Australian National Gravity Database," *Australian Journal of Earth Sciences,* vol. 61, no. 2, pp. 213-226, DOI: 10.1080/08120099.2014.884983, 2014.

[94] Paraview (Kitware). [Online]. Available: https://github.com/Kitware/ParaView. [Accessed 15 July 2019].

[95] DEM2xyz (RSE SpA), "https://github.com/AndreaAmicarelliRSE/DEM2xyz," 2020. [Online].

[96] "ply2SPHERA_perimeter (RSE SpA)," 2021. [Online]. Available: https://github.com/AndreaAmicarelliRSE/ply2SPHERA_perimeter.

[97] Grid_Interpolator (RSESpA), "https://github.com/AndreaAmicarelliRSE/Grid_Interpolator," 2020. [Online].

[98] E. Plate, Urban Climates and Urban Climate Modelling, Kluwer Academic Publishers, 1995.

[99] Gnuplot (Williams & Kelley), "http://www.gnuplot.info/," 2021. [Online].

[100] "GSView," Ghostgum Software Pty Ltd, 2021. [Online]. Available: https://www.ghostscript.com/.

[101] "Image Magick," ImageMagick Studio LLC, 2021. [Online]. Available: https://www.imagemagick.org.

[102] "Virtual Dub," Avery Lee, 2021. [Online]. Available: http://www.virtualdub.org/.

[103] S. Manenti, S. Sibilla, M. Gallati, G. Agate and R. Guandalini, "SPH Simulation of Sediment Flushing Induced by a Rapid Water Flow," *Journal of Hydraulic Engineering ASCE,* vol. 138, no. 3, pp. 272-284, 2012.