Towards real-time simulation of physically realistic pressure applied to submerged bodies using explicit and semi-implicit SPH algorithms

Nicolas Gartner, Niels Montanari, M. Richier, Vincent Hugel, Ramprasad Sampath

Towards real-time simulation of physically realistic pressure applied to submerged bodies using explicit and semi-implicit SPH algorithms

M. Richier, Nicolas Gartner, Niels Montanari, Vincent Hugel, Ramprasad Sampath

Towards real-time simulation of physically realistic pressure applied to submerged bodies using explicit and semi-implicit SPH algorithms

Nicolas Gartner[1] Niels Montanari[2] Mathieu Richier[1] Vincent Hugel[1] Ramprasad Sampath[2]

***Abstract*—This paper compares different algorithms using the Smoothed Particle Hydrodynamics (SPH) discretization to provide realistic simulations of submerged bodies in water. Two different techniques for handling fluid-solid boundaries (mirroring and Shepard extrapolation) and two different pressure calculation methods (pressure projection and state equation) are investigated. The first contribution of this paper is to propose a test case with a submerged sphere in a water tank, in which the evolution of fluid pressure over depth is watched to compare the accuracy of the boundary handling techniques and the pressure calculations methods. The second contribution of this paper is to evaluate the accuracy of the solvers in dynamic cases, looking at the evolution of the pressure over time, the evolution of sphere speed over time and comparing with an analytical falling speed based on added mass and/or velocity damping. Third contribution of this paper is to evaluate the solvers using performance metrics based on computational time and time step counts. The results showed that the solver using pressure projection and Shepard extrapolation have reliable accuracy and good computational efficiency and might be a favored approach towards achieving accurate real-time simulation of underwater vehicles.**

I. Introduction

Realistic simulation of movements of submerged bodies like Autonomous Underwater Vehicles (AUVs) or Remotely Operated Vehicles (ROVs) in near real-time is a challenging issue. It could have interesting applications in the design of vehicles shape or the evaluation of the performance of vehicles capable of reconfiguring their structure during a mission.

A first method consists of using the rigid body model like the one described in Fossen [1] to update the vehicle dynamics, with the major drawback of requiring the identification of hydrodynamic parameters, such as drag coefficients, which is a costly procedure. This hydrodynamic parameter estimation can be done by setting up a real experiment, like in Ridao [2], where an unmanned underwater vehicles is put in a water tank and external sensors are used to measure fluid force and the vehicle position, or in Gartner [3], who proposed a method to evaluate these parameters using raw data from a free decay pendulum experiment, involving a sphere shaped object. Many upgrades to this method were made over time, which mainly focused on determining the hydrodynamic parameters for more degrees of freedom, with thruster coefficient or with fin coefficients.

Another method is based on numerically solving the Navier-Stokes equations to obtain the force applied by water on the hull of the vehicle. Common mesh-based techniques have difficulties in efficiently dealing with highly dynamic flows and fast-moving interfaces (free surface, fluid-solid interfaces). A more recent technique, namely Smoothed Particle Hydrodynamics (SPH), for which Violeau [4] made a state-of-the-art review, appears to be more efficient for this purpose. This method does not rely on a mesh but, instead, considers a set of interacting and moving particles to simulate the fluid. SPH solvers can be separated in different families: depending on the time integration method, which can be explicit, semi-implicit or implicit, and depending on the pressure calculation method.

Weakly Compressible SPH (WCSPH) solvers (Monaghan [5]) are solvers in which pressure is calculated through a state equation and are relatively simple to implement. In Incompressible SPH (ISPH) solvers, it is calculated with the pressure projection method, like in Cummins [6] or Shao [7]. WCSPH solvers generally have small time steps but low runtime per time step, while semi-implicit ISPH solvers generally have large time steps but high runtime per time step. There is no consensus on which approach is the most favorable. Comparisons of these algorithms have been made many times, for example in Shadloo [8] or in Lee [9], in which the comparison was only focusing on the fluid simulation and not on rigid-body dynamics and fluid-solid coupling, neither compared in depth computation efficiency of the algorithms.

There are three main approaches to deal with solid boundaries. One is to apply artificial forces with magnitude based on the distance to the boundary, but it is too inaccurate. Another is to calculate boundary integrals, which provides a good accuracy but at a too high computational cost (Violeau [4]). A last approach is to consider fictitious particles at the other side of the boundary. Among the variants relying on fictitious particles, the ones that generate the solid boundary particles only once at the beginning of the simulation are considered. Akinci [10] proposed a pressure mirroring technique that has the advantage of not requiring to calculate solid boundary particles pressure. In Adami [11] the pressure of the solid boundary particles is extrapolated from fluid particles. As both methods involve low computation time, they are good candidate solutions for real-time SPH simulations.

The work presented in this paper proposes to incorporate wall boundary handling techniques and two-way fluid-solid interaction with two SPH fluid solvers in order to enhance the physical realism and simultaneously enhance the speed of simulations of submerged bodies. In the next section, two efficient SPH algorithms with precomputations, stored variables and resolution step are provided, the coupling procedure between SPH solvers and rigid solvers is described. Section III shows comparative tests with a static sphere that interacts with the fluid. In section IV, the time evolution of the pressure gradient

[1]Laboratoire COSMER, Université de Toulon, France Mail: gartner@univ-tln.fr [2]Centroid LAB Inc, 3877 Grand View Blvd, Los Angeles, CA 90066, US

is studied with both algorithms to explain the influence of the modeling. The computational efficiency of both algorithms is analyzed. A test with a dynamic rigid body is conducted, and finally some hints for the improvement of the accuracy will be given.

II. Two SPH algorithms

A. SPH discretization

SPH main feature is to consider a smoothing function as spatial convolution product for quantities. It is called "kernel" function $W\left(|\boldsymbol{r_a}-\boldsymbol{r_b}|, R\right) = \alpha_W f_W\left(\frac{|\boldsymbol{r_a}-\boldsymbol{r_b}|}{R}\right) = W_{ab}$ and weighs the interaction between particles a and b from their distances. $\boldsymbol{r}$ is the position of particles. R is the radius of the support of the kernel function. The kernel is chosen so that, if the discretization of the spatial continuum is regular:

$$\sum_{b\in\mathcal{N}_a} W_{ab} \approx 1 \tag{1}$$

With $\mathcal{N}_a$ the whole set of particles inside the support limit R, particle a included.

In this paper, only 3D case is considered and the cubic B-spline is used (Violeau [12]):

$$\alpha_W = \frac{16}{\pi R^3}$$
$$f_W(q) = \begin{cases} \frac{1}{2} - 3q^2(1-q) & 0 \le q \le \frac{1}{2} \\ (1-q)^3 & \frac{1}{2} \le q \le 1 \\ 0 & q \ge 1 \end{cases} \tag{2}$$

Figure 1 represents the cubic B-spline (Eq. 2) $f_W\left(\frac{\boldsymbol{r_a}-\boldsymbol{r_b}}{R}\right)$ function and its derivative, which represents the gradient. The sign of $\boldsymbol{r_a}-\boldsymbol{r_b}$ was taken into account for the representation of the gradient to show the continuity at 0. The SPH gradient is supported by the unit vector $\boldsymbol{e}_{ab}$ in the direction $b \to a$. This is relevant for the understanding of the behavior of the pressure gradient that is used further on.

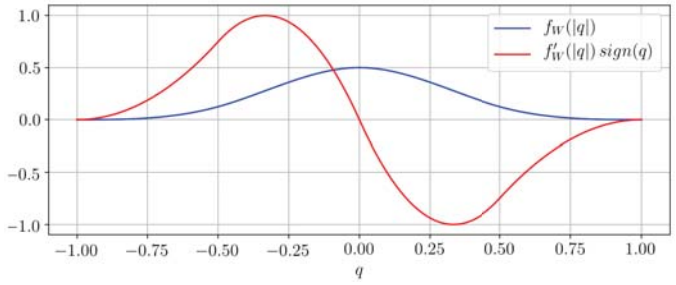

Fig. 1. Representation of $f_W(|q|)$ and $f'_W(|q|)\,sign(q)$ for the cubic B-spline.

Equation 3 shows the spatial interpolation of a thermodynamic quantity A (density, pressure, etc.) with SPH.

$$A_a = m\left(\frac{A_a}{\rho_a}W_{aa} + \sum_{b\in F_a\cup\mathcal{S}_a}\frac{A_b}{\rho_b}W_{ab}\right) \tag{3}$$

With $\mathcal{F}_a$ and $\mathcal{S}_a$ respectively the set of fluid and solid particles inside the kernel support centered on particle a, and m the particle mass. $m = \rho_0\,\delta r^3$, with δr the size of a particle and ρ_0 the rest density of the fluid. Here, the volume of a particle is δr^3, because in the initialization process, particles are generated on a regular grid and their mass has to remain constant. Computing $\mathcal{S}_a$ and $\mathcal{F}_a$ can be very costly, the method developed in Ihmsen [13] is used here. This method makes use of index sorting and ordering of particle data along a space-filling curve, which improves the performance by reducing cache-miss rates and avoiding data races during parallel execution. In particular, the density of particle a can be simply evaluated as:

$$\rho_a = m\left(W_{aa} + \sum_{b\in F_a\cup\mathcal{S}_a} W_{ab}\right) \tag{4}$$

SPH approximation is used to calculate the value of gradient $\boldsymbol{\nabla}A$, divergence $\boldsymbol{\nabla}\cdot\boldsymbol{A}$ and Laplacian $\boldsymbol{\nabla}^2\boldsymbol{A}$. (5, 6, 7), see Violeau [12] (chapter 5.2 and 6.2).

$$\boldsymbol{\nabla}A_a = m\,\rho_a \sum_{b\in\mathcal{F}_a\cup\mathcal{S}_a}\left(\frac{A_a}{\rho_a^2}+\frac{A_b}{\rho_b^2}\right)\boldsymbol{\nabla}W_{ab} \tag{5}$$

$$\boldsymbol{\nabla}\cdot\boldsymbol{A}_a = -\frac{m}{\rho_a}\sum_{b\in\mathcal{F}_a\cup\mathcal{S}_a}(\boldsymbol{A}_a-\boldsymbol{A}_b)\cdot\boldsymbol{\nabla}W_{ab} \tag{6}$$

$$\boldsymbol{\nabla}^2\boldsymbol{A}_a = 10m\sum_{b\in\mathcal{F}_a\cup\mathcal{S}_a}\frac{(\boldsymbol{A_a}-\boldsymbol{A_b})\cdot(\boldsymbol{r_a}-\boldsymbol{r_b})}{\rho_b\,|\boldsymbol{r_a}-\boldsymbol{r_b}|^2}\boldsymbol{\nabla}W_{ab} \tag{7}$$

B. Evaluation of the pressure gradient

Focusing on the evaluation of the gradient of pressure p, from Eq. 5, the following notations are introduced:

$$\boldsymbol{\Sigma}_a^F = m\sum_{b\in F_a}\boldsymbol{\nabla}W_{ab} \quad \text{and} \quad \boldsymbol{\Sigma}_a^S = m\sum_{s\in S_a}\boldsymbol{\nabla}W_{as} \tag{8}$$

$$\boldsymbol{\Sigma}_a = \boldsymbol{\Sigma}_a^F + \boldsymbol{\Sigma}_a^S \tag{9}$$

$$\Theta_a = {}^{p_a}/_{\rho_a^2} \tag{10}$$

$$\boldsymbol{\Gamma}_a = m\sum_{b\in\mathcal{F}_a\cup\mathcal{S}_a}\Theta_b\boldsymbol{\nabla}W_{ab} \tag{11}$$

$\boldsymbol{\Gamma}$ is the acceleration coming from the sum of the contribution of the neighboring particles and $\Theta\,\boldsymbol{\Sigma}$ is the contribution of the particle itself to the acceleration of its neighbors. Equation 12 is obtained by replacing A_a with p_a in Eq. 5.

$$\begin{aligned}\frac{1}{\rho_a}\boldsymbol{\nabla}p_a &= m\sum_{b\in\mathcal{F}_a\cup\mathcal{S}_a}(\Theta_a+\Theta_b)\,\boldsymbol{\nabla}W_{ab} \\ &= \Theta_a\boldsymbol{\Sigma}_a + \boldsymbol{\Gamma}_a\end{aligned} \tag{12}$$

C. WCSPH: Explicit Weakly Compressible SPH

Let's consider a Newtonian fluid and neglect surface tension, the famous continuity (13) and momentum (14) Navier-Stokes equations can be used:

$$\frac{d\rho}{dt} + \rho\boldsymbol{\nabla}\cdot\boldsymbol{v} = 0 \tag{13}$$

$$\frac{d\boldsymbol{v}}{dt} = -\frac{1}{\rho}\boldsymbol{\nabla}p + \nu\boldsymbol{\nabla}^2\boldsymbol{v} + \boldsymbol{g} \tag{14}$$

With p the pressure, ρ the density, $\boldsymbol{g}$ the gravity vector, $\boldsymbol{v}$ the velocity, ν the kinematic viscosity and ${}^{d}/_{dt}$ the Lagrangian

derivative of quantity. Considering a weakly compressible flow (Mach number $Ma < 0.1$), where the pressure of the fluid is considered linked to its variation on density, Murnaghan-Tait state equation can be applied here:

$$p = \frac{\rho_0 c_{s,num}^2}{\gamma}\left[\left(\frac{\rho}{\rho_0}\right)^\gamma - 1\right] \tag{15}$$

With the polytropic index γ, which has commonly for water a value of $\gamma = 7$. A numerical speed of sound $c_{s,num}$ is used, because the use of the physical speed of sound would lead to excessively low time steps.

$$c_{s,num} \geq 10\, max\left(V_{max}, \sqrt{|g|\, H}\right) \tag{16}$$

With V_{max} and H respectively the maximal expected speed and fluid height. In both schemes that are presented here, the density is interpolated using Eq. 4. The pressure is chosen relative to atmospheric pressure. To avoid occurrences of the tensile stability [14] and implicitly enforce the atmospheric pressure at the free surface (here, the interface with air), negative pressures are prevented. From Eq. 14, the acceleration $\dot{\boldsymbol{v}}_a^{\not P}$ due to non pressure force is updated first:

$$\begin{aligned}\dot{\boldsymbol{v}}_a^{\not P} &= \nu \boldsymbol{\nabla}^2 \boldsymbol{v}_a + \boldsymbol{g} \\ &= \frac{10 m \mu_{num}}{\rho_a} \sum_{b \in \mathcal{F}_a \cup \mathcal{S}_a} \frac{(\boldsymbol{v}_a - \boldsymbol{v}_b).(\boldsymbol{r}_a - \boldsymbol{r}_b)}{\rho_b |\boldsymbol{r}_a - \boldsymbol{r}_b|^2} \boldsymbol{\nabla} W_{ab} + \boldsymbol{g}\end{aligned} \tag{17}$$

With μ_{num} a numerical dynamic viscosity, higher than real dynamic viscosity, that helps for stabilizing the simulation. For solid particle ($b \in \mathcal{S}_a$), $\boldsymbol{v}_s$ is the slip velocity, which computation is explained in section II-E. Then, the total acceleration of fluid particles is computed thanks to:

$$\begin{aligned}\dot{\boldsymbol{v}}_a &= \dot{\boldsymbol{v}}_a^{\not P} - \frac{1}{\rho_a}\boldsymbol{\nabla} p_a \\ &= \dot{\boldsymbol{v}}_a^{\not P} - (\Theta_a \boldsymbol{\Sigma}_a + \boldsymbol{\Gamma}_a)\end{aligned} \tag{18}$$

Particle positions and velocities are finally updated with the symplectic Euler scheme. The WCSPH algorithm can be found in Appendix A.

D. ISPH: Semi-implicit Incompressible SPH

Compared to the WCSPH model, the fluid is here considered to be nearly incompressible. The continuity equation can subsequently be also expressed as:

$$\frac{d\rho}{dt} = \boldsymbol{\nabla} \cdot \boldsymbol{v} = 0$$

To update the fluid state n to state $n+1$, first, viscosity and gravity force effects from Eq. 14 are considered. Velocity $\boldsymbol{v}_a^{\not P}$ is obtained by time integration of $\dot{\boldsymbol{v}}_a^{\not P}$, coming from Eq. 17 but with the physical dynamic viscosity μ. From Eq. 13 and Eq. 6 a new value of the density of particles is computed:

$$\begin{aligned}\rho_a^{\not P} = \rho_a + m\,\Delta t \Bigg(&\sum_{b \in \mathcal{F}_a} (\boldsymbol{v}_a^{\not P} - \boldsymbol{v}_b^{\not P}) \cdot \boldsymbol{\nabla} W_{ab} + \\ &\sum_{s \in \mathcal{S}_a} (\boldsymbol{v}_a^{\not P} - \boldsymbol{v}_s^W) \cdot \boldsymbol{\nabla} W_{as}\Bigg)\end{aligned} \tag{19}$$

With $\boldsymbol{v}_s^W$ the velocity of the wall at the particle position.

A discretization of Eq. 14, with the viscosity and gravity terms being already accounted for in $\boldsymbol{v}_a^{\not P}$:

$$\boldsymbol{v}_a^{n+1} = -\frac{\Delta t}{\rho_a}\boldsymbol{\nabla} p_a + \boldsymbol{v}_a^{\not P} \tag{20}$$

Discretizing Eq. 13 at state $n+1$, using Eq. 20 and considering that, at state $n+1$ incompressibility of the fluid is being achieved, so $\rho_a^{n+1} = \rho_0$, we get:

$$\rho_a \boldsymbol{\nabla} \cdot \left(\frac{1}{\rho_a}\boldsymbol{\nabla} p_a\right) = \frac{\rho_0 - \rho_a^{\not P}}{\Delta t^2} \tag{21}$$

The objective is to find the pressures which enforce incompressibility, that we represent as the minimization of $\Delta\rho_a = \rho_0 - \rho_a^*$, with ρ_a^* the density of particle a after accounting for the pressure forces yielded by the pressures we solved for. Here an exact pressure projection is used on Eq. 21 for increased accuracy. Using the SPH approximations of divergence and gradient and by simplifying the expressions, a linear system $\boldsymbol{M}\boldsymbol{p} = \boldsymbol{b}$ is obtained from the combination of all particles:

$$\underbrace{x_{aa}}_{D_a} p_a + \underbrace{\sum_{b \neq a} x_{ab}\, p_b}_{\boldsymbol{R_a}\cdot\boldsymbol{p}} = b_a \tag{22}$$

$b_a = \frac{\rho_0 - \rho_a^{\not P}}{\Delta t^2}$ and with, in the case of extrapolation:

$$D_a = -\frac{(\boldsymbol{\Sigma}_a)^2 + \Sigma_a^{F,2}}{\rho_a^2} \tag{23}$$

$$\begin{aligned}\boldsymbol{R_a} \cdot \boldsymbol{p} = \Theta_a \Sigma_a^{F,2} + \boldsymbol{\Sigma}_a \cdot \boldsymbol{\Gamma}_a + \\ m \sum_{b \in F_a} (\Theta_b \boldsymbol{\Sigma}_b + \boldsymbol{\Gamma}_b) \cdot \boldsymbol{\nabla} W_{ab}\end{aligned} \tag{24}$$

With:

$$\Sigma_a^{F,2} = m^2 \sum_{b \in F_a} (\boldsymbol{\nabla} W_{ab})^2 \tag{25}$$

The relaxed Jacobi method is chosen to solve the linear system. This method has many advantages: there is no need to first build the matrix M and store it in memory and the rows of the system can be efficiently processed in parallel. The low convergence rate is generally not problematic when aiming only at a moderately high level of incompressiblity enforcement (1% to 0.01% of density deviation). All diagonal coefficients D_a being strictly negative, the convergence is guaranteed. The pressure after one iteration becomes:

$$p_a = (1-\omega)\, p_a + \omega \frac{b_a - \boldsymbol{R_a} \cdot \boldsymbol{p}}{D_a} \tag{26}$$

With ω the relaxation factor. The same principle as in WCPSH applies here: every time a new pressure is computed, it is replaced by $p = max(p, 0)$. New density is:

$$\rho_a^* = \rho_a^{\not P} + \Delta t^2 \left(D_a p_a + \boldsymbol{R_a} \cdot \boldsymbol{p}\right) \tag{27}$$

Finally, the level of incompressiblity enforcement is estimated by averaging over all fluid particles and compared to a threshold ϵ. A new iteration is performed if:

$$\frac{\underset{a\in\mathcal{F}}{\text{avg}}(\max(\rho_a^*,\rho_0))-\rho_0}{\rho_0} > \epsilon \tag{28}$$

Algorithms 2 and 3 describe the ISPH algorithm used and can be found in Appendix B.

E. Rigid bodies boundary handling

To simulate the interaction between fluid and solid, fictitious boundary particles s are generated from the surface of the solid shape, always on the opposite side of the fluid (Fig. 2). A sufficient number of layers is created so that the kernel of a fluid particle can be filled if it is close to the surface of the solid. We ensure that the number of fictitious boundary particle layers $n_{layer} \geq {}^{R}/_{\delta r} = \alpha_R$. With α_R the ratio of interaction radius and particle size.

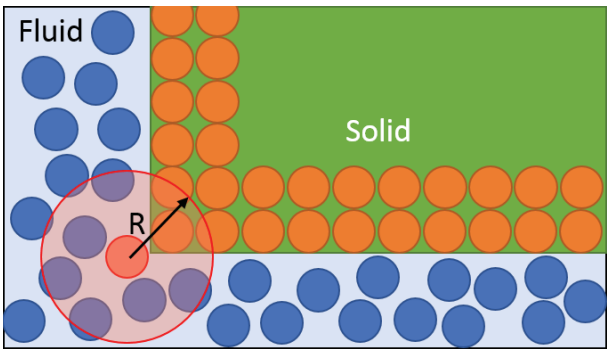

Fig. 2. Solid particles generated (orange) inside the solid shape (green) with fluid particles around (blue) and the kernel radius for a particular fluid particle (red).

Two boundary conditions are taken into account: no slip velocity condition and non-homogeneous Neumann pressure condition. To apply these conditions Shepard extrapolation method, that was applied by Adami [11], is used. Equations 29, 30 and 31 show how density, pressure and velocity of the boundary particles are computed.

$$\rho_s = \frac{\sum_{b\in\mathcal{F}_s}\rho_b W_{sb}}{\sum_{b\in\mathcal{F}_s} W_{sb}} \tag{29}$$

$$p_s = \frac{\sum_{b\in\mathcal{F}_s}[p_b+\rho_b(\dot{\boldsymbol{v}}_b^{\mathcal{P}}-\dot{\boldsymbol{v}}_s).(\boldsymbol{r}_s-\boldsymbol{r}_b)]W_{sb}}{\sum_{b\in\mathcal{F}_s} W_{sb}} \tag{30}$$

$$\boldsymbol{v}_s = 2\boldsymbol{v}_s^W - \frac{\sum_{b\in\mathcal{F}_s}\boldsymbol{v}_b W_{sb}}{\sum_{b\in\mathcal{F}_s} W_{sb}} \tag{31}$$

These values are unique for each particle and used for the fluid update computations. Another method that does not require the calculation of boundary particle pressure or density a contrario to Shepard extrapolation, introduced in Akinci [10], is explained in section II-H.

F. Dynamics of rigid bodies

Aside from the fluid solvers, a rigid dynamic solver is used to simulate the dynamics of rigid bodies. The principle of the rigid solver is to compute and sum up all the external forces applied on rigid bodies. For every solid fictitious particle, $\boldsymbol{F}_s = \boldsymbol{F}_s^P + \boldsymbol{F}_s^V$. With $\boldsymbol{F}_s^V$ the counter local viscous force that has to be applied on walls to respect Newton's third law:

$$\boldsymbol{F}_s^V = \frac{10m^2\mu}{\rho_s}\sum_{b\in\mathcal{F}_s}\frac{(\boldsymbol{v}_b-\boldsymbol{v}_s).(\boldsymbol{r}_b-\boldsymbol{r}_s)}{\rho_b|\boldsymbol{r}_b-\boldsymbol{r}_s|^2}\nabla W_{sb} \tag{32}$$

And $\boldsymbol{F}_s^P$ the local force computed from pressure (Eq. 30):

$$\boldsymbol{F}_s^P = -m^2\sum_{s\in S}\sum_{b\in\mathcal{F}_s}(\Theta_s+\Theta_b)\nabla W_{sb} \tag{33}$$

Then the resulting force at the center of mass (CM) is computed as follows:

$$\boldsymbol{F}^{Fluid\to Solid} = \sum \boldsymbol{F}_s \tag{34}$$

and the resulting momentum:

$$\boldsymbol{M}_{CM}^{Fluid\to Solid} = \sum(\boldsymbol{r}_s-\boldsymbol{r}_{CM})\times\boldsymbol{F}_s \tag{35}$$

Finally, the new velocity and position are calculated by time integration using velocity Verlet scheme with a constant time-step Δt_{Rigid}.

G. Time-step conditions

In the solvers presented in this paper, a dynamic time-step is used, like in Sampath [15]. To ensure a stable simulation, the time step depends on the simulation state and is given by the minimum value of these three conditions:

$$\Delta_{t,1} = \lambda_{CFL}\frac{\delta r}{\max(c_{s,num},\ \underset{a\in\mathcal{F}\cup\mathcal{S}}{\max}|\boldsymbol{v}_a|)} \tag{36}$$

$$\Delta_{t,2} = \lambda_{CFL}\sqrt{\frac{2\delta r}{\max(|\boldsymbol{g}|,\ \underset{a\in\mathcal{F}\cup\mathcal{S}}{\max}|\dot{\boldsymbol{v}}_a|)}} \tag{37}$$

$$\Delta_{t,3} = \lambda_{diff}\frac{\delta r^2}{\nu} \tag{38}$$

With λ_{CFL} and λ_{diff} constant time-step scaling coefficients. First and second conditions (Eq. 36, Eq. 37) are both related to Courant-Friedrichs-Lewy (CFL) condition, to prevent particles from moving too much at a single time step. The criterion based on the numerical speed of sound in condition (36) is specific to the WCSPH model. Third condition (Eq. 38) corresponds to a diffusion time-step to avoid overshoot during one timestep. In addition, the fluid solver and the rigid have separate time-step, with Δt being always a multiple of Δt_{Rigid}.

H. Fluid-solid interaction: Pressure mirroring technique

This section explains the mirroring boundary handling technique, that is different from Shepard extrapolation, and the changes it involves. Solid boundary particle contribution is directly taken into account in fluid particle acceleration and the computation of the force applied on rigid bodies. To apply this

method, the following changes are required: precomputation $\mathbf{\Gamma}_a$ from Eq. 11 is now $\mathbf{\Gamma}_a^{Mir}$:

$$\mathbf{\Gamma}_a^{Mir} = m \left[\sum_{b \in \mathcal{F}_a} \Theta_b \boldsymbol{\nabla} W_{ab} + \frac{1}{\rho_a} \sum_{b \in \mathcal{S}_a} (\dot{\boldsymbol{v}}_b^{\not P} - \dot{\boldsymbol{v}}_s).(\boldsymbol{r}_s - \boldsymbol{r}_b) \boldsymbol{\nabla} W_{as} \right] \quad (39)$$

The local pressure force applied on rigid bodies (Eq. 33) for this boundary handling method is computed with:

$$\boldsymbol{F}_s^P = -m^2 \sum_{s \in S} \sum_{b \in \mathcal{F}_s} \left(\Theta_b + \frac{\left(\dot{\boldsymbol{v}}_b^{\not P} - \dot{\boldsymbol{v}}_s \right) . (\boldsymbol{r}_s - \boldsymbol{r}_b)}{\rho_b} \right) \boldsymbol{\nabla} W_{sb} \quad (40)$$

The total acceleration from WCSPH solver given by Eq. 18 becomes:

$$\dot{\boldsymbol{v}}_a^{Mir} = \dot{\boldsymbol{v}}_a^{\not P} - \left(\Theta_a \mathbf{\Sigma}_a + \mathbf{\Gamma}_a^{Mir} \right) \quad (41)$$

And the Matrix $\boldsymbol{M}$ (Eq. 23 & Eq. 24) from ISPH becomes Eq. 42 and Eq. 43:

$$D_a = -\frac{(\mathbf{\Sigma}_a)^2 + \Sigma_a^{F,2} + \mathbf{\Sigma}_a \cdot \mathbf{\Sigma}_a^S}{\rho_a^2} \quad (42)$$

$$\boldsymbol{R_a} \cdot \boldsymbol{p} = \Theta_a \Sigma_a^{F,2} + \mathbf{\Sigma}_a \cdot \mathbf{\Gamma}_a^{Mir} + m \sum_{b \in F_a} \left(\Theta_b \left(\mathbf{\Sigma}_b + \mathbf{\Sigma}_b^S \right) + \mathbf{\Gamma}_b^{Mir} \right) \cdot \boldsymbol{\nabla} W_{ab} \quad (43)$$

III. Simulation experiments and results

A. Test scene description

For all tests, the same scene is used. Figure 3 is a view of the test scene, which consists of a rectangular tank containing generated fluid particles, in which a sphere of diameter $0.4\,m$ will be dropped. This figure also shows the generated rigid body boundary particles.

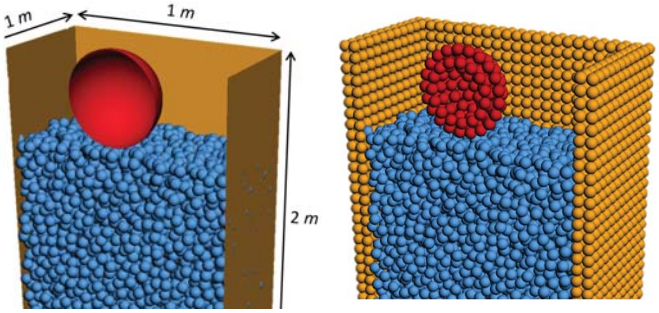

Fig. 3. Left: Simulation scene with the rigid bodies. Right: Simulation scene with the rigid bodies boundary particles.
Tank in yellow, sphere in red and fluid particles in blue.

Our fluid is water with standard physical properties of water at 293 K: rest density is $\rho_0 = 998\,kg.m^{-3}$ and dynamic viscosity is $\mu = 0.000998\,kg.m^{-1}.s^{-1}$ Computation time of the trials are saved and are summed up in Tab. III.

B. Comparison of the boundary handling techniques

1) Trial description: In order to compare the accuracy of pressure estimation resulting from the boundary handling methods, the first series of experiments consists of checking the hydrostatic pressure of still water inside a tank. We let the water particles stabilize for 3 seconds of simulated time, to let them self-rearrange and approximately satisfy the hydrostatic condition. A column of $1.62\,m$ of water is formed with about 13k particles. Parameter used for the WCSPH solver are given in Tab. I.

TABLE I
WCSPH algorithm parameters

$c_{s,num}$	80	δr	$0.05\,m$
λ_{CFL}	0.4	α_R	2
$\lambda_{diff} =$	0.125	μ_{num}	$0.01\,kg.m^{-1}.s^{-1}$

The choice of $c_{s,num}$ is about double the height criteria and largely above the speed criteria that will happen in this scene (Eq. 16). This relatively low value enables much larger time steps than if we were to use the physical speed of sound, the corresponding criterion in Eq. 36 being the most limiting, while keeping the compressibility at a moderate level. The value for δr corresponds, with respect to the tank dimensions, to a very coarse spatial resolution, to maximize computational performance.

2) Results: Figure 4 shows the whole pressure field obtained using the mirroring technique compared with using the extrapolation technique. The black curve represents the analytical solution of the evolution of static pressure as a function of depth.

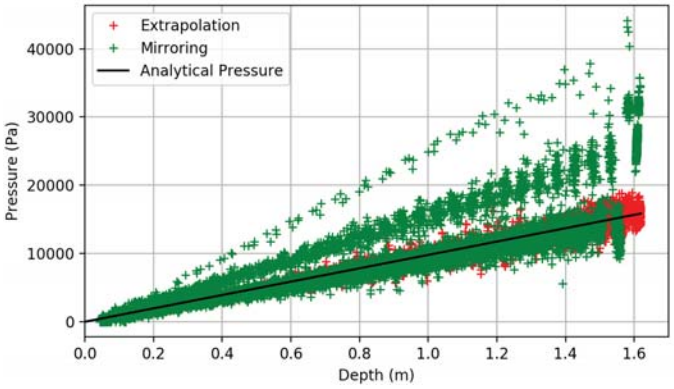

Fig. 4. Evolution of pressure versus depth using extrapolation and mirroring boundary techniques. (WCSPH)

This figure shows the excessive error made with the mirroring technique, which is sometimes higher than 100%. To see the localization of these errors, two pressure field layers at two different depths are drown on Figs 5 and 6. Pressure is color-coded. As water should have stabilized, the theoretical value is the hydrostatic pressure:

$$p_{th}(z) = \rho g z \quad (44)$$

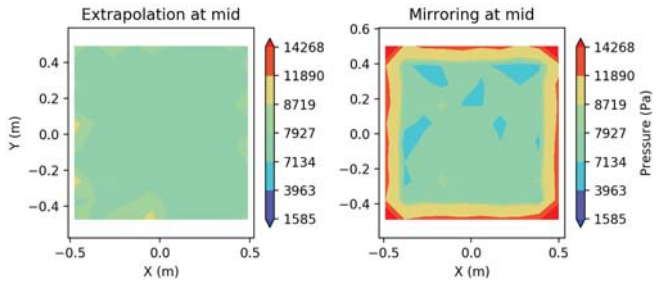

Fig. 5. View of the pressure field at middle depth with extrapolation and mirroring. $z = 0.81\,m$ and $p_{th} = 7931\,Pa$. (WCSPH)

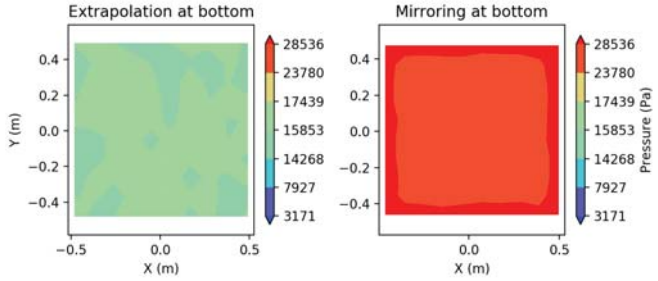

Fig. 6. View of the pressure field at the bottom of the tank with extrapolation and mirroring. $z = 1.62\,m$ and $p_{th} = 15862\,Pa$. (WCSPH)

For the mirroring technique, the pressure at the solid boundary is too high, especially at the corners, and it is even worse at the bottom of the scene because of the fictitious boundary particle influence from the bottom of the tank. This also happens for dynamic or submerged bodies and it has a detrimental effect on the fluid-solid pressure force interaction. Similar results are observed using the ISPH solver. Further on, the only technique used is Shepard extrapolation.

C. Comparison of the fluid solvers

1) Trial Description: In a second series of simulation tests with a submerged rigid body, pressure fields using the extrapolation method with the explicit WCSPH solver and the semi-implicit ISPH solver are compared. Water particles are generated and let stabilize freely for 3 seconds of simulated time. Then, the sphere is dropped from above surface and is stopped after reaching a certain depth. Water is then stabilized for 3 more seconds. Same parameters as for the first serie of tests are used for WCSPH (see Tab. I), and the parameters used for ISPH can be found in Tab. II. Most of them are common to both.

TABLE II
IPSH ALGORITHM PARAMETERS

λ_{CFL}	0.4	ϵ	0.0005
λ_{diff}	0.125	ω	0.5
δr	$0.05m$	Ω	0.6
α_R	2	it_{min}	5

Incompressibility factor ϵ is chosen small enough to give an accurate pressure computation but not too small to keep computation efficient. Further study could define how precisely pressure p can be computed with this value of ϵ. Relaxation ω and Ω factors are chosen to have a resilient algorithm.

2) Results: The extrapolated pressure on a sphere inside the water pressure field with ISPH and WCSPH is shown on Fig. 7. The presented pressure field is instantaneous and shown after stabilization. Green dots that represent the pressure on the boundary particles of the sphere are within the red dots and are close to the analytical pressure at each depth. For ISPH, the mean absolute error for the pressure on sphere boundary particles is 3.56% while for the fluid pressure it is 14.8%. For WCSPH, respectively 4.69% and 8.13%.

So both algorithms have correct precision for the evaluation of pressure on solid boundaries ($< 5\%$ error) and WCSPH seems to have better fluid pressure evaluation precision, but to go further, it has to be verified if this precision still holds in dynamic cases and if this precision is not achieved at the expense of efficiency.

IV. Discussion and Extension to Dynamic Simulation

A. Evolution of the pressure gradient during trial

An additional study is made to verify the stability of the algorithms to dynamic behavior. This time, the sphere is dropped from above the surface and stopped at different depths. Each time the sphere stops pressure is stabilized. The evolution over time of the pressure coefficient C_p and offset Off_p is analyzed:

$$p(z) = C_p\, z + \text{Off}_p \tag{45}$$

From Eq. 44, C_p should be $\rho\, g$ and Off_p should be 0. Figure 8 displays their evolution over time for the WCSPH solver. The black line displays the theoretical value of the coefficients.

Each time the sphere is stopped a shock happens to the fluid, which perturbates the presssure field. The pressure offset and coefficient stabilize and converge through time to the theoretical value. For comparison, Fig. 9 displays the evolution of C_p and Off_p through time for the ISPH solver.

The evolution of the coefficients with this solver is quite different from the WCSPH solver. As in ISPH the incompressibility is more strictly enforced and pressures are implicitly solved, variations in pressure are directly dissipated and it seems to be more reliable to handle shocks. However, the WCSPH solver converges to a more accurate static state. ISPH solver accuracy is directly linked to the incompressibility factor ϵ; reducing ϵ, of course, yields to a larger amount of computation. Further investigation could determine if the selected ϵ parameter gives a sufficiently accurate pressure for our purpose.

B. Algorithm efficiency: computation time

To evaluate the algorithm efficiency, computation times on two different machines are compared. Results from a mid-range computer and a higher-range computer are shown to

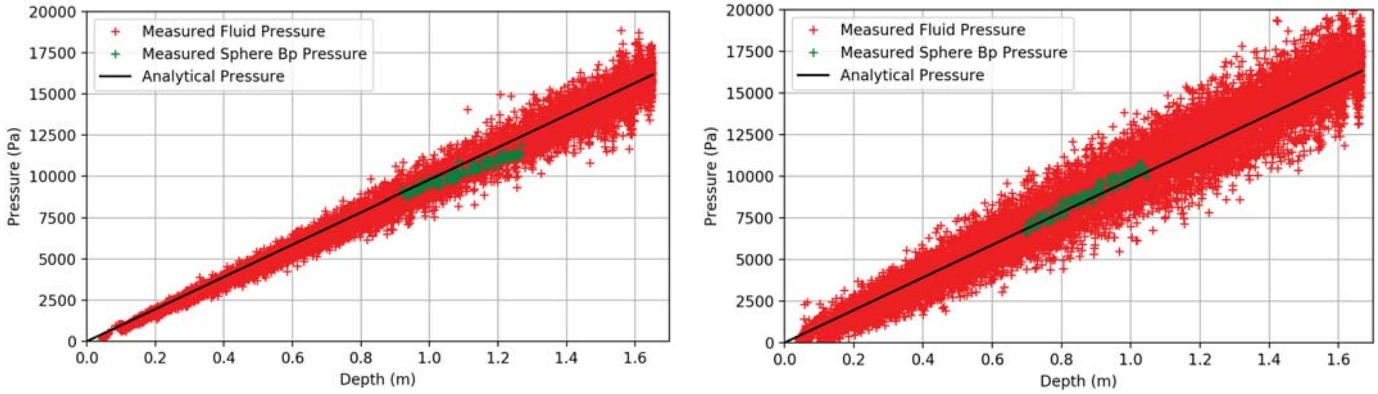

Fig. 7. Extrapolated pressure on the sphere with WCSPH *(left)* and ISPH *(right)*.

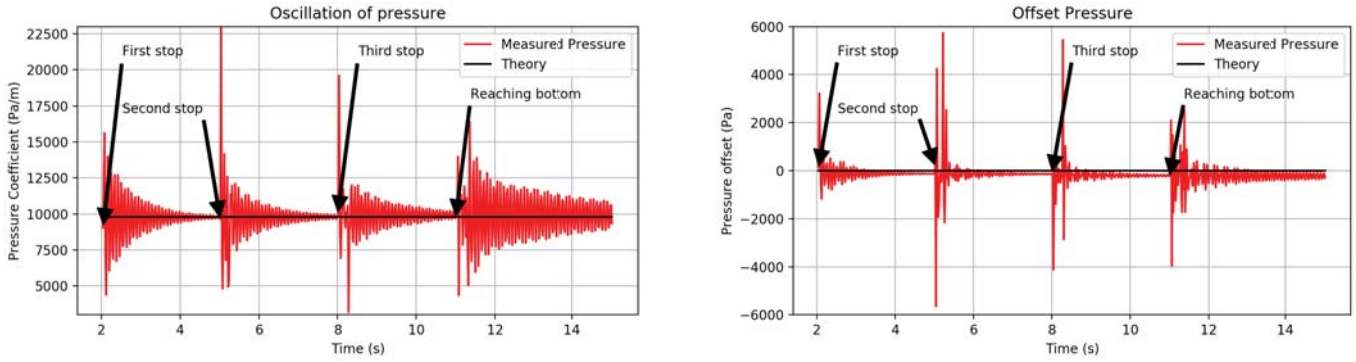

Fig. 8. Evolution of the coefficient of Eq. 45 through time in the case of WCSPH.

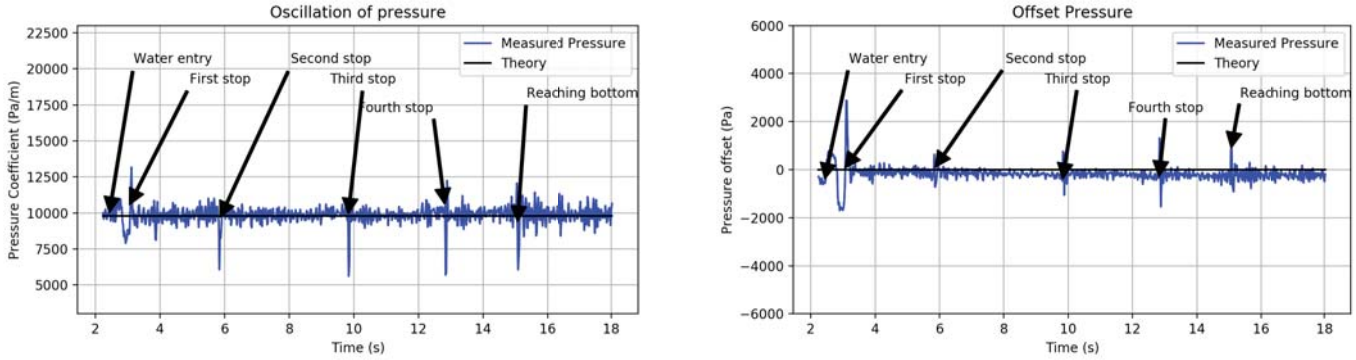

Fig. 9. Evolution of the coefficient of Eq. 45 through time in the case of semi-implicit ISPH.

demonstrate the capability of these algorithms to be used for real-time applications, even on standard PCs. Computations are made using C++ multi-threaded on CPU with the OpenMP API [16]. Our mid-range computer has an Intel Core i7 6700HQ CPU, 8 threads clocked at 2.6 GHz, 6 MB cache, 16 GB RAM and a 5400 rpm hard drive. Our high-range computer has an Intel Core i7 5820K CPU, 12 threads clocked at 3.3 GHz, 15MB cache, 32 GB RAM and a SSD hard drive. Table III sums up the average computation time for $1\,s$ of simulated time with the different solvers presented in this paper. As an addition to our study with 13k particles, experiments were made with a twice smaller particle size, which brings the scene to 104k fluid particles.

The gap in computation time between the two boundary handling methods is not big, at maximum $+15\%$ when using ISPH, and as seen before, Shepard extrapolation leads to much more accurate simulations, especially near the solid boundaries. More than 5 times less computation time is required with the ISPH solver. Although the cost per time step is higher, it is more than compensated by the much larger time-steps. A faster CPU clock and more available threads are making the simulations even faster reducing computation time per second of simulated time by 37%. Even if the real-time objective is not achieved yet, there is significant room for optimizing more,

TABLE III
MEAN COMPUTATION TIME (C. TIME) IN SECONDS, MEAN TIME STEP COUNT (T.STEP CNT) AND AVERAGE COMPUTATION TIME PER TIMESTEP FOR $1\,s$ OF SIMULATED TIME WITH THE DIFFERENT SOLVERS.

			Mid-Range			High-Range		
Solver	B. handling	Fluid Part.	C. time	T. step cnt.	Avg. C. time / T. step	C. Time	T. step cnt.	Avg. C. time / T. step
WCSPH	Mirroring	13k	167,13	4000,00	0,042	121,24	4000,00	0,030
	Shepard	13k	184,70	4000,00	0,046	108,66	4000,00	0,027
ISPH	Mirroring	13k	17,82	199,00	0,090	10,88	198,93	0,055
	Shepard	13k	20,45	196,49	0,104	12,98	196,47	0,066
WCSPH	Shepard	104k	1890,87	8000,00	0,236	1372,14	8000,00	0,172
ISPH	Shepard	104k	314,32	402,70	0,781	185,66	402,68	0,461

and it could be ported to a massively parallel architecture, such as a GPU.

C. Tests with a dynamic rigid body

A test was conducted with the sphere falling in water to check whether a correct dynamic behavior can be observed. At the beginning of the trial, the sphere is in the tank fully submerged and water is still, then the sphere is let fall freely. To compare our results, a dynamic model of the sphere, neglecting velocity damping, would be:

$$\dot{\boldsymbol{v}}_S = \frac{\rho_S - \rho_0}{\rho_S\,(1 + C_a)}\boldsymbol{g} \tag{46}$$

With C_a the added mass values and S referring to the solid sphere. C_a were chosen to be close to the experimental values found in Gartner [3].

Figure 10 shows the acceleration of a sphere of density $\rho_S = 2000\,kg.m^{-3}$ during one trial. Black curve and dotted black curves represent theoretical trajectories with two different added mass coefficients C_a.

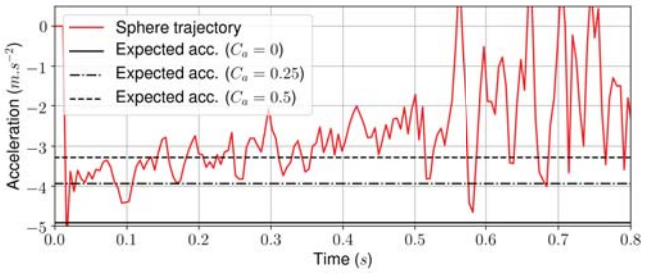

Fig. 10. Acceleration of the sphere during test with ISPH. Density of the sphere $2000\,kg.m^{-3}$.

At the beginning of the trial, the sphere acceleration is in the acceleration range obtained with the theoretical added mass coefficient. If the submerged sphere is to move with a speed above a certain threshold, a correct dynamic behavior cannot be observed, because the pressure field becomes noisy, see Fig. 11. This is very problematic for simulation of rigid bodies with densities close to water density. This might be due to the restriction of use of non-negative pressures, which causes disorder in the particle distribution. The incorporation of negative pressures combined with a procedure to prevent

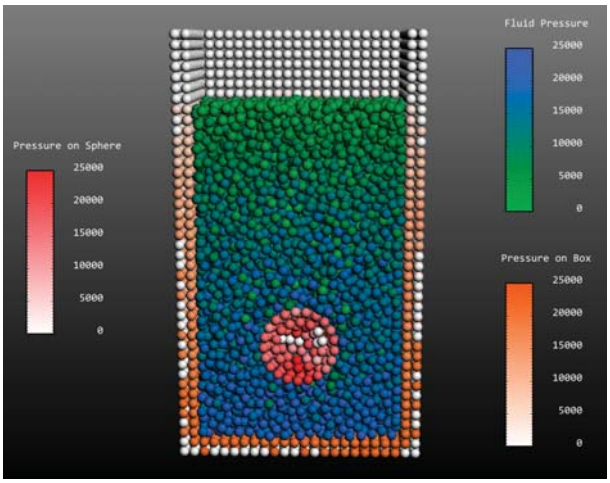

Fig. 11. Caption of the fall of a sphere of density $2000\,kg.m^{-3}$ with noisy pressure field.

tensile instability and to impose the atmospheric pressure at the free surface (see Section II-C) could constitute a more reliable solution for dynamic simulations.

To improve the accuracy of the results and obtain a better dynamic behavior, the use of the quintic B-spline kernel, that can be found in Violeau [12], and a higher interaction radius R, which allows more particles to be inside the neighborhood of a particle, are investigated. The amount of neighbor particles in the kernel, which is linked to the interaction radius choice, is limited by certain values for each kernel function, beyond which the so-called pairing instability may occur. This is explained by Dehnen [17]. Figure 12 shows the velocity of the sphere during time, with and without taking into account velocity damping (dotted black and black curves) in the dynamic model of the sphere (Gartner [3]) and choosing $C_a = 0.4$ which seems to be a correct average value. Equation 47 presents the dynamic model of the sphere with velocity damping.

$$\dot{\boldsymbol{v}}_S = \frac{\rho_S - \rho_0}{\rho_S\,(1 + C_a)}\boldsymbol{g} + \frac{\rho_0 C_d A_S}{2m_S\,(1 + C_a)}{\boldsymbol{v}_S}^2 \tag{47}$$

With $C_d = 0.4$ the sphere drag coefficient and A_S the projected area of the sphere.

As can be seen before having the perturbations of the noisy pressure field, the velocity of the sphere is very close to the

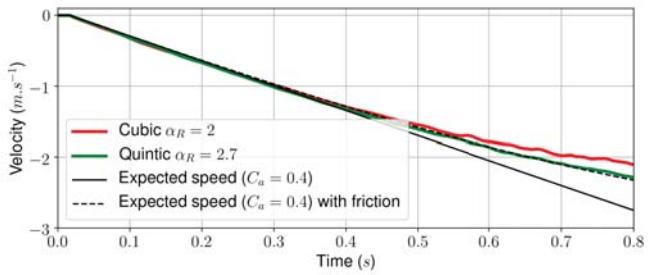

Fig. 12. Velocity of the sphere during fall with ISPH and Shepard extrapolation. Density of the sphere $2000\,kg.m^{-3}$.

analytical solution of the trajectory. This is very encouraging, taking into account that the discretization is coarse and that at the end of the trial, the sphere is very close to the bottom of the tank, which adds some boundary effects. The behavior of the sphere, with the same coarse discretization, is quite enhanced with the quintic B-spline kernel.

Figure 13 shows the absolute error between the analytical solution taking into account velocity damping and $C_a = 0.4$ and the two kernels.

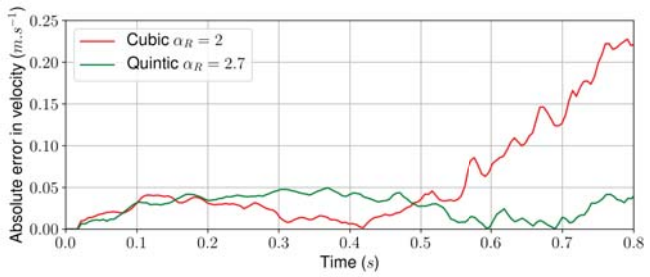

Fig. 13. Difference with analytical velocity. Sphere fall simulation with ISPH and Shepard extrapolation. Density of the sphere $2000\,kg.m^{-3}$.

These results could be enhanced, of course, with a less coarse discretization, but the fact that pressures are prevented from being negative appears to be a limiting factor. This shall be further investigated.

V. Conclusion and further work

In this study, the possibility to achieve a real-time sufficiently accurate fluid-solid coupling simulation has been investigated. First, the Shepard extrapolation boundary handling method from Adami [11] has proven to be more accurate than the mirroring method from Akinci [10] without causing a big loss in term of computation efficiency. The mirroring technique shown in this paper is too inaccurate to be used for fluid-solid dynamic simulation, further development on this technique will not be pursued as there is no real benefit in term of computation time. Results with a moving linear least-squares extrapolation, like the one from Band [18], which should give more accurate results than Shepard extrapolation, could be compared and the gain in accuracy versus computation cost could be also discussed. Secondly, even if the ISPH algorithm is less precise when considering a still water and static solid objects, it has proven to be more efficient and stable to dynamic perturbation than the WCSPH algorithm. Therefore, it appears to be more suitable for real-time simulations with dynamic fluid-solid coupling. Thirdly, a dynamic simulation has been tried with the ISPH solver, which highlighted that the impossibility for the solver to simulate negative values of pressure (pressure below atmospheric pressure) leads to a noisy pressure field. It was shown that the kernel choice and number of neighbors within the kernel support are key parameters to improve the simulation accuracy. Further development of the ISPH method will be conducted to allow negative pressure without causing instability. This is expected to enable a correct dynamic behavior of rigid bodies in the fluid. This would enable the identification of hydrodynamic parameters in simulation, and also the complete simulation of underwater vehicles missions using the SPH method and without having to determine hydrodynamic parameters beforehand.

VI. Acknowledgment

This work is a result of the collaboration of CENTROID LAB Inc. (*Los Angeles, USA*) which develops the Neutrino software and the Laboratoire COSMER (*Toulon, France*). This work is also partly supported by a DGA RAPID grant in partnership with SUBSEA-TECH (*Marseille, France*) and ROBOPEC (*Six-Fours-les-Plages, France*) and by the CARTT of Toulon Institute of Technology. (IUT *Toulon, France*)

References

[1] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control: Vademecum de Navium Motu Contra Aquas Et de Motu Gubernando*. JOHN WILEY & SONS INC, 2011.

[2] P. Ridao, A. Tiano, A. El-Fakdi, M. Carreras, and A. Zirilli, "On the identification of non-linear models of unmanned underwater vehicles," *Control Engineering Practice*, vol. 12, pp. 1483–1499, dec 2004.

[3] N. Gartner, M. Richier, and V. Hugel, "Hydrodynamics parameter identification of submerged bodies: numerical methods comparison and friction model analysis," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Madrid, Spain), Oct. 2018.

[4] D. Violeau and B. D. Rogers, "Smoothed particle hydrodynamics (SPH) for free-surface flows: past, present and future," *Journal of Hydraulic Research*, vol. 54, pp. 1–26, jan 2016.

[5] J. Monaghan, "Simulating free surface flows with SPH," *Journal of Computational Physics*, vol. 110, pp. 399–406, feb 1994.

[6] S. J. Cummins and M. Rudman, "An SPH projection method," *Journal of Computational Physics*, vol. 152, pp. 584–607, jul 1999.

[7] S. Shao and E. Y. Lo, "Incompressible SPH method for simulating newtonian and non-newtonian flows with a free surface," *Advances in Water Resources*, vol. 26, pp. 787–800, jul 2003.

[8] M. S. Shadloo, A. Zainali, M. Yildiz, and A. Suleman, "A robust weakly compressible SPH method and its comparison with an incompressible SPH," *International Journal for Numerical Methods in Engineering*, vol. 89, pp. 939–956, oct 2011.

[9] E.-S. Lee, C. Moulinec, R. Xu, D. Violeau, D. Laurence, and P. Stansby, "Comparisons of weakly compressible and truly incompressible algorithms for the SPH mesh free particle method," *Journal of Computational Physics*, vol. 227, pp. 8417–8436, sep 2008.

[10] N. Akinci, M. Ihmsen, G. Akinci, B. Solenthaler, and M. Teschner, "Versatile rigid-fluid coupling for incompressible SPH," *ACM Transactions on Graphics*, vol. 31, pp. 1–8, jul 2012.

[11] S. Adami, X. Hu, and N. Adams, "A generalized wall boundary condition for smoothed particle hydrodynamics," *Journal of Computational Physics*, vol. 231, pp. 7057–7075, aug 2012.

[12] D. Violeau, *Fluid Mechanics and the SPH Method: Theory and Applications*. OXFORD UNIV PR, 2012.

[13] M. Ihmsen, N. Akinci, M. Becker, and M. Teschner, "A parallel SPH implementation on multi-core CPUs," *Computer Graphics Forum*, vol. 30, pp. 99–112, nov 2010.
[14] J. Monaghan, "SPH without a tensile instability," *Journal of Computational Physics*, vol. 159, pp. 290–311, apr 2000.
[15] R. Sampath, N. Montanari, N. Akinci, S. Prescott, and C. Smith, "Large-scale solitary wave simulation with implicit incompressible sph," *Journal of Ocean Engineering and Marine Energy*, vol. 2, pp. 313–329, Aug 2016.
[16] L. Dagum and R. Menon, "OpenMP: an industry standard API for shared-memory programming," *IEEE Computational Science and Engineering*, vol. 5, no. 1, pp. 46–55, 1998.
[17] W. Dehnen and H. Aly, "Improving convergence in smoothed particle hydrodynamics simulations without pairing instability," *Monthly Notices of the Royal Astronomical Society*, vol. 425, pp. 1068–1082, aug 2012.
[18] S. Band, C. Gissler, A. Peer, and M. Teschner, "MIs pressure extrapolation for the boundary handling in divergence-free sph," 2018.

APPENDIX

A. Explicit WCSPH algorithm

Algorithm 1 WCSPH Solver

for each particle $a \in \mathcal{F}$ **do**
  Compute domains $\mathcal{F}_a$ and $\mathcal{S}_a$
Compute $\mathcal{S}^*$, $\forall a \in \mathcal{F}$ $(\mathcal{S}_a \subset \mathcal{S}^*)$
Compute δt with CFL conditions
for each particle $a \in \mathcal{F}$ **do**
  Compute $\forall b \in \mathcal{F}_a \cup \mathcal{S}_a, \boldsymbol{\nabla} W_{ab} = W'_{ab}\boldsymbol{e}_{ab}$
  Compute ρ_a (Eq. 4)
  Compute p_a (Eq. 15)
  Compute $\Theta_a = {}^{p_a}/_{\rho_a^2}$
if using Shepard extrapolation **then**
  for each particle $s \in \mathcal{S}^*$ **do**
    Extrapolate ρ_s and $\boldsymbol{v}_s$ (Eq. 29 & Eq. 31)
for each particle $a \in \mathcal{F}$ **do**
  Compute non-pressure acceleration $\dot{\boldsymbol{v}}_a^{\not P}$ (Eq. 17)
if using Shepard extrapolation **then**
  for each particle $s \in \mathcal{S}^*$ **do**
    Compute p_s (Eq. 30) and ρ_s using reverse (Eq. 15)
    Compute $\Theta_s = {}^{p_s}/_{\rho_s^2}$
for each solid S **do**
  Compute $\boldsymbol{F}_s^P$ and $\boldsymbol{F}_s^V$ depending on Shepard extrapolation or Mirroring
for each particle $a \in \mathcal{F}$ **do**
  Compute $\dot{\boldsymbol{v}}_a$ depending on Shepard extrapolation (Eq. 18) or Mirroring (Eq. 41)
  Compute $\boldsymbol{v}_a = \boldsymbol{v}_a + \delta t \dot{\boldsymbol{v}}_a$
  Compute $\boldsymbol{r}_a = \boldsymbol{r}_a + \delta t \boldsymbol{v}_a$

B. Semi-Implicit ISPH algorithm

Algorithm 2 ISPH Algorithm

for each particle $a \in \mathcal{F}$ **do**
  Compute domains $\mathcal{F}_a$ and $\mathcal{S}_a$
Compute $\mathcal{S}^*$, $\forall a \in \mathcal{F}$ $(\mathcal{S}_a \subset \mathcal{S}^*)$
Compute δt with CFL conditions
for each particle $a \in \mathcal{F}$ **do**
  Compute $\forall b \in \mathcal{F}_a \cup \mathcal{S}_a, \boldsymbol{\nabla} W_{ab} = W'_{ab}\boldsymbol{e}_{ab}$
  Compute ρ_a (Eq. 4)
if using Shepard extrapolation **then**
  for each particle $s \in \mathcal{S}^*$ **do**
    Extrapolate ρ_s and $\boldsymbol{v}_s$ (Eq. 29 & Eq. 31)
for each particle $a \in \mathcal{F}$ **do**
  Compute non-pressure acceleration $\dot{\boldsymbol{v}}_a^{\not P}$ (Eq. 17)
for each particle $a \in \mathcal{F}$ **do**
  $\boldsymbol{v}_a^{\not P} = \boldsymbol{v}_a + \Delta t\, \dot{\boldsymbol{v}}_a^{\not P}$
if using Shepard extrapolation **then**
  for each particle $s \in \mathcal{S}^*$ **do**
    Extrapolate pressure p_s (Eq. 30)
Compute $(p_a, \Theta_a, \boldsymbol{\Sigma}_a), \forall a \in \mathcal{F}$ and $(p_s, \Theta_s), \forall s \in \mathcal{S}^*$ with alg. 3
for each solid S **do**
  Compute $\boldsymbol{F}_s^P$ and $\boldsymbol{F}_s^V$ depending on Shepard extrapolation or Mirroring
for each particle $a \in \mathcal{F}$ **do**
  Compute $\boldsymbol{\Gamma}_a$ or $\boldsymbol{\Gamma}_a^{Mir}$
  Compute $\dot{\boldsymbol{v}}_a$ depending on Shepard extrapolation (Eq. 18) or Mirroring (Eq. 41)
  Compute $\boldsymbol{v}_a = \boldsymbol{v}_a + \delta t \dot{\boldsymbol{v}}_a$
  Compute $\boldsymbol{r}_a = \boldsymbol{r}_a + \delta t \boldsymbol{v}_a$

Algorithm 3 ISPH Pressure Solver Algorithm

for each particle $a \in \mathcal{F}$ **do**
  Compute $\rho_a^* = \rho_a^{\not P}$ (Eq. 19)
  $b_a = \frac{\rho_0 - \rho_a^{\not P}}{\delta t^2}$
  Compute $\boldsymbol{\Sigma}_a^F \Sigma_a^{F,2}, \boldsymbol{\Sigma}_a^S, \boldsymbol{\Sigma}_a$
  Compute D_a depending on Shepard extrapolation (Eq. 23) or Mirroring (Eq. 42)
  $p_a = \Omega\, p_a$, $\Omega < 1$ (Under relaxed first guess)
  Compute Θ_a
$it = 0$
while $\frac{\underset{a \in \mathcal{F}}{\text{avg}}(\max(\rho_a^*, \rho_0)) - \rho_0}{\rho_0} > \epsilon$ **and** $it_{min} < it$ **do**
  for each particle $a \in \mathcal{F}$ **do**
    Compute $\boldsymbol{\Gamma}_a$ or $\boldsymbol{\Gamma}_a^{Mir}$
  for each particle $a \in \mathcal{F}$ **do**
    Compute $\boldsymbol{R_a} \cdot \boldsymbol{p}$ depending on Shepard extrapolation (Eq. 24) or Mirroring (Eq. 43)
  for each particle $a \in \mathcal{F}$ **do**
    Compute p_a (Eq. 26)
    Compute ρ_a^* (Eq. 27)
    $b_a = \frac{\rho_0 - \rho_a^*}{\Delta t^2}$
  if using Shepard extrapolation **then**
    for each particle $s \in \mathcal{S}^*$ **do**
      Extrapolate pressure p_s (Eq. 30)
  $it = it + 1$