Politecnico di Torino

Politecnico di Torino

Master's Degree in Automotive Engineering
A.y. 2023/2024
October 2024

Development of an SPH solver for incompressible fluids with an interface to pre- and post-processing software

Supervisors:
Carlo Rosso

Candidate:
Riccardo Balducci

ABSTRACT

The smoothed-particles hydrodynamics method (SPH) – in which the continuum is modeled as the arbitrary lattice of interacting particles/interpolation points – is often advantageous in modeling scenarios involving extreme deformations, fluid motion, and fragmentation over traditional mesh-based techniques.

This study is focused on developing an SPH solver for simulating the motion of incompressible fluids, including the following tasks:

- Development of the interface with the pre- and post-processing software.
- Implementation of the SPH solver for Navier-Stokes equations in MATLAB.
- Identification of the most efficient technique for density calculation within the SPH solver through comparison of the "Summation Density" and the "Rate Density" algorithms.
- Validation of the developed SPH solver against three benchmark physical experiments: (i) dam break, (ii) dam break with a ramp, and (iii) fall of a droplet in water.
- Solving a practical problem using the developed SPH solver, involving a comparison of baffle designs to reduce sloshing in an automobile fuel tank.

ACKNOWLEDGEMENTS

I would like to thank Politecnico di Torino, University of Windsor and STELLANTIS for allowing me to take part in this experience. I also want to thank my advisors: Dr.Cherniaev and Dr. Rosso for their support and help in the completion of this master's thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1 Numerical methods

Numerical methods are computational techniques used to approximate solutions to complex or unsolvable mathematical problems. They utilize algorithms and computer simulations to find numerical solutions for various mathematical equations, including differential and integral equations, optimization problems, and systems of linear and nonlinear equations. These methods are essential when analytical solutions are too complex or non-existent, allowing professionals in fields such as physics, engineering, economics, biology, and finance to address a wide range of problems (Dahlquist 2003). The development and application of numerical methods have revolutionized scientific research and technological advancements by enabling the simulation of physical phenomena, the design of engineering systems, process optimization, data analysis, and prediction in real-world scenarios (Corless 2013).

In this wide realm, this work will focus on a specific branch of numerical methods, which is responsible for solving the partial differential equations (PDE). Here, several techniques have been developed and they can be distinguished into to two different categories:

- Mesh-based methods.
- Particle-based methods.

Currently, the most widely utilized mesh-based methods in industry include the Finite Element Method (FEM), the Finite Volume Method (FVM), and the Finite Difference Method (FDM) (Ye 2019). These methods involve discretizing the initial model using a mesh, enabling the solving of differential equations for each element (or volume) within the mesh. This approach effectively simplifies the complexity of the original problem (Bhavikatti 2005). A notable characteristic of these methods is the dynamic nature of the mesh, which deforms along with the model it represents (Lagrangian mesh), as displayed in Figure 1.

Conversely, the Eulerian mesh remains static and does not deform with the model it represents. Rather than following the motion of the material, the Eulerian approach observes changes within a fixed spatial grid. This can make it challenging to maintain a precise representation of moving interfaces or discontinuities.

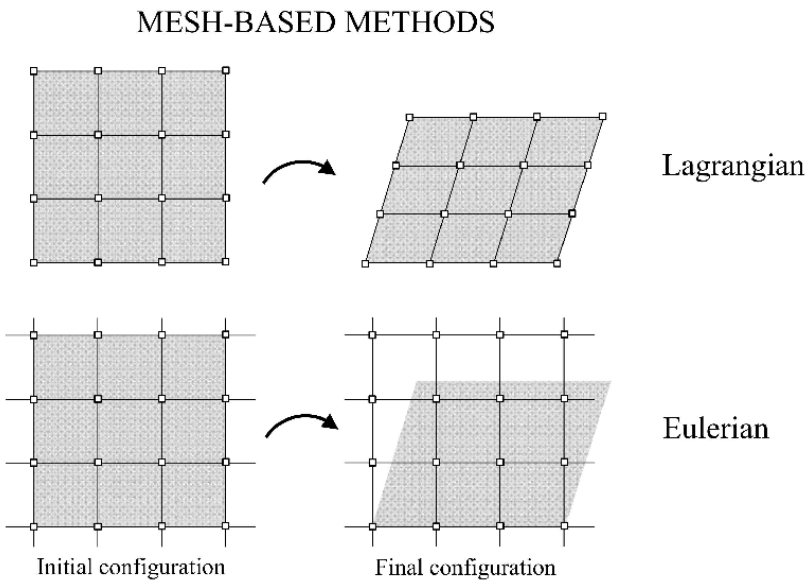


Figure 1 Lagrangian and Eulerian mesh comparison for mesh-based methods (Yerro 2015).

This adaptability can be advantageous given consistent mesh quality throughout the simulation. However, from the mesh quality definition provided by Knupp: "Mesh quality concerns the characteristics of a mesh that permit a particular numerical PDE simulation to be efficiently performed, with fidelity to the underlying physics, and with the accuracy required for the problem" (Knupp 2007), it appears evident that the mesh-based methods are strictly influenced by the mesh quality, highlighting an important limitation (Huerta 2004). Specifically, these methods encounter challenges in accurately resolving certain problems, such as those involving (Garg 2017):

- Large deformations
- Fragmentations
- Free surface objects
- Complex fluid motion

For instance, when addressing large deformation issues, the dynamic nature of the mesh necessitates re-meshing at regular intervals during simulation. However, severe deformations can distort mesh elements or volumes to such an extent that solving the differential equations yields inaccurate results (Li 2002), as highlighted in Figure 2. Understanding these numerical difficulties is crucial for refining the applicability of these methods to real-world scenarios.

Furthermore, in the context of fragmentation issues, the continual re-meshing of the model during simulation is significantly impacted. Fragmentation causes portions of the model to separate from the main

structure, resulting in a severely compromised mesh. Therefore, in FEM analysis the implementation of artificial techniques, such as the element erosion, becomes essential. However, this technique, which involves removing elements that exceed a certain strain threshold to manage severe material deformation, has notable drawbacks. It can lead to inaccuracies in representing the system's total mass and energy, create non-physical artifacts such as artificial voids and sharp edges, and produce results that are highly dependent on the initial mesh configuration. Additionally, this technique is sensitive to applied boundary conditions, potentially causing unrealistic stress concentrations and failure patterns. These issues necessitate careful consideration and validation to ensure accurate and reliable simulation outcomes (Yosef 2023).

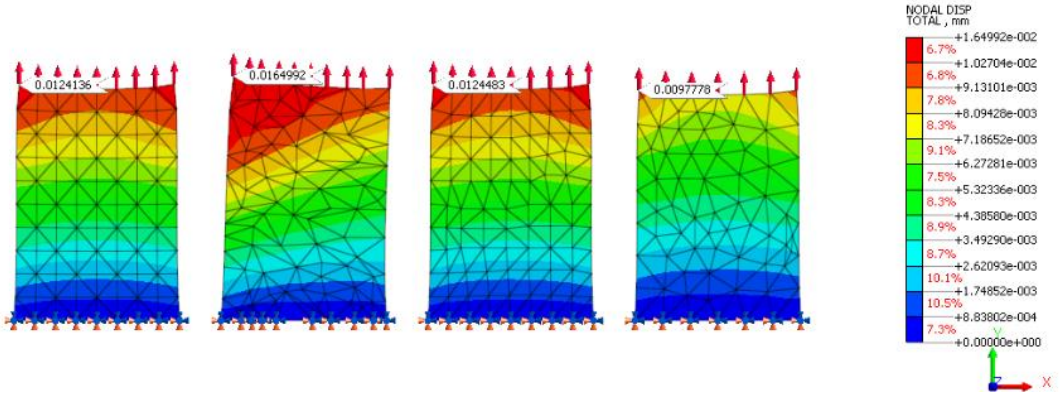


Figure 2 Effect of the mesh distortion on the results (Image Source: https://feaforall.com/wrong-mesh-impact-results/).

Therefore, to overcome these limitations presented by mesh-based methods, researchers developed particle-based methods, which offer a mesh-free representation of the domain. At the heart of particle-based methods lies the concept of discretizing a system into a collection of particles, each representing a specific portion of the material or substance being simulated (Belytschko 1996). These particles interact with each other and their environment according to prescribed rules, often derived from fundamental principles such as conservation of mass, momentum, and energy.

Due to the fact that, between the particles there is no nodal connection, particle-based methods are particularly well-suited for problems with free surfaces, fragmentations, large deformations, and complex fluid motion, where traditional mesh-based methods struggle (Vesenjak 2007) (Yagawa 1999).

The most adopted particle-based methods in industry today are:

- Smoothed Particle Hydrodynamics (SPH): This study focuses on using this mesh-free method to develop the solver (J. Monaghan 1977).

- Discrete Element Method (DEM): A numerical method for computing the motion and effect of a large number of small particles. It considers the individual interactions between particles, often using Newton's laws of motion (Cundall 1979).
- Molecular Dynamics (MD): A simulation method for studying the physical movements of atoms and molecules. It uses classical mechanics to model the trajectories of particles over time based on interatomic potentials (Allen 1989).
- Particle-In-Cell (PIC): A hybrid method combining particles and a computational grid. Particles move through the grid, and their properties are interpolated to the grid points to solve field equations, then interpolated back to update particle properties (Birdsall 1991).
- Lattice Boltzmann Method (LBM): A method for simulating fluid flows using a discrete lattice grid. It models the fluid with particle distributions at each lattice site, evolving based on collision and streaming operations (Succi 2001).

1.2 Objectives

The objective of this study is to develop a Smoothed Particle Hydrodynamics (SPH) solver, employing two distinct approaches: the rate density method and the summation density method, for analyzing the motion of incompressible fluids. The investigation comprises the following tasks:

- Selection of pre-processing methodology, involving the choice of software for pre-processing to design the model. Additionally, defining how the solver extracts initial parameters from the input file generated by the pre-processing software.
- Determination of post-processing methodology, particularly selecting the post-processing software and specifying the output file format required for visualizing results during the post-processing phase.
- Implementation of the SPH solver using MATLAB.
- Performance comparison between the Rate Density solver and the Summation Density solver.
- Validation of the solvers through benchmark examples sourced from existing literature.
- Solving a practical problem using the developed SPH solver, involving a comparison of baffle designs to reduce sloshing in an automobile fuel tank.

2 LITERATURE REVIEW

2.1 Introduction to the SPH

Smoothed Particle Hydrodynamics (SPH) is a foundational computational technique primarily employed for simulating fluid dynamics and various processes within continuous media. Originating in the early 1970s, the pioneering works of Gingold and Monaghan established the groundwork for SPH's evolution. In 1977, Gingold and Monaghan formally introduced SPH as a numerical approach specifically designed for modeling astrophysical phenomena (J. Monaghan 1977).

In (M. Liu, Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments 2010) the SPH is delineated as: "a meshfree particle method based on Lagrangian formulation that has been widely applied to different areas in engineering and science". According to this definition, three important features of the SPH method can be outlined:

- Meshfree, because it does not require a mesh to discretize the problem. Even though to perform some neighbor particle algorithms a mesh is required (L. G. Liu 2003).
- Lagrangian, because the value of the variables is strictly linked to the movement of the particles. Therefore, being the particle in motion in the domain, it is possible to investigate the evolution of the variable, which are not linked to a fixed point (Eulerian approach) (Goffin 2013).
- Particle method because the model is discretized into a set of particles.

Indeed, the state of a system is encoded within a collection of particles. Each particle bears pertinent information relevant to the problem at hand. For instance, in hydraulics, key data includes position, velocity, and density. Conversely, in solid mechanics, particles also retain records of stresses and strains.

Throughout the subsequent decades, SPH garnered momentum as researchers recognized its adaptability and relevance across diverse domains encompassing fluid dynamics, astrophysics, and engineering. Its intrinsic capability to navigate complex geometries without fixed grids rendered SPH particularly appealing for simulations entailing free surfaces, fluid-solid interactions, and turbulent flows.

For these reasons nowadays the SPH method is applied to many different application fields:

- Astrophysics, as mentioned above it is its original application filed, researchers are exploiting the SPH method to investigate the galaxy formation (Marri 2002), to understand the process of collision between two stars (Lombardi 1998) but also investigate the several steps of the star formation process (Springel 2002), which is displayed in Figure 3, and in many other studies.

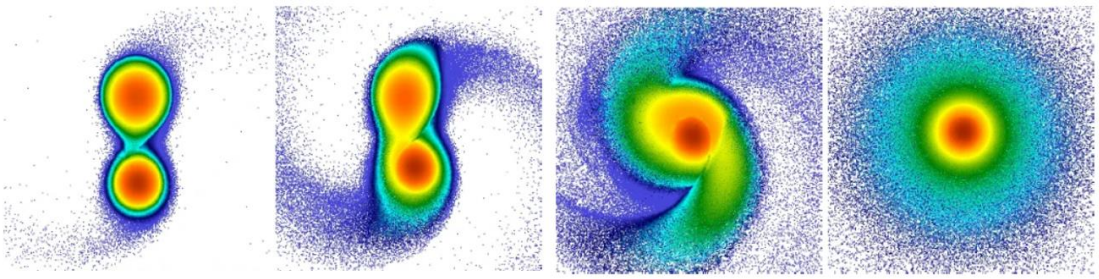


Figure 3 White Dwarf-White Dwarf merger (Diehl 2008).

- Solid Mechanics, the main advantage of SPH in this application is the possibility of dealing with larger local distortion than grid-based methods. SPH has been involved spready in the high speed impacts analysis, as done by Mehra (Mehra 2005) who analyzed the impact of a metal sphere on a metallic plate, an example is shown in Figure 4. Furthermore, it is used to study metal forming problems, where SPH has outclassed many other types of methods as the finite element method (Takamiya 2011).

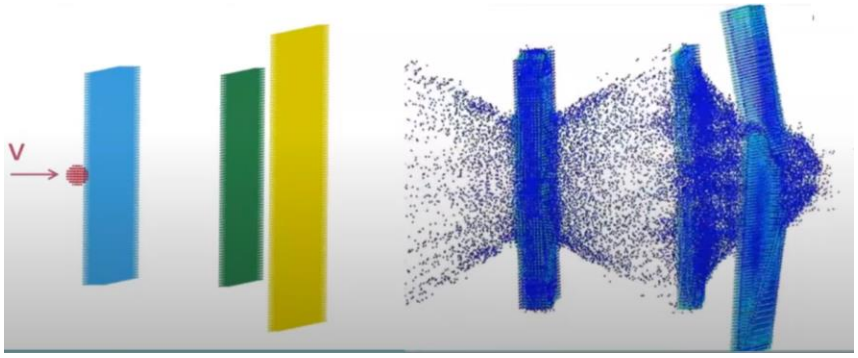


Figure 4 Sphere high speed impact on multiple walls (Image source: Youtube:DYNAmore Express: Beyond FEA - Smoothed Particle Hydrodynamics (SPH)).

- Fluid dynamics, SPH has been utilized to explore the movement of both compressible and incompressible fluids. Regarding the latter, key areas of focus include:
    1. Study of the natural disasters, such as the dam break problem, shown in Figure 5.
    2. Investigate the motion of the ocean waves.
    3. Injection of fluids.

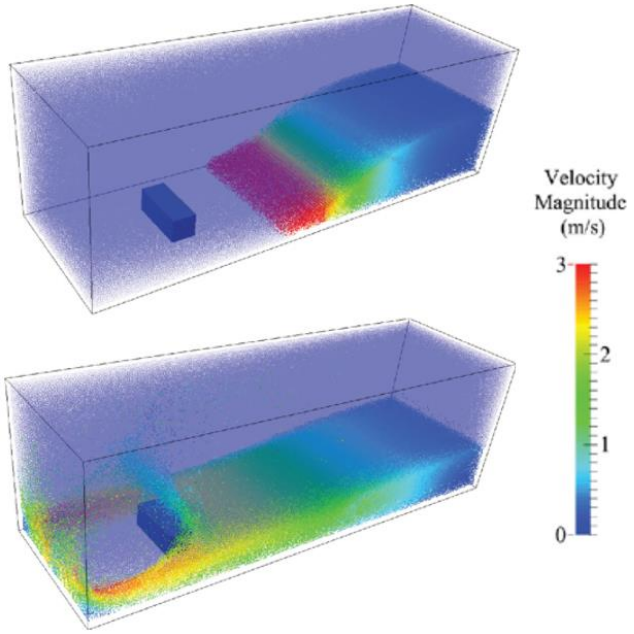


Figure 5 Dam break simulation with obstacle (Mokos 2016).

For what concerns the automotive industry SPH is applied to study the movement of the oil for bearing and gearbox lubrication (Paggi 2020) (Ji 2018), to analyze the way air and fuel mix in the combustion chamber in a compression ignition engine (Maghbouli 2015), but especially to optimize the topology of fuel tank rib to suppress fuel sloshing (S. Zheng 2021). In particular, the solver developed in this study will be applied to this problem to investigate which solution could reduce more the movement of the fuel inside of the tank (refer to "The Sloshing Problem" paragraph).

2.2 Basics of the SPH method

2.2.1 Kernel representation of a function

The SPH method is based on the idea of the integral representation of a function, that can be interpreted as: considering a generic function (*f*) any property can be expressed through its vector position $r = (x; y; z)$:

$$f(r) = \int_{\Omega} f(r')\delta(r - r') \cdot dr' \tag{2-1}$$

Where **Ω** is the volume on which the vector *r* is defined. In particular, looking at (2-1), the integral representation is based on the Dirac delta function, which is outlined as:

$$\delta(r - r') = \begin{cases} \frac{1}{2h} \; if \; r' = r \\ 0 \; if \; r' \neq r \end{cases} \tag{2-2}$$

From (2-2), it can be understood that the delta function has a value different from zero only if $r' = r$ (M. Liu, Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments 2010). Moreover, its value is influenced by the parameter *h,* which represents the support domain of the delta function, as it is shown in Figure 6.

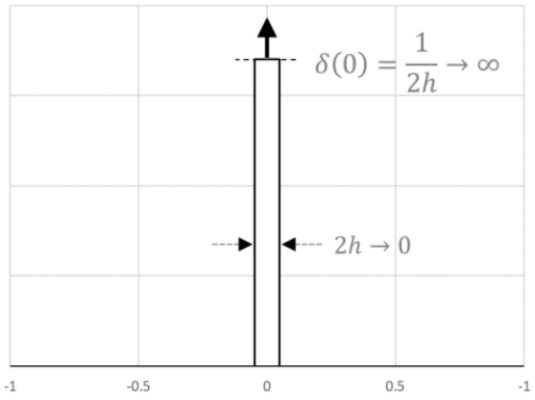


Figure 6 Dirac delta function (Cherniaev 2023).

A fundamental property of the Dirac delta function, that is the basis of the integral representation (2-1), is that the integral of the delta function on its domain, is equal to one:

$$\int_{\Omega} \delta(r - r') \cdot dr = 1 \tag{2-3}$$

From the Dirac delta function, we can derive the Kernel function. As depicted in Figure 7, adjusting the parameter h, known as the smoothing length, when related to the kernel function, it alters the shape. In particular, when $h \to 0$, the kernel function coincides with the Dirac delta function.

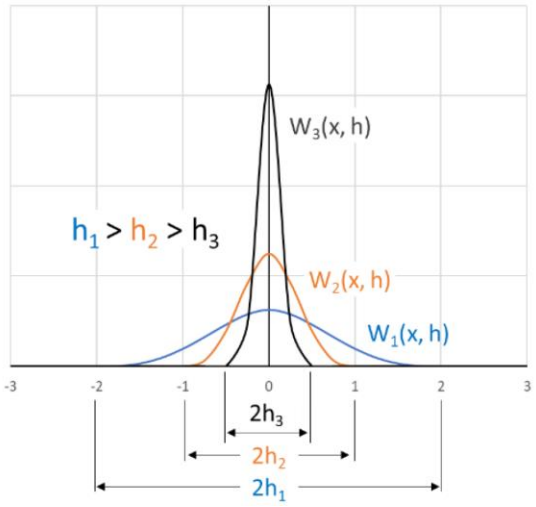


Figure 7 Kernel function variation according to the smoothing length (Cherniaev 2023).

Therefore, once the kernel function (or smoothing function) is outlined, the equation (2-1), can be written as:

$$f(r) \approx \int_{\Omega} f(r')W(r - r', h) \cdot dr' \tag{2-4}$$

Equation (2-4), is the basic equation of the SPH method. Nevertheless, it is important to underline that, exploiting the kernel function, It is no longer possible to obtain the exact value of the function *f*, but only an approximated value (Gomez-Gesteira 2010).

However, the SPH method refers to particles, thus it is necessary to shift from the continuum domain, see Figure 8, where up to now all the equations have been discussed, to the discrete domain, where the particles are defined, displayed in Figure 9. Therefore, moving to the discrete domain, equation (2-4) is written as follows:

$$f(r) \approx \sum_{j=1}^{N} f(r_j) W(r - r_j, h) \cdot \Delta V_j \tag{2-5}$$

In the discrete domain the integral is substituted by the summation term. Then, (r_j), represents the j-neighbor particle with respect to the selected one. Finally, it must be pointed out that dr' is substituted by the particle volume ΔV_j, which will be a fundamental term in the approximation of the equation of motion of the incompressible fluids through the SPH method (Cherniaev 2023).

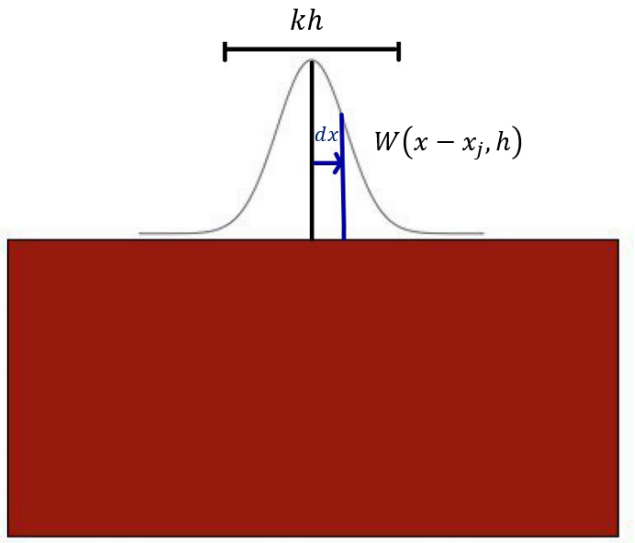


Figure 8 Model representation in the continuum domain.

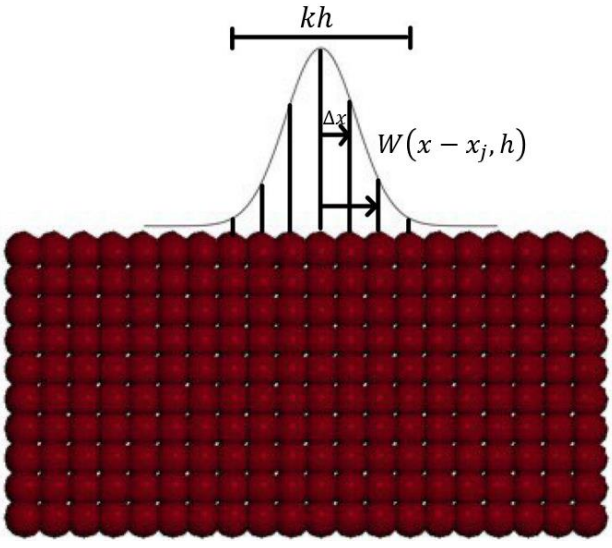


Figure 9 Model representation in the discrete domain.

Starting from equation (2-4), it is possible to outline the kernel approximation also for the gradient of a function, $\nabla f(r)$:

$$\nabla f(r) \approx \int_{\Omega} \nabla f(r') W(r - r', h) \cdot dr' \tag{2-6}$$

The term inside of the integral on the RHS can be written as (Wang 2016):

$$\nabla f(r') W(r - r', h) = \nabla [f(r') W(r - r', h)] - f(r') \nabla W(r - r', h) \tag{2-7}$$

Thus, exploiting the linearity property of the integral, equation (2-6), (2-7) can be expressed as (Wang 2016):

$$\nabla f(r) \approx \int_{\Omega} \nabla [f(r') W(r - r', h)] \cdot dr' - \int_{\Omega} f(r') \nabla W(r - r', h) \cdot dr' \tag{2-8}$$

Through the use of the divergence theorem the first integral on the RHS can be rearranged as (M. Liu, Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments 2010):

$$\nabla f(r) \approx \int_{S} f(r') W(r - r', h) \cdot \vec{\mathrm{n}} dS - \int_{\Omega} f(r') \nabla W(r - r', h) \cdot dr' \tag{2-9}$$

Where $\vec{\mathrm{n}}$ is the normal vector with respect to the surface S.

Given that the smoothing function W typically possesses compact support, it follows that its value on the integral's surface, is null within the framework of SPH. Consequently, the surface integral on the right-hand side similarly evaluates to zero (M. Liu, Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments 2010) (Wang 2016):

$$\nabla f(r) \approx -\int_{\Omega} f(r')\nabla W(r-r',h)\cdot dr' \tag{2-10}$$

The Equation (2-10) has a very important meaning in the Smoothed Particle Hydrodynamics (SPH) method because it highlights the fact that the gradient of a function can be expressed solely through the gradient of the kernel. Indeed, this becomes immediately evident when comparing it with equation (2-4. To get the higher order of kernel derivatives the same procedure can be applied (M. Liu, Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments 2010).

Finally, moving to the discrete domain, Equation (2-10) can be written as:

$$f(r) \approx \sum_{j=1}^{N} f(r_j)\nabla W(r-r_j,h)\cdot \Delta V_j \tag{2-11}$$

2.2.2 Properties of the kernel function

The kernel function has three fundamental properties (Yang 2013):

1. Normalization condition:

$$\int_{\Omega} f(r')W(r-r',h)\cdot dr' = 1 \tag{2-12}$$

   This condition is also known as the "unity condition" because the integration of the kernel function on its support domain defines a unit value.

2. Limit condition:

$$\lim_{h\to 0} W(r-r',h) = \delta(r-r') \tag{2-13}$$

   This condition underlines the strict correlation between the kernel function and the Dirac delta function, as mentioned above.

3. Compactness condition:

$$W(r - r', h) = 0 \quad if \quad r - r' > k \cdot h \tag{2-14}$$

As displayed in Figure 6, the smoothing function, can be multiplied by a factor h to enlarge the support domain of the kernel. However, if a point is outside of the domain, its contribution to the kernel approximation is equal to zero. This is a very important condition, on which is based the concept of the neighbor particle search algorithm that is explained in the "Neighbor particles search algorithms." section.

2.2.3 Kernel functions

In this study several kernel functions have been considered to simulate the validation examples. Here is reported the analysis of each kernel function that has been taken into account:

1. Bell kernel (Lucy 1977):

$$W(r - r', h) = \frac{105}{16\pi h^3}\begin{cases}(1 - 6R^2 + 8R^3 - 3R^4) & if\ R \leq 1 \\ 0 & if\ R > 1\end{cases} \tag{2-15}$$

Where $R = \frac{r - r'}{h}$

The first characteristic of this kernel is its supporting domain. Considering the parameter R, it appears evident that the domain is equal to $1h$, this has an important consequence, because it defines a low number of neighbor particles, making the kernel's computation cost quite low (if compared to other kernels with a larger support domain). Despite its compact support, the Bell Kernel can be more computationally demanding than simpler kernels. The increased computational cost arises from evaluating the more complex mathematical form of the kernel and its derivatives. Moreover, looking at Figure 10, the kernel tends to zero already when $R = 0.8$. This means that if we do not enlarge the support domain, usually $h = 1.2\,\Delta x$ (where Δx is the initial spacing between the particles), the particles will not generate a sufficient repulsion force in their rest position, leading to an oscillating behavior.

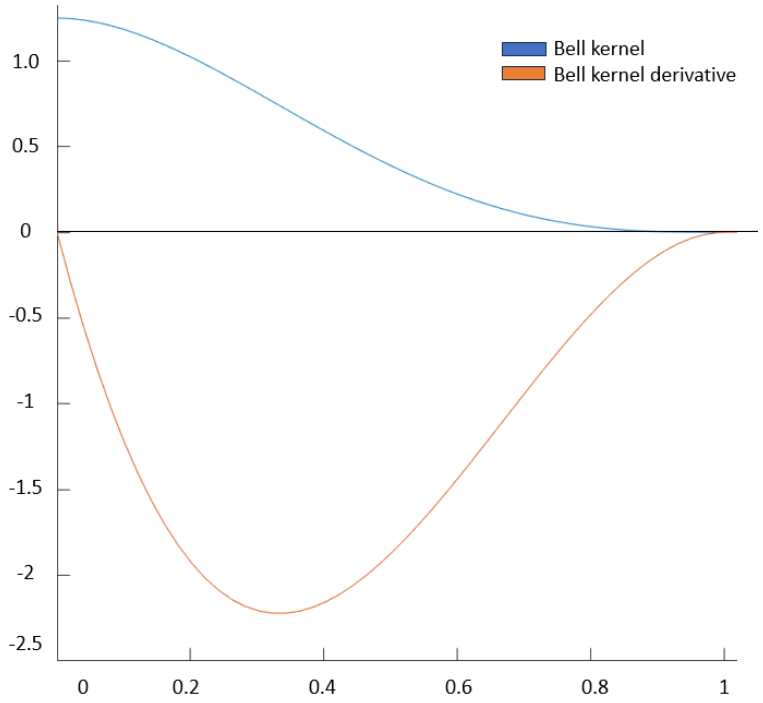


Figure 10 Bell kernel and its derivative.

2. Cubic Spline kernel (Cherniaev 2023):

$$W(r-r',h)=\begin{cases}\frac{2}{3}-R^2+\frac{1}{2}R^3 & when\ 0\le R<1\\ \frac{1}{6}(2-R)^3 & when\ 1\le R<2\\ 0 & when\ R>2\end{cases} \quad (2\text{-}16)$$

Where $R=\frac{r-r'}{h}$

The cubic spline kernel has an enlarged support domain, which is twice the one of the Bell kernel. Therefore, the computational cost is higher because the number of neighbor particles that will be identified is increased, which is the drawback of this kernel. A further consideration, related to its mathematical definition, regards that the kernel is not the expression of a unique function, but it is made up by two different functions, according to the interval that we are referring through R. Nevertheless, looking at Figure 11, the Cubic Spline kernel is a very smooth function, especially considering its derivative. These characteristics make it well-suited for accurately capturing fluid behavior, especially in regions of high gradient or density variation.

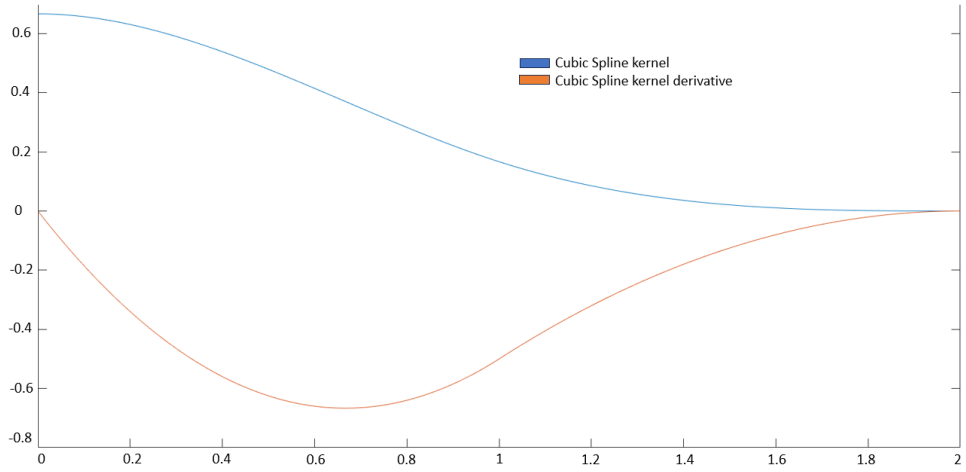


Figure 11 Cubic spline kernel and its derivative.

3. Gaussian kernel (J. Monaghan 1977):

$$W(r - r', h) = (\frac{1}{\pi h^2})^{\frac{3}{2}} \cdot e^{\frac{-r^2}{h^2}} \qquad (2\text{-}17)$$

Where $R = \frac{r - r\prime}{h}$

A narrow look at equation 2-17 reveals a key feature of the Gaussian kernel: its theoretical infinite support domain. To achieve a value of zero, R must tend towards infinity. However, in practical applications, a finite value for the support domain must be set within the solver. This property makes the Gaussian kernel particularly suitable for scenarios where particle distribution in space is uneven. Additionally, as depicted in Figure 12, the kernel exhibits a smooth shape, especially with a similar derivative profile, resembling that of the Cubic Spline. Moreover, focusing on the right boundary of the graph, where the maximum value of R is set to 2 allowing the comparison with the Cubic Spline, it becomes evident that the kernel is truncated. However, it narrow approaches zero ensuring a high approximation efficacy.

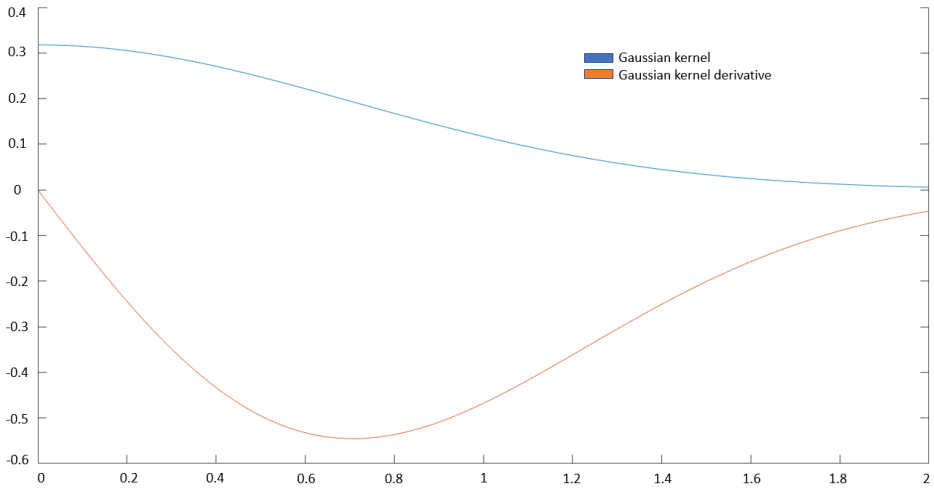


Figure 12 Gaussian kernel and its derivative.

4. Spiky kernel (Desbrun 1996):

$$W(r-r',h)=\frac{15}{\pi h^6}\begin{cases}(h-R)^3 & if\ R\le 1\\ 0 & if\ R>1\end{cases} \qquad (2\text{-}18)$$

Where $R=\frac{r-r'}{h}$

Looking at Figure 13, in particular on the derivative shape, among all the kernel functions, the Spiky kernel stands out as the sole one lacking a derivative with a bell shape. Notably, as the distance between two particles diminishes, the gradient of this kernel reaches its maximum value (an attribute not observed in the preceding kernels). This behavior makes the Spiky kernel particularly useful for generating substantial repulsive forces between particles, ensuring they do not penetrate each other, as the derivative remains nonzero when particles are in close proximity. However, the function's range yields notably high values, rendering the kernel more aggressive. Consequently, there's an increased likelihood of particles moving farther apart from each other, causing severe oscillations in the particle motion.

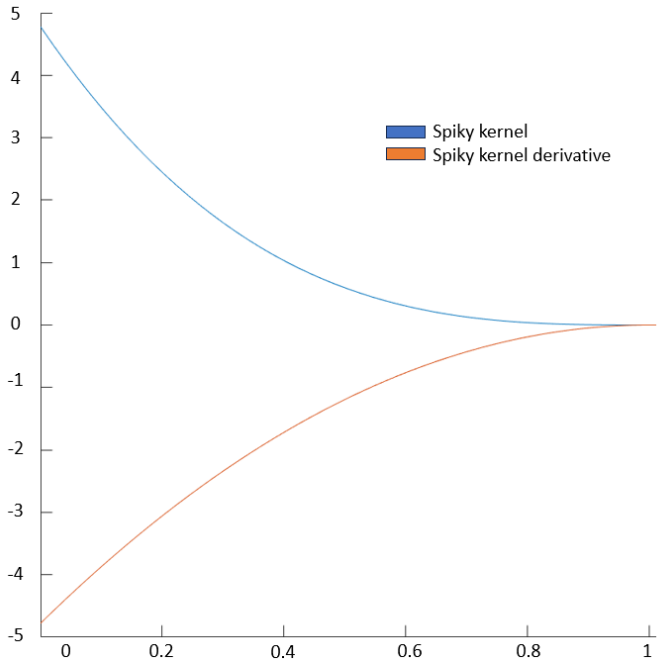


Figure 13 Spiky kernel and its derivative.

5. Poly6 kernel (Müller 2003):

$$W(r-r',h)=\frac{315}{64\pi h^9}\begin{cases}(h^2-R^2)^3 & if\ R\le 1\\ 0 & if\ R>1\end{cases} \qquad (2\text{-}19)$$

Where $R=\frac{r-r\prime}{h}$

Although the Poly6 kernel, as illustrated in Figure 14, differs mathematically from the Bell kernel, its shape closely resembles the kernel devised by Lucy in 1997. However, two main distinctions arise, particularly concerning its derivative. Firstly, when examining the range of values attained by the derivative of the kernel function, they surpass the absolute values found in the Bell kernel, resulting in a more powerful repulsive force between particles. Secondly, the derivative shape resembles a parabola centered near the middle of the support domain, thus facilitating the definition of a more stable and smooth interaction between neighboring particles.

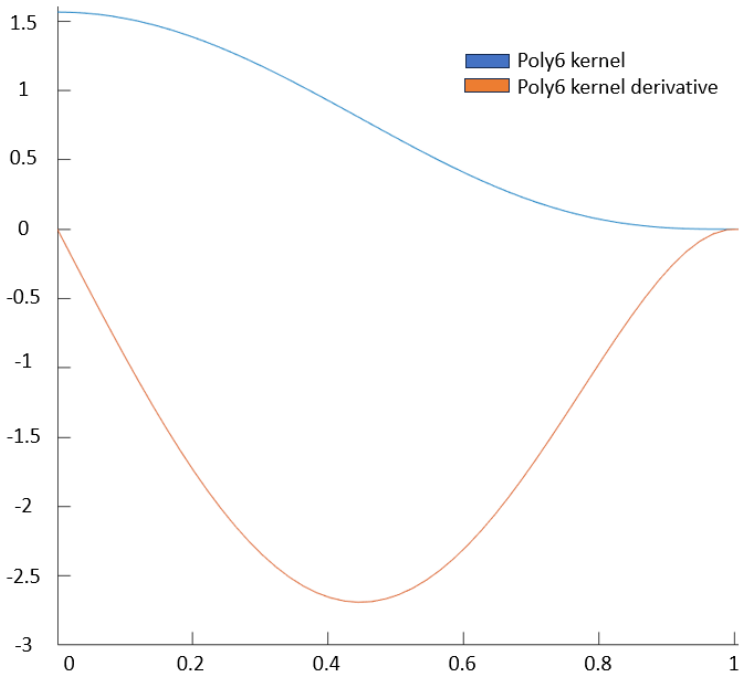


Figure 14 Poly6 kernel and its derivative.

2.3 Motion of Fluids

2.3.1 Fluid definition and its properties

A fluid is “any material that cannot sustain a tangential, or shearing, force when at rest and that undergoes a continuous change in shape when subjected to such a stress” (Encyclopedia Britannica 2021). Fluids can be categorized into three distinct groups:

- Liquids, such as water, are characterized by a relatively high density and have a distinct volume, meaning that changes in pressure or temperature have minimal impact on their volume.
- Gases, on the other hand, lack a specific volume and can be compressed or expanded indefinitely when subjected to varying pressures. Both liquids and gases take on the shape of the container that holds them.
- Plasma represents one of the four fundamental states of matter, marked by the presence of a substantial number of charged particles in the form of ions or electrons.

Having defined these categories, it's crucial to highlight two essential properties of fluids, which describe their behavior in both static and dynamic conditions. In the case of a static fluid, the sole property that defines its behavior is its density, which is the ratio of the fluid's mass to its volume:

$$\rho = \lim_{\Delta V \to 0} \frac{\Delta m}{\Delta V} \; [kg/m^3] \qquad (2\text{-}20)$$

The limit $\Delta V \to 0$ must be considered in the hypothesis of the continuum.

It is an intensive property, so it does not depend on the amount of matter or the dimension of the object, but only on the nature and the condition at which it is.

The values of water density at the variation of the temperature are reported in Table 1.

Table 1 Water density (Kestin 1978).

Temperature [C]	Density [kg / m^3]
10	999.7281
15	999.1286
20	998.2336
25	997.0751
30	995.6783
35	994.0635
40	992.2473
45	990.24
50	988.07
55	985.73
60	983.24
65	980.59
70	977.81

The other fundamental property is the viscosity, which measures the resistance of a fluid to induced flow by the action of a shear stress (Hack 2018). It is a macroscopic property of fluids that is direct consequence of the motion of molecules in the fluid.

$$Viscosity = \mu\,[Pa \cdot s] \tag{2-21}$$

Fluids tend to exhibit greater viscosity when their intermolecular forces are stronger. This is notably the case in liquids compared to gases, or when the temperature of a given fluid rises. When a fluid possesses higher viscosity, it requires a greater force to induce the same flow within it. In the context of moving fluids, velocity gradients are a common feature, leading to alterations in the volume of the fluid in motion. Viscosity serves as the property that links such deformations, represented by velocity gradients, to the transfer of momentum within the fluid, as displayed in Figure 15.

Considering a fluid that is flowing, it is possible to define its:

- Shear stress: $\tau = \frac{F}{A}$, where F is the force applied to the surface A.
- Shear rate: $\Gamma_s = \frac{U}{h}$, where U is the difference between the top and bottom velocity of a fluid flowing into a two plates and h is the distance between the two plates. With $h << A$.

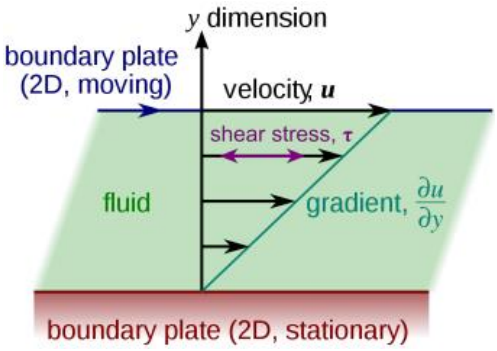


Figure 15 Laminar shear in a fluid (Image Source: https://en.wikipedia.org/wiki/File:Laminar_shear.svg).

In Table 2, are reported the values of the water viscosity at the variation of the temperature.

Table 2 Water viscosity (Korson 1968).

Temperature [C]	Viscosity [mPa s]
0	1.7916
5	1.5192
10	1.3069
15	1.1382
20	1.002
25	0.8903
30	0.7975
35	0.7195
40	0.6532
45	0.5963
50	0.5471
55	0.5042
60	0.4666
65	0.4334
70	0.4039

Finally, a further definition of a fluid consists in:

- Newtonian is a fluid whose viscosity remains constant regardless of the applied shear rate. This means that its flow behavior is linear, and the relationship between shear stress and shear rate is directly proportional (Cherniaev 2023). Water, air, and most simple liquids exhibit Newtonian behavior. For Newtonian fluids, the viscosity does not change with varying flow conditions:

$$\frac{d\tau}{d\Gamma_s} = constant \implies \tau = \mu \cdot \Gamma_s \implies \mu = \frac{\tau}{\Gamma_s} \qquad (2\text{-}22)$$

- Non-Newtonian has a viscosity that changes with the applied shear rate. This means its flow behavior is non-linear. Depending on the specific type of non-Newtonian fluid, the viscosity may

either increase (shear-thickening) or decrease (shear-thinning) with an increase in shear rate (Chhabra 2010):

$$\frac{d\tau}{\mathrm{d}\Gamma_s} \neq constant \Longrightarrow \tau = \tau_0 + \mu(\Gamma_s) \cdot \Gamma_s \qquad (2\text{-}23)$$

2.3.2 Governing equations of fluid dynamics

The governing equations of fluid dynamics are:

- Continuity Equation.
- Momentum Equation.
- Energy Equation.

The SPH solver developed in this study relies on the first two fundamental equations of fluid dynamics. Therefore, this section presents a review of the Continuity and the Momentum equations.

Continuity Equation:

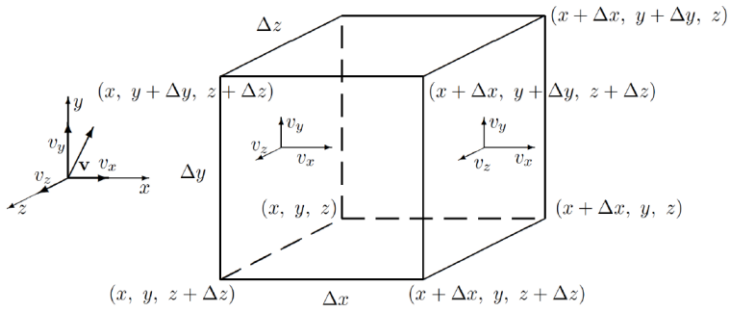


Figure 16 Control volume (Marchioli 2020).

The principle of mass conservation states that, considering the control volume presented in Figure 16, the rate of change of mass within the control volume is equal to the net mass flow rate across the control surfaces (Childs 2011) (Gao 2016):

$$\frac{\partial m}{\partial t} = \dot{m}_{in} - \dot{m}_{out} \qquad (2\text{-}24)$$

Starting from Equation 2-24, it is possible to obtain in few steps (Computational Fluid Dynamics 2009) the following expression of the mass conservation:

$$\frac{\partial \rho}{\partial t} + \overrightarrow{\rho \nabla} \cdot (\vec{v}) = 0 \tag{2-25}$$

But being the fluid incompressible, the density variation over the time is equal to zero (John R. Fanchi 2002). Thus, Equation 2-25 can be simplified getting as final step:

$$\vec{\nabla} \cdot \vec{v} = 0 \tag{2-26}$$

Equation 2-26 is called Continuity Equation, and it ensures that the necessary and sufficient condition for the conservation of mass in an incompressible fluid is that the divergence of the fluid velocity vector is zero.

Momentum Equation:

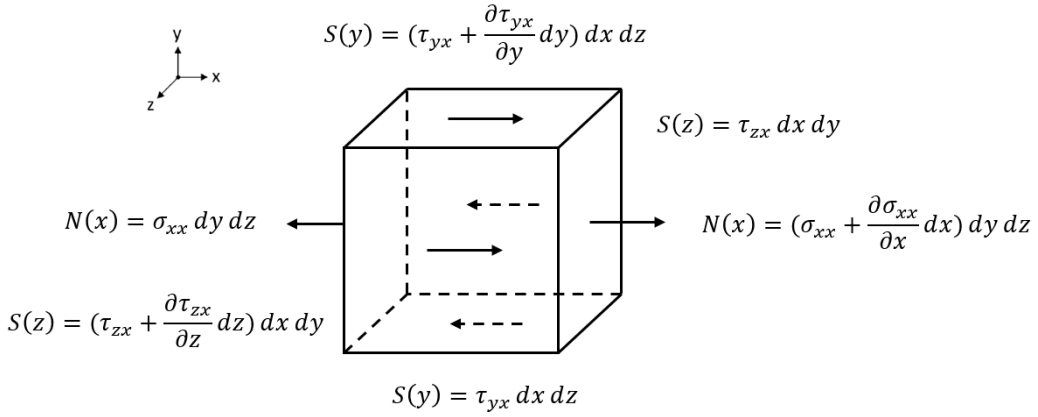


Figure 17 Stresses applied to the control volume in x direction.

Starting from the stresses applied to the control volume in the x direction, displayed in Figure 17, it is possible to write the 2nd Newton's Law as (Goffin 2013) (Cherniaev 2023):

$$ma_x = mg_x + [\sigma_{xx} + \frac{\partial \sigma_{xx}}{\partial x} dx] dydz - \sigma_{xx} dydz + [\tau_{yx} + \frac{\partial \tau_{yx}}{\partial y} dy] dxdz - \tau_{yx} dxdz + [\tau_{zx} + \frac{\partial \tau_{zx}}{\partial z} dz] dxdy - \tau_{zx} dxdy \tag{2-27}$$

From Equation 2-27, the terms with the opposite sign can be removed and noting that the mass can be rewritten as $m = \rho \cdot dxdydz$, we get:

$$\rho a_x = \rho g_x + \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} \tag{2-28}$$

Applying Equation 2-28 to a Newtonian fluid, such as water, it is possible to write the shear stress in yx directions as (Deissler 1976) (Cherniaev 2023):

$$\tau_{yx} = \mu \frac{\partial \gamma_{yx}}{\partial t} = \mu\left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right) \tag{2-29}$$

In the same way the shear stress in y and z directions can be obtained.

For what concerns the normal stress, with few steps (Cherniaev 2023), we get:

$$\sigma_{xx} = \frac{\sigma_{xx} + \sigma_{yy} + \sigma_{zz}}{3} + 2\mu \frac{\partial v_x}{\partial x} - \frac{2\mu}{3}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}\right) \tag{2-30}$$

The pressure in a moving fluid is defined as the negative of the mean stress (Coleman 2010) (Cherniaev 2023):

$$p = \sum_{j=1}^{n} \frac{\sigma_{jj}}{n} \tag{2-31}$$

Thus, through the Continuity Equation (2-26) and Equation 2-31, it is possible to rewrite Equation 2-30 as:

$$\sigma_{xx} = -p + 2\mu \frac{\partial v_x}{\partial x} \tag{2-32}$$

The same procedure can be applied to normal stress in y and z directions.

Therefore, through Equation 2-29 and Equation 2-32 the 2nd Newton's Law (2-28) can be rewritten as:

$$\rho a_x = \rho g_x - \frac{\partial p}{\partial x} + \mu\left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2} + \frac{\partial^2 v_x}{\partial z^2}\right) + \mu \frac{\partial}{\partial x}\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}\right) \tag{2-33}$$

Finally, through the Continuity Equation (2-26) the last term on the RHS can be removed. By applying the same procedure used in the x-direction to the other two coordinates, the Navier-Stokes Equation is obtained:

$$a = g - \frac{\nabla p}{\rho} + \frac{\mu}{\rho}\nabla^2 v \qquad (2\text{-}34)$$

On the RHS the Navier-Stokes Equation (2-34) is made up of three different components (Bistafa 2018):

- g, is the gravitational acceleration, which clearly is a constant term.
- $\frac{\nabla p}{\rho}$, represents the pressure gradient, which underlines that the variation in pressure along a given direction is indicated by the minus sign, which reflects the fact that, for example, in a pipe, the pressure consistently decreases as the fluid moves forward.
- $\frac{\mu}{\rho}\nabla^2 v$, is the viscous term, which accounts for internal friction within the fluid, acting to resist the flow and smooth out velocity differences.

In conclusion, it is important to note that an analytical solution to the Navier-Stokes equation has not yet been achieved. However, by applying a sufficient number of boundary conditions, for specific applications a numerical solution can be obtained (Schneiderbauer 2014).

2.4 SPH applied to incompressible fluids.

In this section the governing equations of motion of the incompressible fluids (refer to "Motion of Fluids" section) will be approximated through the SPH method (refer to " Basics of the SPH method" section).

2.4.1 Mass Conservation

In the SPH method the density can be expressed according two different approaches. In the first SPH solver developed by Monaghan (J. Monaghan 1977) the mass conservation was based on the summation density approach. Starting from Equation (2-5 and substituting $f(x)$ with the density of the particle ρ, we get:

$$\rho \approx \sum_{j=1}^{n} m_j W_{ij} \qquad (2\text{-}35)$$

Equation 2-35 is the basic equation of the SPH method which defines the density of the particle i, by summing the mass of the neighboring particles multiplied by the kernel. However, this equation can lead to some issues in the calculation of the density when a free surface problem is involved. By looking at Figure 18, it is possible to observe that all the particles that are close to the boundaries of the recipient and to the free surface, are surrounded by just few neighboring particles, if compared to the one in the middle of the recipient.

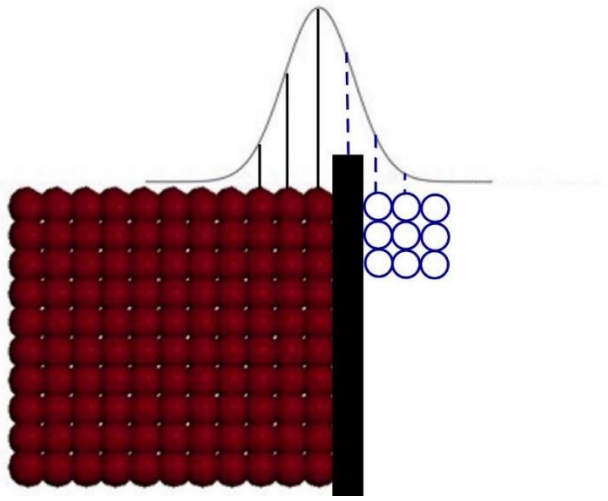


Figure 18 Boundary deficiency.

Therefore, the density calculation is deeply affected by this issue, which is called "boundary deficiency", because the presence of a low number of neighboring particles leads to a very rough approximation of the density.

A possible solution to this problem is the implementation of a different equation to calculate the density, where each contribution is normalized by the SPH summation of the smoothing function itself (Cherniaev 2023) (Monaco 2014):

$$\rho \approx \frac{\sum_{j=1}^{n} m_j W_{ij}}{\sum_{j=1}^{n} \frac{m_j}{\rho_j} W_{ij}} \tag{2-36}$$

To overcome the issue of the boundary deficiency it is possible to change the mathematical approach to calculate the density. Monaghan (J. Monaghan 1994) computed the density starting from the continuity equation (2-26), in this way the density of the particle i will be calculated exploiting the relative velocity between the two particles and the gradient of the kernel function:

$$\rho_i \approx \sum_{j=1}^{n} m_j (v_j - v_i) \nabla W_{ij} \tag{2-37}$$

However, the drawback of the rate density approach is that it does not automatically ensure the mass conservation of the model, because the formulation of the density starts form the continuity equation (J. J. Monaghan, Smoothed Particle Hydrodynamics 1992).

2.4.2 Momentum conservation

Considering the Navier-Stokes equation (2-34), the first term which represents the gravity acceleration, clearly it is not approximated. For what concerns the second term, the pressure gradient, it is necessary to start from the quotient rule for calculation of spatial derivative of a ratio of two functions (Cherniaev 2023):

$$\nabla\left(\frac{p_i}{\rho_i}\right) = \frac{\rho_i \nabla p_i - \nabla \rho_i p_i}{\rho_i^2} \tag{2-38}$$

From Equation 2-38, it is possible to explicit the pressure gradient:

$$\frac{\nabla p_i}{\rho_i} = \nabla\left(\frac{p_i}{\rho_i}\right) + \frac{\nabla \rho_i p_i}{\rho_i^2} \tag{2-39}$$

Equation 2-39 represents the pressure gradient, thus the SPH approximation equation 2-11 can be applied to the two terms on the RHS, by substituting *f(x)* with $\frac{p_i}{\rho_i}$ for the first and for the second ρ_i:

$$\nabla\left(\frac{p_i}{\rho_i}\right) \approx \sum_{j=1}^{n} m_j \frac{p_j}{\rho_j^2} \nabla W_{ij} \tag{2-40}$$

$$\nabla \rho_i \approx \sum_{j=1}^{n} m_j \nabla W_{ij} \tag{2-41}$$

Therefore, substituting Equation 2-40 and Equation 2-41 with the two terms on the RHS of Equation 2-39 the SPH approximation of the pressure gradient is obtained (J. J. Monaghan, Smoothed Particle Hydrodynamics 1992):

$$\frac{\nabla p_i}{\rho_i} \approx \sum_{j=1}^{n} m_j \left(\frac{p_j}{\rho_j^2} + \frac{p_i}{\rho_i^2}\right) \nabla W_{ij} \tag{2-42}$$

The second term to approximate is the viscous term of the Navier-Stokes equation (2-34). Starting from the product rule for calculation of spatial derivative of a product of two functions (Cherniaev 2023):

$$\nabla \cdot (\rho_i \cdot v_i) = \nabla \rho_i \cdot v_i + \nabla v_i \cdot \rho_i \tag{2-43}$$

From Equation 2-43 the velocity gradient can be explicated:

$$\nabla v_i = \frac{1}{\rho_i} [\nabla \cdot (\rho_i \cdot v_i) - \nabla \rho_i \cdot v_i] \tag{2-44}$$

The second term on the RHS of Equation 2-44 can be approximated as Equation 2-41, while the first can be approximated substituting *f(x)* with $\nabla \cdot (\rho_i \cdot v_i)$ in Equation 2-11:

$$\nabla \cdot (\rho_i \cdot v_i) \approx \sum_{j=1}^{n} m_j \cdot v_j \cdot \nabla W_{ij} \tag{2-45}$$

Thus, substituting Equation 2-41 and Equation 2-45 to the RHS of Equation 2-44 we get:

$$\nabla v_i \approx \sum_{j=1}^{n} \frac{m_j}{\rho_j} \cdot (v_j - v_i) \cdot \nabla W_{ij} \tag{2-46}$$

Finally, noting that the Laplacian can be written as $\nabla \cdot (\nabla \cdot v_i)$ from Equation 2-46, the SPH approximation of the viscous term is obtained (J. Chen 2009):

$$\nabla^2 v_i \approx \sum_{j=1}^{n} \frac{m_j}{\rho_j} \cdot (v_j - v_i) \cdot \nabla^2 W_{ij} \tag{2-47}$$

Once the pressure gradient and the viscous term have been approximated, it is possible to substitute Equation 2-42 and Equation 2-47 to the RHS of the Navier-Stokes Equation (2-34), defining:

$$a_i = g - \sum_{j=1}^{n} m_j \left(\frac{p_j}{\rho_j^2} + \frac{p_i}{\rho_i^2} \right) \nabla W_{ij} + \frac{\mu}{\rho_i} \sum_{j=1}^{n} \frac{m_j}{\rho_j} \cdot (v_j - v_i) \cdot \nabla^2 W_{ij} \tag{2-48}$$

2.4.3 Pressure evaluation

To evaluate the pressure of the water particles the Equation of State is used, because of the direct relationship between the density and the pressure itself (Batchelor 1967) (Gomez-Gesteira 2010):

$$p = B \left(\left(\frac{\rho}{\rho_0} \right)^{\gamma} - 1 \right) \tag{2-49}$$

Where: ρ is the density of the particle, $\rho_0 = 1000 \text{ kg}/m^3$ is the reference density of water, $\gamma = 7$ which is the adiabatic coefficient and the parameter $B = \frac{c^2}{\rho \gamma}$.

In certain scenarios, hydrostatic pressure is present in the fluid at the start of a simulation, such as during a dam break event. This necessitates initializing the hydrostatic pressure at the outset. However, pressure is not a direct attribute of a particle; rather, it is derived from the density, which is a property of the particle. Consequently, the initial pressure distribution in the fluid is established based on the initial densities of the particles (Goffin 2013) (J. Monaghan 1994). The hydrostatic pressure is defined as:

$$p = \rho_0 \, g \, H \tag{2-50}$$

Where: $\rho_0 = 1000 \text{ kg}/m^3$ is the reference density of water, g is the gravitational acceleration and H is the depth of the particle.

By setting equation 2-49 and equation 2-50 equal and expressing the density (ρ), it is possible to derive the equation needed to initialize the density according to the hydrostatic pressure (J. Monaghan 1994):

$$\rho = \rho_0 \left(1 + \frac{\rho_0 \, g \, H}{B}\right)^{\frac{1}{\gamma}} \tag{2-51}$$

2.4.4 Boundary Conditions

In SPH, there is no universal guideline for defining boundaries because the choice depends on the specific requirements and constraints of the simulation. Researchers often experiment with different methods and adapt their approaches based on the needs of their particular applications. Thus, the literature presents many different techniques, with others still in development.

A primary technique involves treating the impact of particles against the boundary as an elastic collision. This means that when particles approach the boundary, they will reverse their motion. Additionally, their velocity will be multiplied by a damping coefficient, which is lower than one, to reduce the velocity after the impact (Bindel 2011).

This solution allows for simple modeling of basic geometries, such as a vertical wall. However, when defining a complex boundary or obstacle (refer to "Dam break with a ramp." section), it is more effective to implement boundaries and objects using particles:

- Repulsive particles (J. Monaghan 1994). This solution locates fixed particles which exert central forces on the fluid particles, defined through the following expression:

$$f(r) = \begin{cases} \left(\left(\frac{r_0}{r}\right)^{p_1} - \left(\frac{r_0}{r}\right)^{p_2}\right)\frac{r}{\|r\|} & if \; r < r_0 \\ 0 & if \; r \geq r_0 \end{cases} \tag{2-52}$$

  From the mathematical definition of Equation 2-52 it appears evident that the forces are pure repulsive according to the parameter r_0, which defines the distance within the particle exert the force. For what concerns the other parameters Monaghan proved that D can be chosen equal to $5 \; g \; H$ or $10 \; g \; H$, where H is the depth of the particles, because both choices retrieve similar results. Moving two the last two parameters p_1 and p_2, also in this case two different choices have been investigated and finally it has been proved that the parameters must satisfy the following condition: $p_1 > p_2$.

- Boundary particles. These particles share the same properties as the fluid particles, except that their momentum is not solved. Therefore, the particles can either remain in their fixed positions or follow a specific motion rule. This approach is particularly useful for modeling moving boundaries. A significant advantage is that these particles can be computed in the same loop as the fluid particles, simplifying implementation and saving computational time.

  Crespo (Crespo 2007) proposed that the continuity equation should be solved for boundary particles. Meanwhile, Lobovsky (Lobovsky 2007) suggested that maintaining a constant density throughout the simulation would improve the motion of fluid particles near the boundary. Goffin (Goffin 2013) highlighted that implementing Crespo's solution caused boundary particles to generate an additional viscous effect, resulting in several fluid particles remaining attached to the boundaries. Additionally, a thin separation layer could form due to the interaction between fluid and boundary particles (Goffin 2013).
- Ghost particles (Randles 1996). When a fluid particle approaches the boundary, an identical particle with the same density and opposite velocity is generated just beyond the boundary. This interaction ensures that the fluid particle remains within the boundaries. However, the primary disadvantage of this technique is that the number of particles varies throughout the simulation, complicating the code implementation. To overcome this issue (Marrone 2011) suggested to exploit fixed ghost particles whose values are calculated at interpolation nodes located within the fluid domain. These values are obtained using a Moving Least Squares (MLS) interpolation over the fluid particles within their range of influence. Thus, their number is fixed from the beginning of the simulation allowing an easier implementation in the code.

2.5 Explicit integration

In the fields of solid and fluid mechanics, step-by-step time integration methods are commonly employed to solve dynamic motion equations. These methods are generally categorized into two types: explicit and implicit (Gui 2014).

Implicit methods determine future values of a function by solving equations that involve both current and future states. This often requires iterative solutions, making implicit methods more computationally intensive. However, they are generally more stable and allow for larger time steps, making them well-suited for stiff equations and long-term simulations in field such as structural mechanics (M. Zheng 2017), as shown in Figure 19.

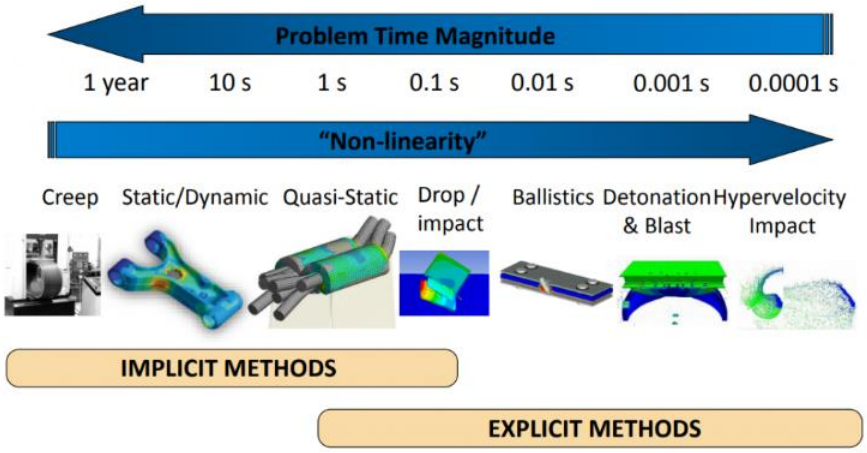


Figure 19 Implicit vs explicit integration (Image Source: mechead.com).

For this study, the explicit integration method has been utilized. This numerical technique is used to solve ordinary differential equations (ODEs) in computational simulations and modeling. It provides a straightforward approach, calculating future values of a function based on its current state and derivatives.

To gain a deeper understanding of this method, let's begin by examining the graph at Figure 20.

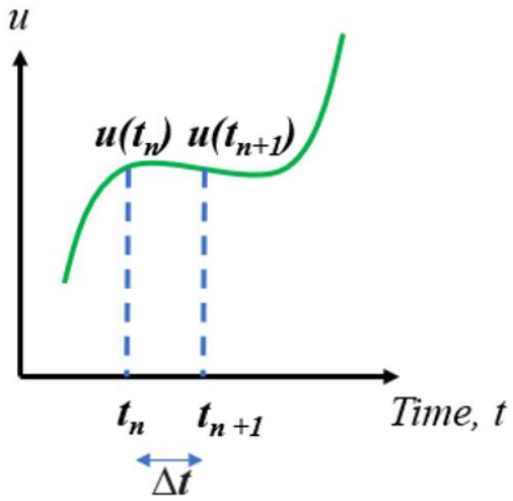


Figure 20 Explicit integration (Image Source: https://www.fidelisfea.com)

Here is represented a generic function $u(t)$, which is a function of time (t). So, assuming that the value of the function at a generic time is well known, for example at time (t_n), and consider discretizing the time such that it will be possible to express each time step as t_k. In this way if the initial value that we are considering is $u(t_n)$, the next one will be $u(t_{n+1})$.

This $u(t_{n+1})$ is not known, however, knowing $u(t_n)$ we can derive it, obtaining $u'(t_n)$. Then, it is possible to define the time variation between the two time steps that will be: $\Delta t = t_{n+1} - t_n$.

Once we have outlined all these concepts, we can obtain the following equation:

$$u(t_{n+1}) = u(t_n) + \Delta t \cdot u'(t_n) \qquad (2\text{-}53)$$

The advantage of explicit integration becomes evident focusing on the right-hand side of the equation, where all terms that are present are known and pertain solely to the initial time step. By employing this iterative approach, all future time step values can be determinate through the initial value and the function derivative (Vrh 2009).

More in details, in this study, a specific explicit integration method has been exploited: the leap-frog integration, whose logic scheme is shown at Figure 21.

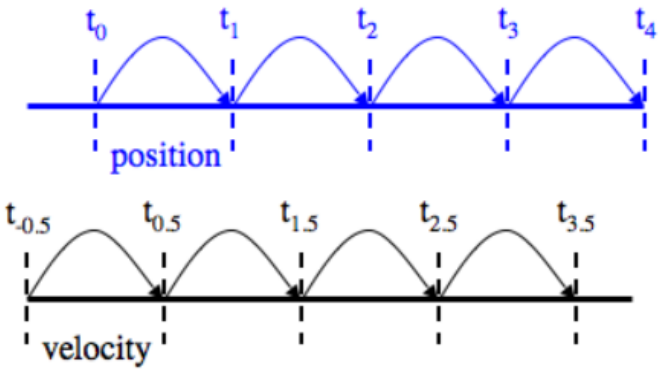


Figure 21 The leap-frog integration scheme (Image source: https://www.particleincell.com/2011/velocity-integration/).

In the leap-frog integration scheme, updates to position and velocity alternate such that each advances from one half step behind to one half step ahead of the other, effectively "leapfrogging" over one another (Wetzstein 2009), as it is outlined at Figure 21. More specifically, the governing equation of this integration method are in the following reported (Bui 2008) (Kelley 2018):

$$v_{n+1/2} = v_{n+1/2} \cdot \ a_n \cdot \Delta t \qquad (2\text{-}54)$$

$$x_{n+1} = x_n \cdot \ v_{n+1/2} \cdot \Delta t \qquad (2\text{-}55)$$

Where: *a* is the acceleration, *v* is the velocity, *x* is the position, Δt is the time step and *n* is the current time.

However, the primary challenge of the explicit integration method is linked to the size of the time step (Δt). When selecting the level of discretization, we must consider the following issues: if the time step is too small, it significantly escalates the computational effort required to solve the problem for all time steps.

Conversely, if it's too large, it can compromise stability, resulting in an unstable solution. Therefore, a trade-off must be struck to achieve a stable and efficient solution (Rio 2005).

To ensure stability, the Courant-Friedrichs-Lewy condition can be employed (Moura 2013). It dictates that the time step size must be smaller than the time required for a stress wave to traverse the smallest element length:

$$\Delta t = f * \frac{l}{c} \tag{2-56}$$

Where:

- f is a safety factor (lower than 1).
- l is the smallest element length.
- c is the sound speed wave in the material.

In this study the fluid that will be investigated is water, whose speed of sound is equal to 1480 m/s. However, referring to equation 2-56, it can be outlined that applying the effective speed of sound the resulting time step will be remarkably small (Molteni 2009).

To overcome this issue (J.J. Monaghan 1999) proposed a different method to set the parameter c. In a first step, through equation 2-57, which is simply the Torricelli equation, knowing the position of the deepest model particle the maximum fluid velocity can be calculated:

$$v_{max} = \sqrt{2gH} \tag{2-57}$$

Then, the incompressibility of the fluid must be ensured. Monaghan (J. Monaghan 1994) in his study has outlined that the density fluctuations are proportional to the M^2, where M is the Mach number. Thus, to keep the density variation below 1%, the Mach number must satisfy the following condition:

$$M^2 \leq 0.1 \tag{2-58}$$

Therefore, from the definition of the Mach number it is possible to define the artificial speed of sound as:

$$c \geq 10^2 v_{max} \tag{2-59}$$

Finally, inserting Equation 2-57 in Equation 2-59, the artificial speed of sound is outlined as:

$$c \geq 10 \cdot \sqrt{2gH} \tag{2-60}$$

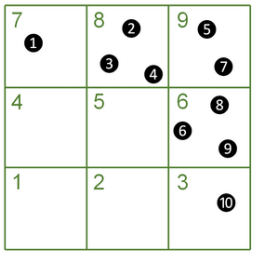
Neighbor particles search algorithms.

One of the cornerstone principles in SPH lies in the kernel function, which plays a crucial role in assigning weights to neighboring particles (refer to the "Kernel functions" section). Consequently, the identification of neighboring particles becomes imperative in the SPH approximation process. In scenarios where a solver lacks a dedicated neighbor particle search algorithm, all particles within the element are treated as neighbors for each particle at every time step. Subsequently, their contributions are computed using the defined smoothing length and kernel function.

However, in cases involving a large number of particles, most contributions will retrieve a value equal to zero, a valid outcome but an inefficient computational approach. Thus, the remedy lies in implementing a neighbor particle search algorithm (Fernández-Fernández 2022). This algorithm's function is to pinpoint particles within the smoothing length, thereby excluding those beyond it, whose contributions is zero (Viccione 2008). This optimized approach enables the application of the SPH solver even in models with an extensive particle count.

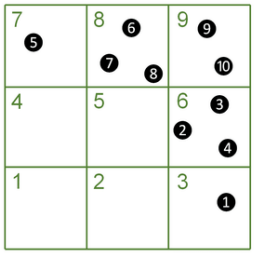
Various algorithms are available in the literature for this purpose, in particular, three specific algorithms were considered in the development phase of this solver:

- Linked-list algorithm (Domínguez 2010) (Cercos-Pita 2015), Figure 22. A linked list is a linear data structure where nodes are connected via pointers, rather than being stored contiguously in memory. Each node contains data and a pointer to the next node in the sequence. This dynamic structure allows for efficient memory allocation and deallocation during operations like insertion and deletion, making it adaptable to changing needs. It is particularly useful when the size of the list is dynamic and unknown beforehand or when efficient insertion and deletion operations are required. The list begins with a head node, pointing to the first node, while the last node, known as the tail node, points to null to signify the end of the list.



Before sorting

Particle	1	2	3	4	5	6	7	8	9	10
part_cel	7	8	8	8	9	6	9	6	6	3
part_idx	1	2	3	4	5	6	7	8	9	10



After sorting

Particle	1	2	3	4	5	6	7	8	9	10
part_cel	3	6	6	6	7	8	8	8	9	9
Part_idx	10	6	8	9	1	2	3	4	5	7

Cell	1	2	3	4	5	6	7	8	9
cell_beg	-	-	1	-	-	2	5	6	9
cell_end	-	-	1	-	-	4	5	8	10

Figure 22 Linked list algorithm search example (Peng 2019).

- Tree search algorithm (Purkayastha 2022). Tree search algorithms are methods used to traverse or search through tree data structures efficiently. Trees are hierarchical data structures consisting of nodes connected by edges, with each node containing a value and pointers to its children's nodes. The following are the most common algorithms.

  Depth-First Search (DFS): DFS explores a tree by traversing down one branch as far as possible before backtracking. It starts at the root node and recursively explores each branch until it reaches a leaf node.

  Breadth-First Search (BFS): BFS explores a tree level by level, starting from the root node. It systematically visits all nodes at a given depth before moving on to nodes at the next depth. BFS is typically implemented using a queue data structure.

  Binary Search Tree (BST) Search: BST search is used specifically for binary search trees, a type of tree where each node has at most two children, and nodes to the left have smaller values while nodes to the right have larger values.

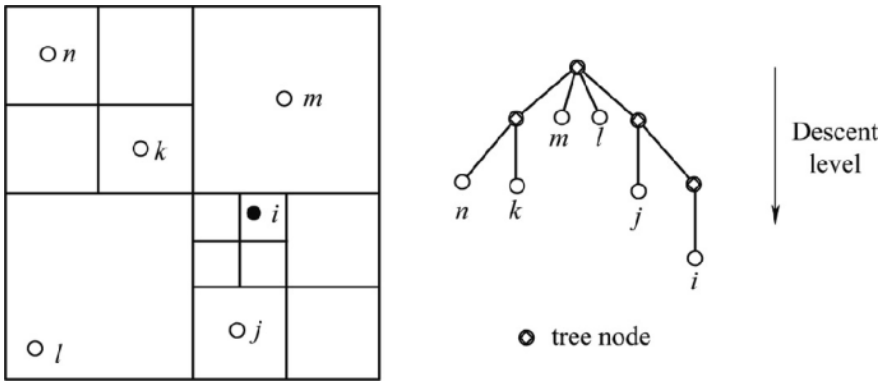


Figure 23 Tree search algorithm example (S. Chen 2023).

- Pairwise interaction (L. G. Liu 2003). The algorithm seeks pairs of particles within a specified distance, provided as the first parameter, typically the smoothing length h in SPH applications. Once h is established, the algorithm iterates over each particle i in the model, calculating the distance to all other particles j. If the distance r_{ij} between particle i and particle j is less than h, particle j is identified as a neighbor. If the support domain is symmetric, particle i is also recognized as a neighbor of particle j. Consequently, particle i and particle j form a pair of neighboring particles. This process is repeated for every particle in the model, resulting in an all-pair search approach for particles i = 1,2, ..., N, with searching conducted for all particles j = 1,2, ..., N. It is evident that the complexity of this all-pair search approach is $O(N^2)$.

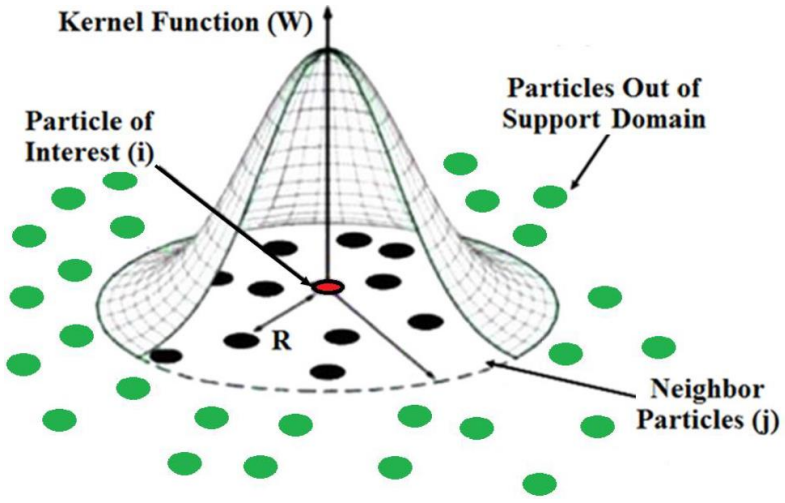


Figure 24 Pairwise interaction algorithm search example (Bagheri 2023).

2.6 The Sloshing Problem

The sloshing problem in fuel tanks has a significant consideration in various industries, ranging from automotive and aerospace to marine and industrial applications. It pertains to the dynamic movement of liquid fuel inside a tank when the tank-containing vehicle or vessel is in motion. This phenomenon poses multifaceted challenges that engineers and designers must address to ensure safety, stability, and operational efficiency.

Sloshing can lead to instability in vehicles, especially in high-speed situations or during sudden maneuvers. Excessive sloshing can cause the vehicle to lose control or stability, potentially leading to accidents. Furthermore, the dynamic movement of the fuel inside the tank can exert significant forces on the tank structure, leading to structural fatigue and potential damage over time. Moreover, Sloshing can affect the accuracy of fuel level measurements. When a vehicle is in motion, the fuel level sensor may not provide accurate readings due to the movement of the fuel. This can result in incorrect fuel level indications, leading to operational issues or unexpected fuel shortages (Rajagounder 2016).

Therefore, in the development phase of a vehicle, it is crucial to reduce as much as possible this phenomenon. Researchers have investigated this problem, through the decades, exploiting several numerical methods.

Moretti (Moretti 2014) in his work has investigated the contribution of the sloshing to NHV of the vehicle, simulating the problem through a FEM model, exploiting the ALE formulation (Arbitrary Lagrangian-Eulerian).

Moreover, to investigate the motion and the pressure variation inside of the tank, one the most exploited numerical method is the CFD (Computational Fluid-Dynamics) as performed by Singal (Singal 2013), who investigated the motion of the Kerosene inside of an aircraft tank under a uniform acceleration. The solution carried out in his study is to exploit three different baffles equally spaced inside of the tank, which are presenting two holes on their profile. The introduction of these components has remarkably reduced the fuel sloshing, as reported in Figure 25.

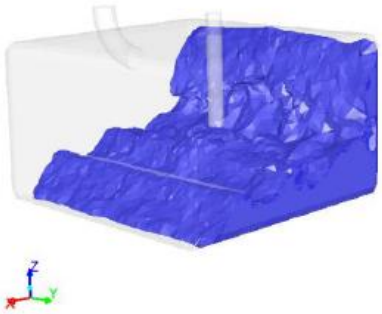


Fig.5 Liquid Interface at t = 0.45 seconds
(for tank with Baffles).

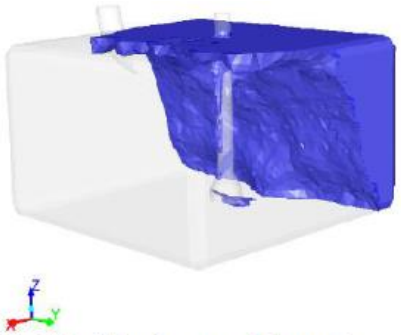


Fig. 6 Liquid Interface at t = 0.45 seconds
(for Tank without Baffles).

Figure 25 Fuel sloshing in an aircraft tank under uniform acceleration.

The mathematical reason behind the choice of introducing one or more baffles has been analyzed and discussed by (Rajagounder 2016). Taking into account a rectangular tank, as shown in Figure 26, and assuming that it is undergone a uniform acceleration, the line that can be drawn to outline free surface will change its inclination according to the intensity of the acceleration. Thus, the points belonging to the free surface will change their z_s coordinate, which can be expressed as:

$$z_s = h_0 + \frac{a_x}{g}\left(\frac{L}{2} - x\right) \tag{2-61}$$

Where:

- z_s is the z coordinate of a point belonging to the free surface.
- h_0 is the initial height of the free surface.
- a_x is the lateral uniform acceleration.
- g is the gravity acceleration.
- L is the length of the tank.

Through equation 2-61, it is possible to understand that to reduce the variation of the z coordinate it is necessary to reduce as much as possible the length of the tank.

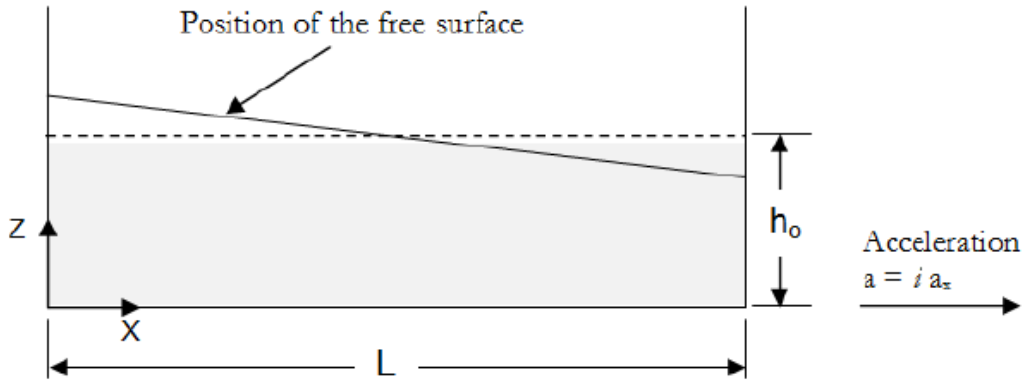


Figure 26 Fuel tank under uniform acceleration.

Thus, the role of the baffles is to divide the tank into smaller containers and in this way, we can obtain an immediate improvement in the fuel sloshing suppression, and this is the reason why they are deeply exploited in the industry for this purpose.

In particular, Zhang (Zhang 2020), in his study, analyzed different baffle shape and position inside of the tank, carrying out that: higher baffles are more effective than lower baffles, and positioning in parallel two or three baffles the sloshing is deeply reduced. Furthermore, Hosseini (Hosseini 2012) carried out that to optimize the fuel sloshing suppression, when a rectangular tank is involved, it is necessary to set the baffles equally spaced inside of the tank.

For what concerns the presence of holes in the baffles, Nimisha (Nimisha 2022) explains that perforated baffles, while useful for reducing the smooth, wave-like movements (convective sloshing) in a tank, are less effective at damping the sudden, forceful movements (impulsive sloshing). Solid baffles create a blockage that better reduces the impact of impulsive forces, leading to less severe sloshing. In contrast, perforated baffles allow more fluid flow, decreasing their ability to dampen these sudden movements, although they still help mitigate convective sloshing.

Besides the mesh-based methods, researchers are exploiting the advantages of the SPH (refer to "Introduction to the SPH" paragraph) to perform analysis regarding the fuel sloshing problem. In fact, Calderon-Sanchez (Calderon-Sanchez 2019) has chosen the SPH to model complex baffle geometries to be implement in the tank.

Moreover, Zheng (S. Zheng 2021) performed a topology optimization of the baffles to reduce fuel sloshing, in his work, he has exploited the SPH to investigate violent sloshing and the hybrid fluid–solid problems, because "SPH avoids mesh distortion and singularity problem and significantly enhances the computational accuracy at the same level of discretization resolution within violent sloshing problems" (S. Zheng 2021).

Furthermore, Hosain (Hosain 2018) studied the fuel sloshing in a tank of a carrier ship, exploiting and comparing two different methods: CFD and SPH. From this analysis it can be highlighted how the application of the SPH retrieves values that are more accurate and closer to the experimental data, if compared to the CFD ones.

In addition, Cai (Cai 2021), who investigate the rigid body force generated by fuel movement in the tank, through various simulations using different numerical methods were performed. The study revealed that the SPH simulation produced more accurate results compared to those obtained from the ALE (Arbitrary Lagrangian-Eulerian) simulation.

Thus, it has been decided to apply the developed SPH to a sloshing problem, to provide a further demonstration of its accuracy, but especially, to find out a possible design to suppress the fuel sloshing inside of an automotive tank.

3 PRE-, POST-PROCESSING AND SPH SOLVER IMPLEMENTATION

In this section is presented the solver logic scheme and the MATLAB script that has been developed for the realization of the SPH solver for incompressible fluids.

The diagram, at Figure 27, outlines that the initial model is developed in LS Pre-Post- and it is saved as a .k file. The solver, developed in MATLAB, performs the calculation solving the equation of motion approximated through the SPH method (refer to " SPH applied to incompressible fluids." section). Finally, the results are saved into a .vtk file and are visualized in Paraview.

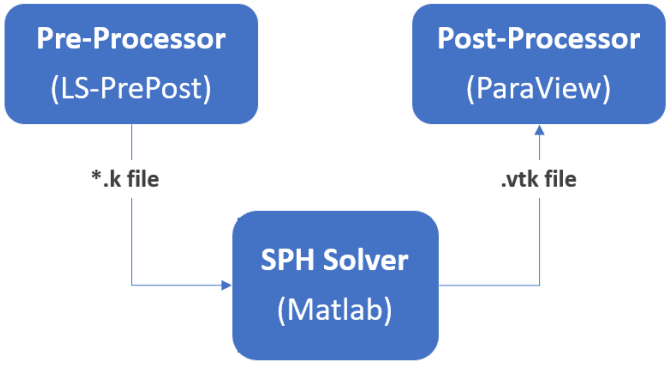


Figure 27 Solver schematics.

3.1 Pre-Processing

For the purposes of this thesis, the pre-processing program selected to generate the model is LS Pre-Post. This software facilitates the creation of 3D models and their representation through particle discretization. The model is then saved in a .k file, the native format for LS Pre-Post.

The initial step in the pre-processing phase involves analyzing the .k file generated by LS Pre-Post to identify and understand the useful information it contains. Consequently, the following section will discuss the .k file (SPH generation) structure modeled with LS Pre-Post:

- Header Information, this section often contains general information about the model, such as the software version, date of creation, model units, and other metadata, Figure 28.

```
$# LS-DYNA Keyword file created by LS-PrePost(R) V4.10.8-06Nov2023
$# Created on Apr-16-2024 (17:26:35)
*KEYWORD
*PART
$#                                                                 title
SphNode
$#     pid     secid       mid     eosid      hgid      grav    adpopt      tmid
         1         0         0         0         0         0         0         0
```

Figure 28 Header information .k file.

- “Element SPH”, is the part of the file in which it is possible to find the information related to the mass of the particles, the line is structured as it follows: particle number, element number to which the particle belongs, mass of the particle, as shown in Figure 29. The MATLAB script is able to read and save in the matrix the particle number and the mass.

```
*ELEMENT_SPH
$#   nid     pid            mass      nend
       1       1           0.125         0
       2       1           0.125         0
       3       1           0.125         0
       4       1           0.125         0
```

Figure 29 Mass definition in .k file.

- “Nodes”, in this part of the file are presents the information related to the position of the particles, in particular the line is structured: particle number, x coordinate, y coordinate, z coordinate, as displayed in Figure 30. However, it is important to highlight that if there are multiple objects present in the file, and some are not formed by particles but by nodes, it is fundamental to recognize when the file shows the coordinates of the nodes or of the particles, for this reason the script immediately saves the particle number, in order to distinguish the particles from the nodes.

```
*NODE
$#   nid               x               y               z      tc      rc
       1           0.025           0.025           0.025       0       0
       2           0.025           0.025           0.075       0       0
       3           0.025           0.025           0.125       0       0
       4           0.025           0.025           0.175       0       0
       5           0.025           0.025           0.225       0       0
       6           0.025           0.025           0.275       0       0
       7           0.025           0.025           0.325       0       0
       8           0.025           0.025           0.375       0       0
```

Figure 30 Particles coordinates definition in a .k file.

The subsequent step involves extracting the data from the .k file for use in the calculations. Notably, the .k file can be opened as a text file, making it readable by humans and allowing for the detection of relevant information.

This extraction process has been implemented in MATLAB allowing to extract the necessary data, such as the mass, position, and initial velocity of each particle, and organize them into a matrix, as reported in Figure 31. Once this matrix is created, the solver can be implemented, as it will have access to all the essential data in the initial matrix.

```
while ~ feof(fid1)
    tline1 = fgetl(fid1);
      if i==1
        check=sscanf(tline1,'%f')';
        TF=isempty(check);
        if TF==1
            i=0;
        end
        if TF==0
        m=sscanf(tline1,'%f')';
        mass=m(1,3);
        Mass(k,1)=mass; %The mass is in [kg]
        k=k+1;
        end
      end
    if length(tline1)==length('*ELEMENT_SPH')
        if tline1=='*ELEMENT_SPH'
            i=i+1;
            tline1 = fgetl(fid1);
        end
    end

end
```

Figure 31 "While" loop to extract the mass from the .k file.

3.2 Post Processing

One of the important tasks of this work is the selection of the appropriate post-processing software. When the solver completes its calculations, the results must be saved in a suitable output file that can be read by the post-processing software. A challenge encountered in this process is the lack of available information on the structure of many file formats, rendering them unsuitable for this work's purposes.

Files can be categorized into two types: binary and non-binary. Binary files contain sequences of zeros and ones, making them appear meaningless when opened as text files. Therefore, it is essential to have information about their structure to interpret the sequences correctly.

Non-binary files, on the other hand, can be opened as text files and are human-readable. Some files, such as .k files, have clear and immediately understandable information. Others, like TecPlot (.dat) files, are written in formats like ASCII.

In Table 3 are summarized all the file formats that have been analyzed and the respective post-processing software's.

Table 3 File formats analyzed.

Post-processing software	Input file format	
	Binary	Text Format (ASCII...)
LS Pre-Post-	.d3plot	
HyperView	.res	
	.op2	
	.h3d	
	.odb	
ParaView	.xdmf	.dat
	.tlp	.vtk
	.sztlp	

For this thesis, the chosen file format is .vtk, an ASCII file format. Consequently, ParaView has been selected as the post-processing software, as it can efficiently read and process .vtk files.

3.3 SPH solver

Starting from the general scheme of the solver, illustrated in Figure 32, the initial data extracted from the .k file are first utilized to compute the particle densities using the SPH method. Following this, the equation of state is employed to determine the pressure. The subsequent step involves solving the Navier-Stokes equations, approximated through the SPH method. In this process, the pressure is used to evaluate the pressure gradient, while the viscous term is calculated using the viscosity and the velocities of the particles from the previous time step. Once the acceleration is determined, it can be integrated using the leap-frog scheme to obtain the updated velocities and positions of all particles. Finally, before proceeding to the next time step, the new positions of the particles must be checked to ensure they satisfy the boundary conditions. This process will be repeated until the last time step.

In Table 4 are reported the values of the main parameters implemented in the solver.

Table 4 Solver parameters.

Parameter	Value	Unit
Water density	1000	kg/m^3
Water viscosity	0.001	$Pa \cdot s$
Gravity acceleration	9.81	m/s^2
Smoothing length	1.2 * spacing	-

In the next sections each step of the MATLAB script, whose scheme is displayed in Figure 33, is presented and described.

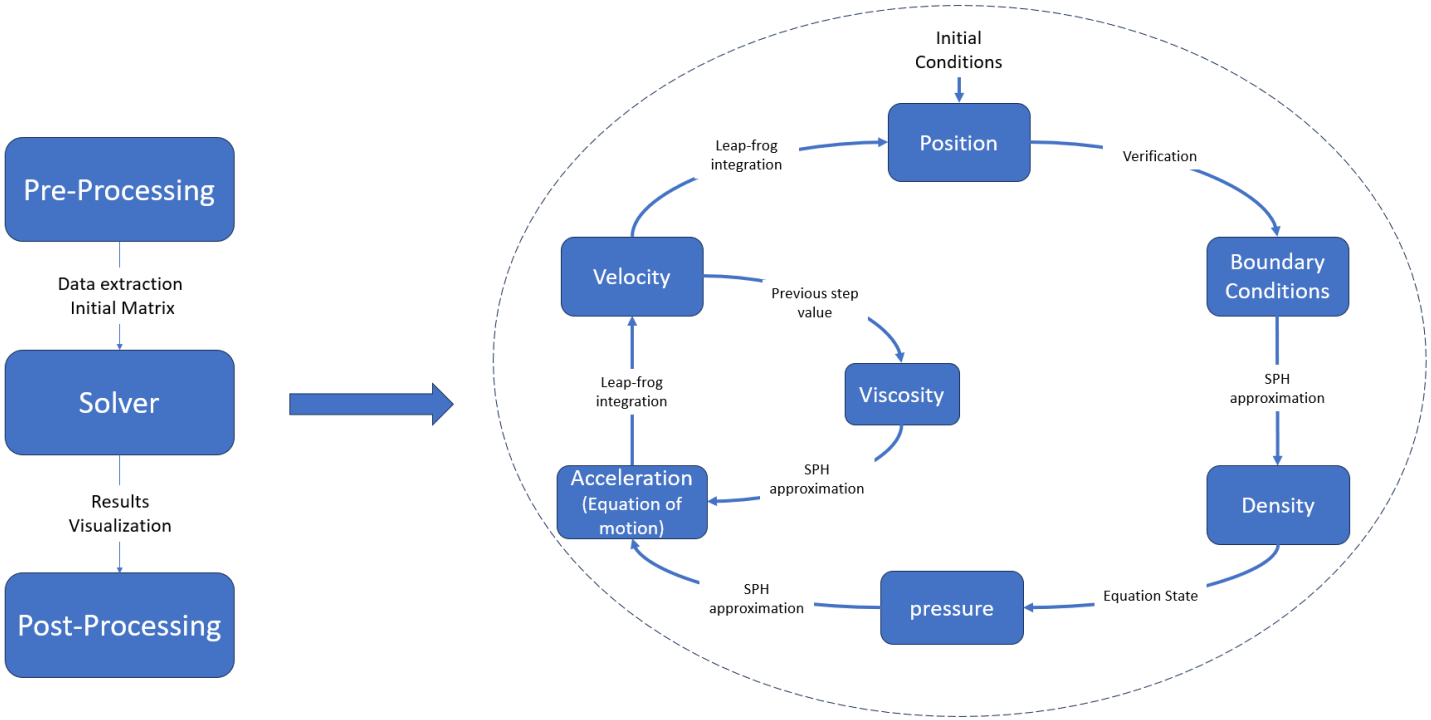


Figure 32 Solver general scheme.

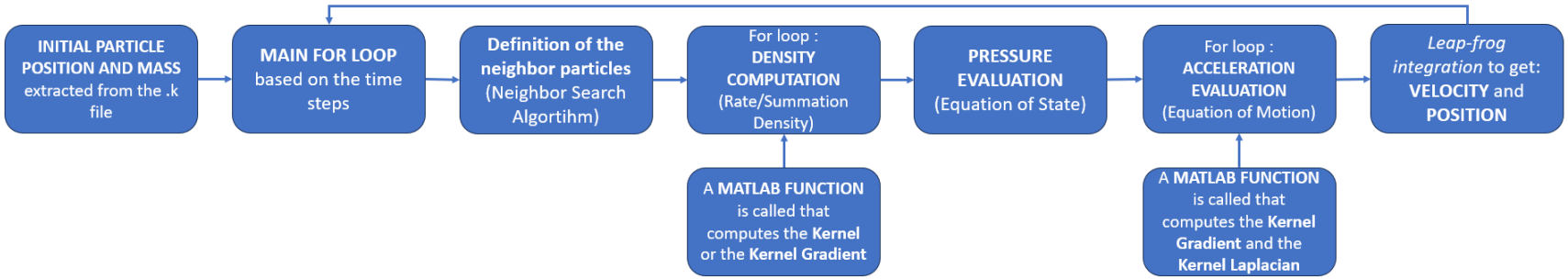


Figure 33 Solver script structure

3.3.1 Neighbor particle search algorithm

The algorithm that has been chosen is the Pairwise interaction algorithm because it is simple to implement in MATLAB through the arrays, as highlighted in Figure 34, but especially, because it is very effective, allowing to reduce drastically the computational time.

The first step is to define the distances between all the points belonging to the matrix "Matcoord", where the coordinates of each particle are listed. Then through the MATLAB function "squareform" a logical value is assigned (1 or 0) to all the points of the matrix, more specifically 1 is assigned if two points are within the distance h, the smoothing length, 0 if their distance is larger than h. Then, trough the MATLAB function "find" it is possible to extract the index of those particles that are within the smoothing length. Finally, the pairs *I* and *j* are listed inside two arrays named "Pair".

The only drawback of the solution reported in Figure 34, is that the array called "D", when the number of particles is considerable becomes very long, it requires a discrete amount of memory (for instance, for a simulation of 40000 particles it requires 14 GB of RAM to manage quickly the array).

```
% Calculate the distance between all the particle pairs in the matrix
D = pdist(Matcoord);

% Build the Adjacency Matrix based on the distance
AdjacencyMatrix = squareform(D <= h);

% Find the pairs that are within the distance h
[idx_j, idx_i] = find(AdjacencyMatrix);

% Define the Arrays containin the particle number
Npairs = length(idx_i);
Pair_j(1:Npairs) = idx_j;
Pair_i(1:Npairs) = idx_i;
```

Figure 34 Pairwise interaction algorithm developed in MATLAB.

3.3.2 Density evaluation

Once the pairs have been defined, the density for all particles can be computed using a simple for loop, as depicted in Figure 35. The indices *i* and j enable the selection of the correct particle. First, the relative speed between particles is evaluated, called "u_{ij}". Next, the gradient of the kernel function, "$gradW_{ij}$", is calculated based on their positions. Finally, the density rate, "rho_dot", is obtained.

```
for k=1:Npairs
    i=Pair_i(1,k);
    j=Pair_j(1,k);
    ri=Matcoord(i,:);
    rj=Matcoord(j,:);
    uij=Velocity(i,:)-Velocity(j,:);
    gradWij=[gradkernel(h,ri,rj)]';
    rho_j=mass*uij*gradWij;
    rho_dot(i,1)=rho_dot(i,1)+rho_j;
end
```

Figure 35 Density evaluation (Rate Density approach).

3.3.3 Pressure evaluation

From the density of each particle, it is possible to implement the equation of state to outline the pressure, as it is illustrated in Figure 36.

```
%Pressure evaluation
B=c^2*rho_rest/gamma;
p=B*(((rho/rho_rest).^gamma)-1);
```

Figure 36 Pressure evaluation.

3.3.4 Equation of motion computation

Figure 33 shows the second for loop of the script, which is responsible for solving the equation of motion for incompressible fluids. In the initial phase, the positions ("Matcoord"), densities ("rho"), and pressures ("p") for each *i*–*j* particle pair are defined. Using these parameters, the gradient and Laplacian of the kernel function are computed ("gradW$_{ij}$" and "grad2W$_{ij}$" respectively), allowing the calculation of the pressure and viscous terms ("accp" and "accv" respectively).

```
for k=1:Npairs
    i=Pair_i(1,k);
    j=Pair_j(1,k);
    ri=Matcoord(i,:);
    rj=Matcoord(j,:);
    rho_i=rho(i,1);
    rho_j=rho(j,1);
    pi=p(i,1);
    pj=p(j,1);
    vi=Velocity(i,:);
    vj=Velocity(j,:);
    gradWij=gradkernel(h,ri,rj);
    m_j=mass;
    accp_j=m_j*((pi/(rho_i^2))+(pj/(rho_j^2)))*gradWij;
    grad2Wij=grad2kernel(h,ri,rj);
    accv_j=(viscosity/rho_i)*m_j*(grad2Wij/rho_j)*(vj-vi);
    accp(i,1:3)=accp(i,1:3)+accp_j;
    accv(i,1:3)=accv(i,1:3)+accv_j;
end
acc=g-accp+accv;
```

Figure 37 Acceleration evaluation solving the equation of motion.

3.3.5 Leap-frog integration

The acceleration is integrated using the leap-frog scheme. For the first time step, the velocity is obtained by multiplying the acceleration by half of the time step (*dt/2*). This approach ensures that the velocity and position will continue leapfrogging each other for all subsequent time steps.

```
%Leap frog integration method to evaluate velocity and position
if step==1
    Velocity(1:length(ParticleNumber),:)=Velocity(1:length(ParticleNumber),:)+acc*dt/2;
end
if step > 1
    Velocity(1:length(ParticleNumber),:)=Velocity(1:length(ParticleNumber),:)+acc*dt;
end
Matcoord=Matcoord+Velocity(1:length(ParticleNumber),:)*dt;
```

Figure 38 Leap frog integration scheme.

3.3.6 Boundary Conditions

In this work it has been decided to treat the impact of the particles against the wall as an elastic impact. When the particle hits the wall, that mathematically is represented by a straight line, its velocity is reversed multiplied by a damper coefficient that has been tuned through a trade and error procedure, fixing its value equal to 0.1.

In the second validation example, the ramp will be implemented through the use of repulsive particles.

In the sloshing problem, the baffles have been designed exploiting the boundary particles.

4 VALIDATION RESULTS

4.1 Pre-Processing

During the pre-processing phase, a sample model was generated in LS Pre-Post, illustrated in Figure 39. To assess the solver's ability to interpret information from the .k file produced by LS Pre-Post, it was essential to plot the particles and generate a model representation in MATLAB, depicted in Figure 40. Upon comparing these visuals, it is evident that the solver accurately interprets the .k file information, establishing the initial matrix with precision, listing mass and particle coordinates, and faithfully reproducing the exact model.

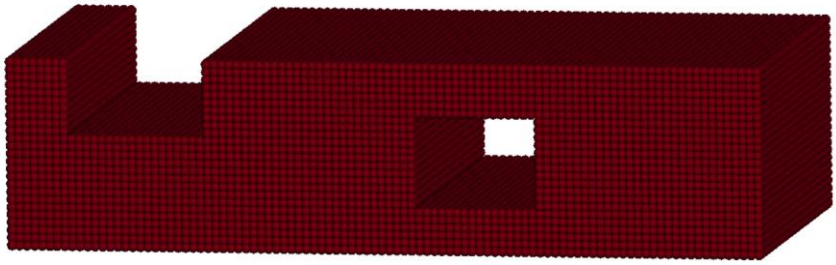


Figure 39 Sample model defined in LS Pre-Post-.

Figure 40 Sample model defined in MATLAB.

4.2 Post-Processing

To verify the solver's capability to accurately save and visualize results, a straightforward yet eye-catching simulation was conducted. Half of the sample model's particles were assigned a constant velocity vector of (0, 5, 0), indicating motion only along the positive y-direction, while the remaining particles retained a velocity of zero. The simulation spanned 10 time steps, allowing for swift verification of particle movement accuracy.

Examining the sequence depicted from Figure 41 to Figure 43, it becomes evident that particles are indeed moving in the expected direction, affirming the MATLAB solver's proficiency in post-processing results.

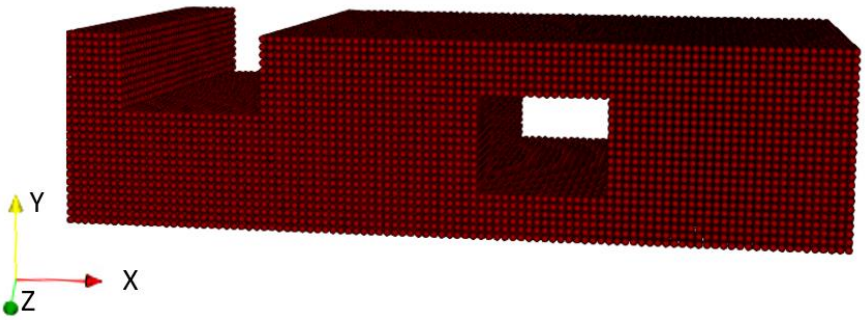


Figure 41 Model at step zero (initial position).

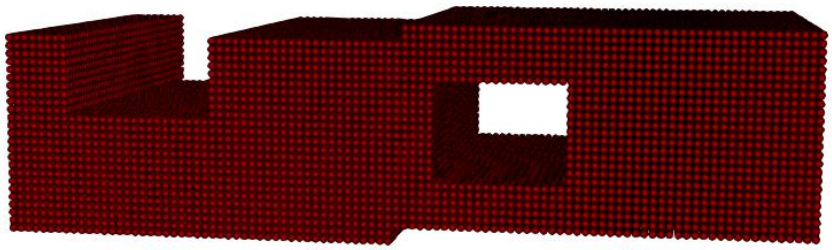


Figure 42 Model at step five (half simulation).

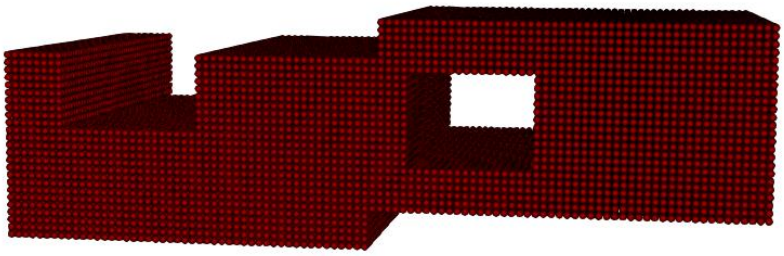


Figure 43 Model at step ten (final position).

4.3 SPH Solver

There exist several benchmark problems with pre-available experimental results that can be reproduced and analyzed to validate an SPH solver. In this section the analysis of the results of three such benchmark examples will be presented:

- Dam break within a box.
- Dam break with a ramp.
- Fall of a droplet in water.

In particular, the simulation results will be compared with respect to the experimental data and the results that are available from the original papers.

To ensure a proper comparison it is necessary to normalize the simulation time. This is a common practice for the benchmark problems, as it allows comparing models that do not have the same dimensions (Nair 2014). The reason for which the experiment and the simulation may have different dimensions is that, when an experiment involves a very large amount of fluid (several tens of meter cubes), to reach a sufficient discretization level in the model, it would be necessary to employ an extensive number of particles, defining a simulation that requires a great computational effort. On the other hand, reducing spatial dimensions of the model while keeping the same number of particles allows finer discretization and lower computational cost. The normalization factor most commonly used in literature (M. Liu, Modeling incompressible flows using a finite particle method 2005) is defined as follows:

$$Normalization\ factor\ = \sqrt{\frac{2 * g}{L}},$$

where g is the gravity acceleration and L is the most relevant dimension for that specific model (e.g., side length for a cube, diameter for a sphere, etc.).

Due to the fact that in this thesis the validation process is carried out taking into account both numerical and graphical results, for each example, the results will be divided into two different sub-sections. Finally, the outcomes of the two solvers will be compared identifying for each case, which solver will retrieve the best results, based on the numerical and graphical accuracy, and on the computational time required to solve the problem.

Regarding the evaluation of graphical accuracy in this study, the following procedure was followed:

- Identify from the published experimental studies, where available, the pictures showing the evolution of the liquid shape at specific time intervals.
- Extract from these images the water profile, as done in Figure 52-A.
- Overlap the experimental water profile on the pictures showing the simulation results, as reported in Figure 52-B.
- Knowing the dimensions of the model, it is possible to define a sufficient number of points to recreate the water profile in MATLAB, as shown in Figure 44. To achieve this result, the code presented in Figure 45 was developed. The coordinates of the water profile were extracted using the software "Engauge Digitizer," saved in Excel, and then imported into MATLAB. Using the "polyfit" function in MATLAB, the desired polynomial function can be defined by selecting the polynomial degree with the variable "grade" to recreate the water profile curve with optimal accuracy.

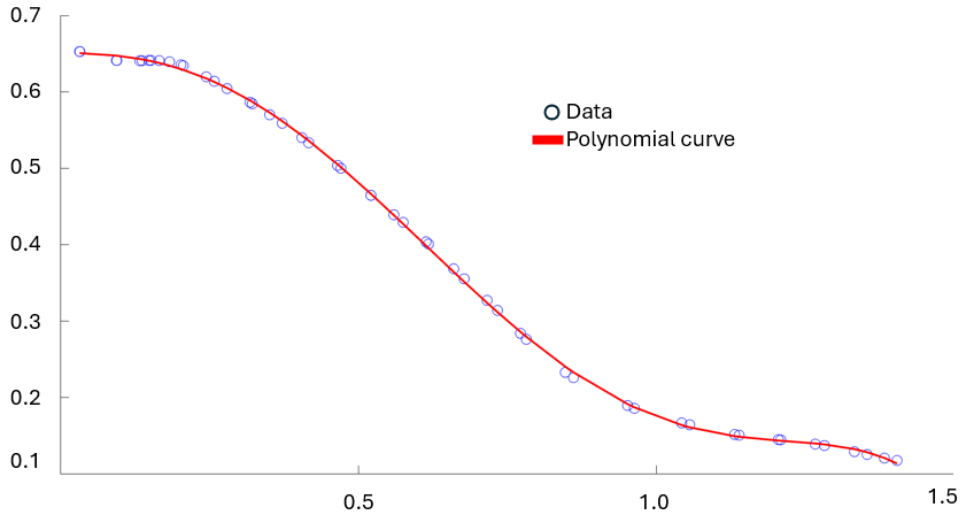


Figure 44 Water profile curve representation in MATLAB.

```
% Extract the columns X and y
X = data.x;
y = data.Curve1;

% Choose the grade of the polynomial function
grade = 3;

% Adpat the model to the data
coefficients = polyfit(X, y, grade);

% Define the function equation
polynomial = @(x) polyval(coefficients, x);
```

Figure 45 Function definition in MATLAB.

- Once the polynomial curves have been computed, it is possible to evaluate the area beneath them by performing the integration. However, it is important to note that when two curves intersect at one or more points, for each interval defined by two intersection points, it is necessary to compute the difference between the two areas, as described by Equation 4-1. Looking at Figure 46, it becomes clear why this process is essential: when the simulation curve is above the experimental one (Blue Area), the calculated difference will be negative. However, after the intersection (Yellow Points), the simulation curve will be below the experimental one (Red Area), resulting in a positive difference. Thus, when evaluating the overall discrepancy across the entire integration domain, these differences will compensate each other, leading to an underestimation of the discrepancy. Therefore, computing the difference at each interval overcomes this issue. Then, to obtain the overall discrepancy, as described by Equation 4-2, it is necessary to compute the average of all the calculated differences (D_i), weighting their contribution by the length of their respective intervals (L_i).

$$D_i = \frac{A_{exp} - A_{sim}}{A_{exp}} \cdot 100 \quad [\%] \tag{4-1}$$

$$Discrepancy = \frac{\sum_j^n D_i \cdot L_i}{L_{tot}} \quad [\%] \tag{4-2}$$

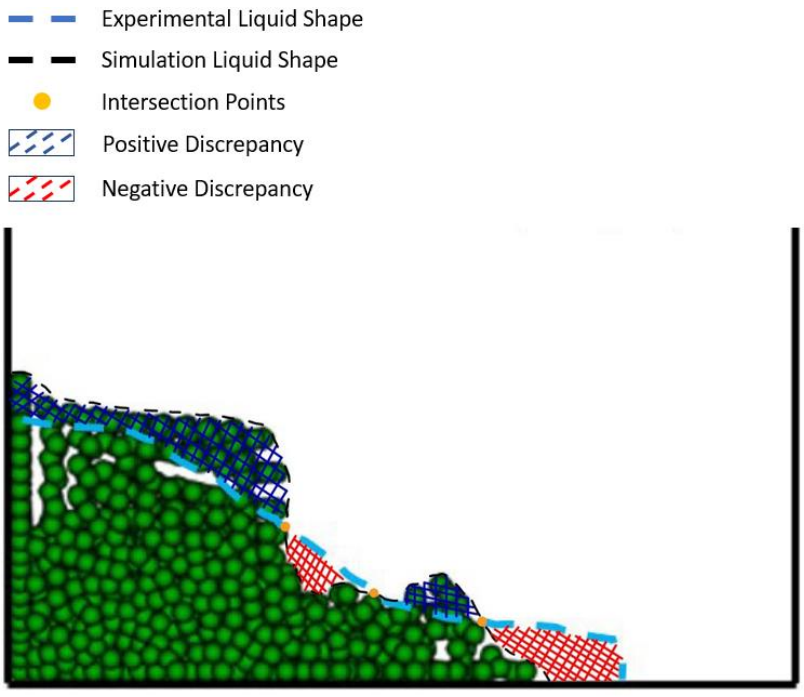


Figure 46 Discrepancy between simulation and experimental outcomes.

- Finally, compute the accuracy, through Equation 4-3, and assign to the solver a score according to Table 5.

$$Accuracy = 1 - Discrepancy \qquad [\%] \tag{4-3}$$

Table 5 Accuracy Legend.

Accuracy	Grade
Poor	$60\% \leq Acc$
Satisfactory	$60\% < Acc \leq 75\%$
Good	$75\% < Acc \leq 90\%$
Excellent	$90\% < Acc \leq 100\%$

4.4 Dam break within a box.

Scientists have started investigating the nature and the behavior of the dam break problem in the early 19th century. The introduction and utilization of computers in dam-break research led to the development of numerous modelling technologies, and the mathematical methods employed expanded in tandem with the computational capabilities of computers. For this reason, the dam break problem is considered a classical benchmark example in the validation of incompressible fluid solvers (Milan Toma 2021).

The example that was analyzed in this study, is shown in Figure 47. The water is contained between a lateral wall of the box and a removable separator. In its initial configuration, looking from the side view, the column of water presents a rectangular shape. The dimensions of the model are described through the parameter L, which in this study it is set to 0.5 m, as reported in Table 6. The simulation duration is equal to 2 seconds.

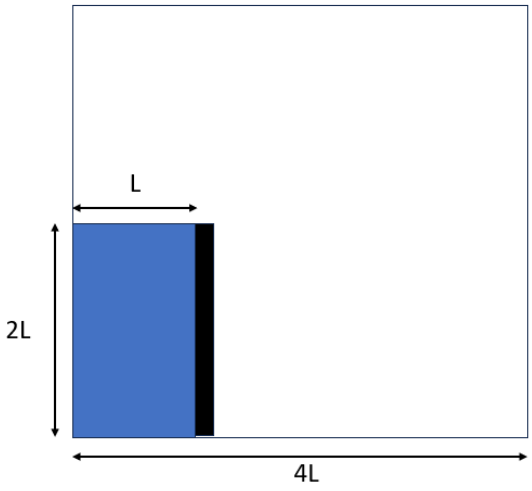


Figure 47 System representation (side view).

To perform the simulation, the system has been modeled in LS Pre- Post-, through the option "SPH Generation", obtaining the discretized model which is represented in Figure 48 . The system comprises 4000 particles distributed as follows: 10 in the x direction, and 20 in both the y and z directions. However, leveraging the symmetry along the x-axis, it's feasible to model it with half the particles, specifically 10 in x, 10 in y, and 20 in z. This modification significantly reduces computational time.

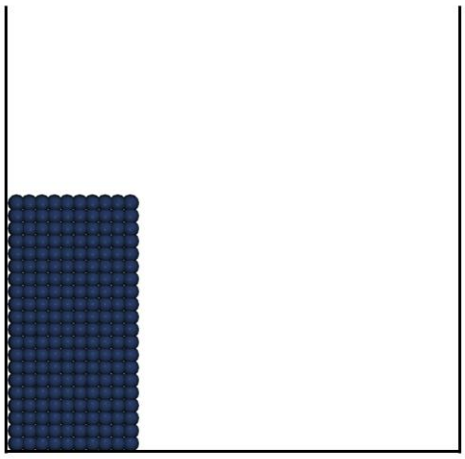


Figure 48 System discretized (side view).

Given a maximum particle speed of 10 m/s during simulation, we can estimate the artificial speed of sound (Equation (2-60)) as 100 m/s for timestep calculation. Using the Lewy-Courant condition and knowing the particle diameter, we can determine the timestep. For the initial setup, it's $3.1023 \cdot 10^{-5}$ s, resulting in a total of 64469 timesteps for the entire simulation, as summarized in Table 6.

Two different approaches are followed: same simulation duration and same time step. The cubic spline kernel has a support domain twice as large as that of the Spiky and Poly6 kernels, making it computationally more expensive (refer to "Kernel functions" section). To ensure a fair comparison, simulations must run for the same duration. Therefore, the time step for the Poly6 and Spiky kernels has been reduced by a factor *f*. In particular, for these simulations, the time step is $4.344 \cdot 10^{-6}$ s, resulting in a total of 460400 timesteps for the entire simulation.

To evaluate the impact of the time step on the simulation results, an additional simulation was conducted with an equal time step for all kernels to observe any resulting changes, as reported in Table 7.

Table 6 Simulation Parameters (same simulation duration).

	Rate Density		Density Summation	Unit
Num of Particles	2000	2000	2000	-
L	0.5	0.5	0.5	m
Kernel Function	Cubic Spline	Poly6	Spiky	-
Simulation Duration	10.5	11	10	h
Time Step	0.031023	0.004344	0.004344	ms
Num of Time Steps	64469	460400	460400	-

Table 7 Simulation Parameters (same time step)

	Rate Density		Density Summation	Unit
Num of Particles	2000	2000	2000	-
L	0.5	0.5	0.5	m
Kernel Function	Cubic Spline	Poly6	Spiky	-
Simulation Duration	10.5	2.75	2.5	h
Time Step	0.031023	0.031023	0.031023	ms
Num of Time Steps	64469	64469	64469	-

4.4.1 Numerical Results

In this section, the numerical results related to first validation example are presented and discussed. As reported in Table 6, three different simulations have been carried out: two applying the Rate Density approach (the first exploits the Cubic Spline kernel, the second exploits the Poly6 kernel), and one applying the Summation Density approach (exploiting the Spiky kernel).

As it was previously mentioned (refer to "Mass Conservation" section), the Rate Density approach does not automatically guarantee mass conservation during simulations. Therefore, this is the first metric that must be analyzed. It can be seen in Figure 49, that the mass is conserved in both simulations conducted in this study with the rate density solver. In particular, for the Cubic Spline, the highest difference between the actual mass and the initial mass was observed at 0.12 s, where the mass is equal to 500.32 kg, representing a difference with respect to the original value of only 0.064 %. For what concerns the

simulation with the Poly6 kernel, the highest discrepancy is noted at 0.24 s, where the mass is equal to 500.25 kg, representing a deviation from the initial value of only 0.050 %.

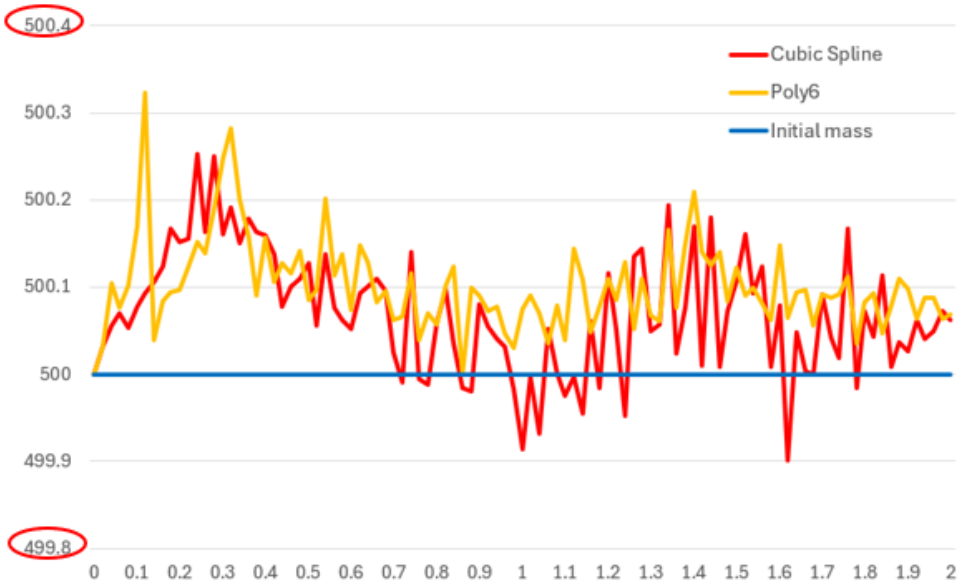


Figure 49 Evolution of the mass model along the simulation.

Once it has been verified that mass is conserved, we can proceed to the next set of numerical results based on the ratio between the actual front surge of the water and the initial one (Z/L). Two different criteria were used to compare the three simulations. First, the simulations were conducted for the same duration, meaning the time step varied according to their computational cost, as shown in Table 6. In the second criterion, the time step was constant for all three simulations, so the simulation duration varied according to the computational cost, illustrating the impact of the time step on the solvers.

Starting from the simulations with the same duration, whose numerical results are reported in Figure 50, it can be outlined that, all three simulations retrieve values which are in agreement with the experimental data and also with the simulation results obtained by (S.Koshizuka 1995). Additionally, the slope of all curves remains nearly constant throughout the entire simulation, indicating a consistent physical velocity gradient among all particles, ensuring smooth and realistic movement. Moreover, the simulation results, using the Poly6 kernel, closely align with experimental data compared to both Koshizuka's simulation and the Cubic Spline kernel implementation. A closer examination of the graph reveals that, in the final moments, the Poly6 kernel simulation yields values slightly below the experimental data, suggesting a slightly lower particle velocity than expected. This observation can also be made by examining the outcomes of the simulation that exploits the Summation Density approach.

For the Rate Density solver, with the same simulation duration the Poly6 kernel yields the most accurate values. However, the best results come from the simulation using the Summation Density solver, which matches the experimental data for more than half of the points.

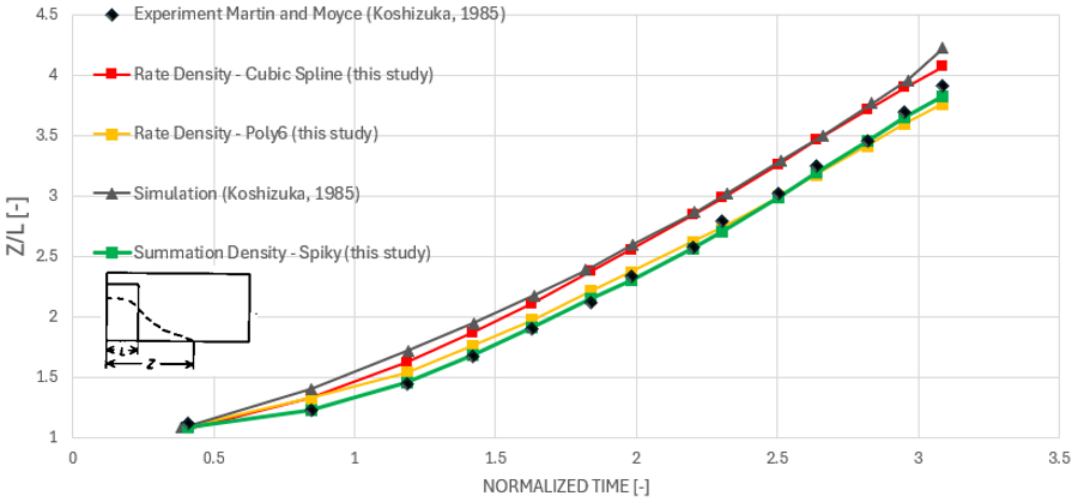


Figure 50 Evolution of the front surge of water with respect to its initial value (constant simulation duration).

Continue the analysis with the same time step simulations, but that present a different duration, whose results are reported in Figure 51.

The initial examination of the graph highlights that when the time step is reduced, to match the one of the Cubic Spline, the simulation using the Poly6 kernel retrieves values comparable to those of the Cubic Spline and Koshizuka's simulations. However, from a normalized time of 2.5 onwards, it is evident that the Poly6 kernel retrieves values worse than those of the Cubic Spline, which contradicts the previous analysis. This indicates that for the Rate Density solver, with the implementation of the Poly6 kernel, the time step has a significant impact on the numerical results, altering them completely.

Regarding the Summation Density solver, the graph shows it remains the most accurate simulation among the three. However, the slope of the curve between the normalized time intervals of 1.5 to 2 and from 2.75 to the end of the simulation is not constant. This implies that the velocity gradient is not completely uniform. Thus, the results of the Summation Density solver also deteriorate when the time step is reduced.

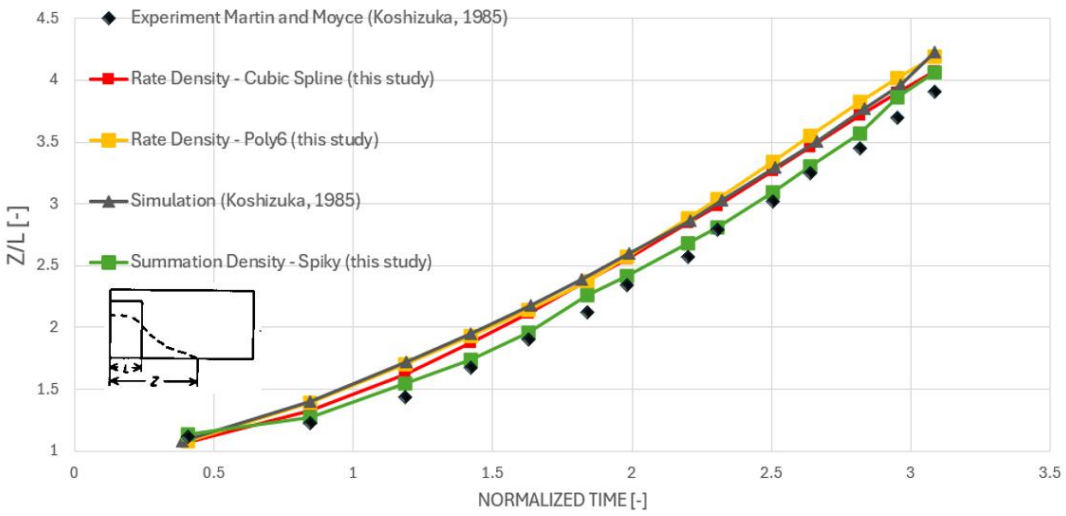


Figure 51 Evolution of the front surge of water with respect to its initial value (constant time step).

4.4.2 Graphical Results

In this section the graphical outcomes of the simulations are discussed and compared to the experimental pictures (S.Koshizuka 1995).

Starting from Figure 52 with the Rate Density approach, it can be highlighted that both simulations – with Cubic and Poly6 kernels – are able to represent the correct descendent movement of particles belonging to the unconstrained side of the model. Moreover, during this motion all the particles are moving together, showing an effective implementation of the SPH equations. However, considering the shape of the water flow, the implementation of the Cubic Spline kernel defines a more accurate representation if compared to the experiment (the discrepancy is equal to 9.2%), while the Poly6 kernel shows a less smooth flow, especially on the left boundary, where some particles are struggling to fall down, and they are still close to their initial position, this influences remarkably the discrepancy calculation, which is equal to 22.1%.

Moving on the Summation Density outcomes, it appears evident already from the qualitative point of view that this is the worst result, in fact the particles belonging to the upper layers struggle to fall down. In particular, it is still possible to recognize the initial rectangular shape of the model, which is clearly an important difference with respect to the experimental picture. This is also highlighted by discrepancy that has been calculated and results being equal to 31.06 %, that is the worst value among the three.

Transitioning to Figure 53, starting from the Rate Density solver (Figure 53 -B and -C), the particle motion in both simulations exhibits remarkable similarity, closely aligning with experimental observations. A more detailed look at the pictures reveals that the implementation of the Poly6 kernel yields a smoother shape which is closer to the one depicted in the experimental picture (Figure 53 - A), in fact the calculated

discrepancy is equal to 13.96%. While the outcomes related to the Cubic Spline kernel are affected by the particles motion close to the left boundary. These for few instants struggle to fall down, therefore, a discrete number of particles is above the experiment water profile, deteriorating the accuracy of the graphical outcomes, with a discrepancy which is equal to 15.84%.

For what concerns the Summation Density solver, whose results are reported in Figure 53 – D, several particles on the left boundary are bonded to the vertical wall, which clearly is a remarkable difference with respect to the experiment. Nevertheless, the remaining particles define a model profile which is close to the one observed experimentally, with just few oscillations of the free surface, in fact its discrepancy is equal to 22.7%. Also, for this second time step, the Summation Density solver is the least accurate, even though its discrepancy is comparable to the one defined by the simulations with the Rate Density solver.

Finally, focusing on Figure 54, in particular on the water that has just impacted on the wall, and it is producing a wave in the opposite direction of the initial flow. Let us start from the results belonging to the Rate Density solver simulations. Considering Figure 54-B and Figure 54-C, they outline that, as expected, the particles after the impact are producing the above-mentioned wave. However, the main difference with respect to the experiment (Figure 54-A) is that, with the implementation of the Poly6 kernel the simulation presents a great number of "splashing" particles, which are not seen in the experiment, this remarkably influences the discrepancy computation that is equal to 11.23%. This can be a consequence of the supporting domain of the kernel function. In fact, the Poly6 kernel has a support domain of $1h$ (refer to the "Kernel functions" section), meaning that each particle has smaller number of neighbors influencing its movement. On the other hand, this condition leads to a lower computational time and effort with respect to kernels with a larger support domain. In fact, considering Figure 54-B, the cubic spline kernel, which presents a support domain of $2h$, defines a smoother flow of the particles after the impact with the wall. This allows to get a wave that is more compact, without splashing particles, producing the result that agrees very well with the experiment, outlining a discrepancy which is equal to 8.52%.

Concluding, looking at Figure 54-D, where are presented the results related to the Summation Density solver, the same phenomenon, described for the left boundary, in Figure 53-D, can be now seen on the right wall of the box. In fact, several particles, after hitting the vertical wall, remain attached to it, exhibiting exclusively vertical motion. Even though other particles can reverse their motion, they are not able to generate a compact wave, defining a great number of splashing particles, which clearly is quite far from the observed water motion in the experiment, with the discrepancy that reaches the value of 46.68%.

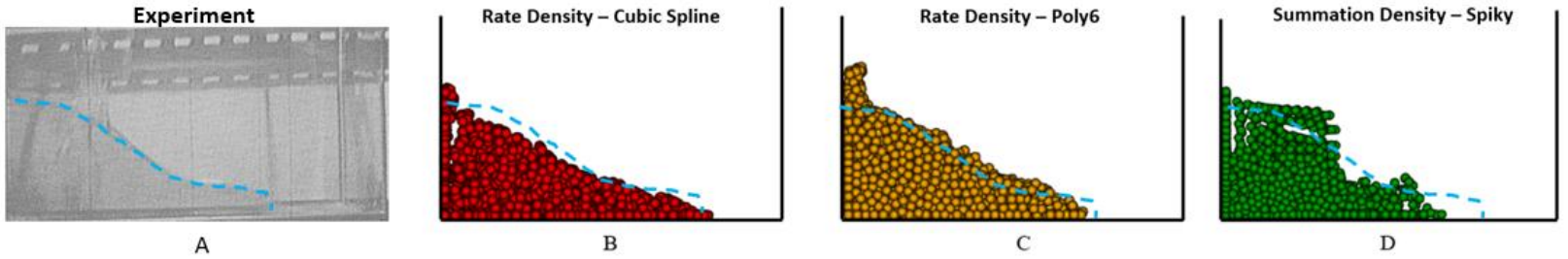


Figure 52 Results at the normalized time of 2.36 (-A experiment picture (S.Koshizuka 1995), -B Rate Density-Cubic Spline simulation frame, -C Rate Density-Poly6 simulation frame, -D Summation Density-Spiky simulation frame).

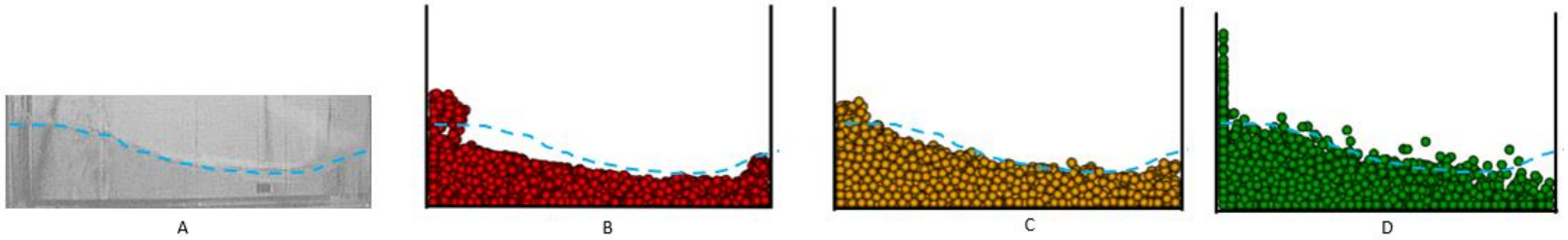


Figure 53 Results at the normalized time of 3.48 (-A experiment picture (S.Koshizuka 1995), -B Rate Density-Cubic Spline simulation frame, -C Rate Density-Poly6 simulation frame, -D Summation Density-Spiky simulation frame).

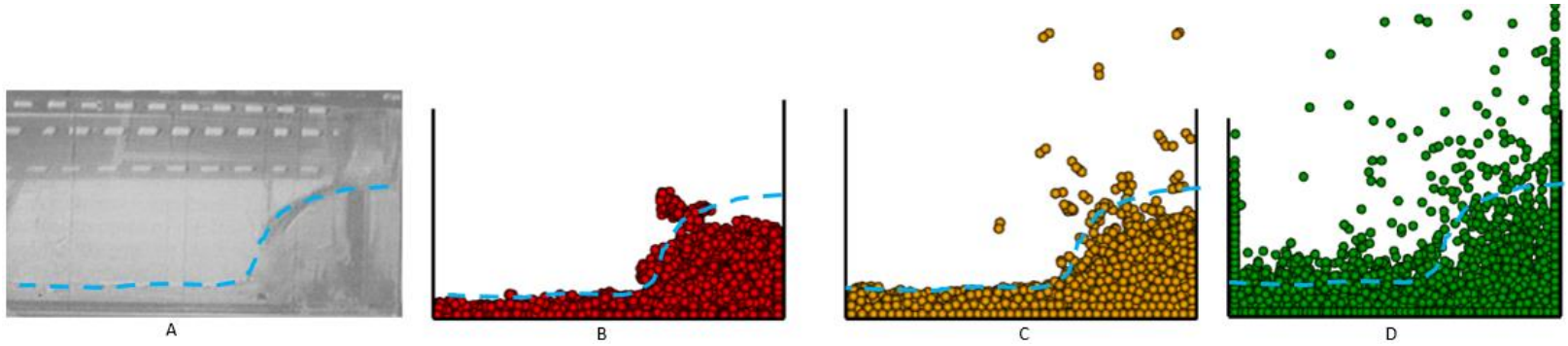


Figure 54 Results at the normalized time of 8.12 (-A experiment picture (S.Koshizuka 1995), -B Rate Density-Cubic Spline simulation frame, -C Rate Density-Poly6 simulation frame, -D Summation Density-Spiky simulation frame).

4.4.3 Comparison of the solvers

The comparison between the Rate Density and the Summation Density solver, for this first example, can be divided into two parts: before and after the impact against the vertical wall.

For what concerns the numerical results prior to the impact against the vertical wall, it can be pointed out that both solvers retrieve values which are consistent with the experiment data. Then, the slope of both simulation curves, shown in Figure 50, is generally constant. Predictions of the Summation Density solver for the front surge motion are the most accurate, with a discrepancy with respect to the experimental data of 3.39% only. Moving to the Rate Density solver with the implementation of the Poly6 kernel, it retrieves values not too far from these, with an error of 3.78%, while the implementation of the Cubic Spline outlines the highest numerical discrepancy, 8.46%. Considering the graphical results shown in Figure 52 - Figure 53 , the Rate Density Solver reproduces water motion in the most physical way. This was especially the case for the solver exploiting the Cubic Spline kernel, which was able to predict particle motion that was visually smoothest among the three simulations, showing the highest mean accuracy with respect to the experimental pictures equal to 87.5% (90.8% at normalized time 2.36 and 84.16% at normalized time 3.48), as highlighted in Figure 55. Thus, considering this first part of the simulation only, it is not completely clear which solver works better.

For the next stage, i.e. after the impact against the vertical wall, displayed in Figure 54, here it is evident that the Rate Density solver with cubic spline kernel is capable of reproducing the motion of water better than the other two solvers. When used with the Poly6 kernel, the solver was able to predict motion that was not too far from reality (88.77%), but clearly was rougher compared to that predicted with the cubic spline kernel (91.48%), as reported in Figure 55. Finally, the Summation Density solver exhibited issues with the motion of particles near boundaries leading to the least physical representation of water motion.

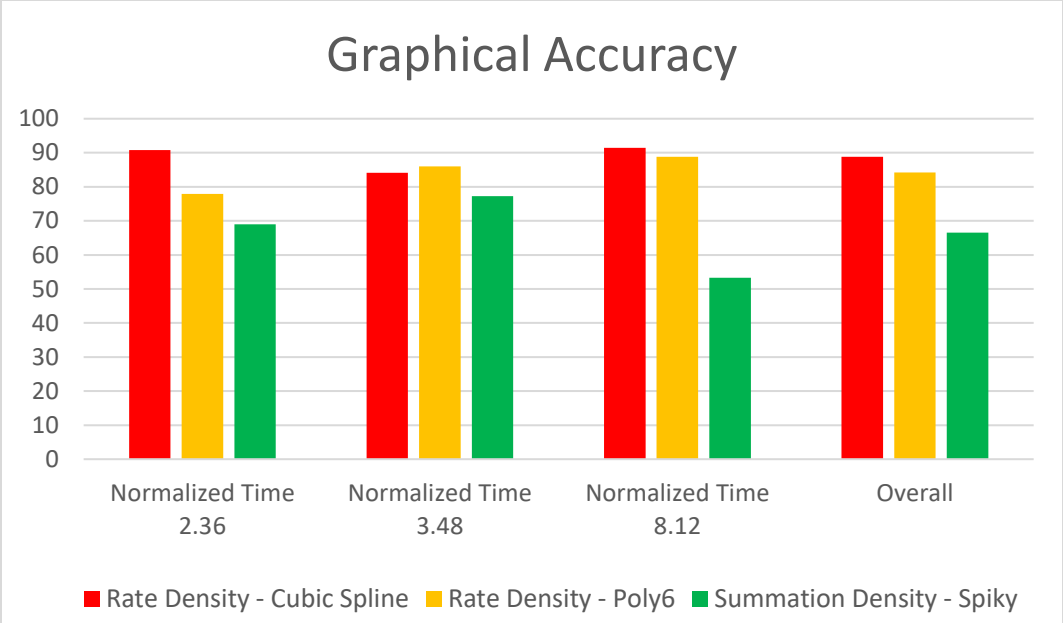


Figure 55 Graphical accuracy for the first validation example.

Concluding, after having considered all these aspects, it can be stated that the optimal solver for the first example is the Rate Density solver, with the implementation of the Cubic Spline kernel, because it remarkably retrieves the most accurate graphical results ensuring at the same time an excellent accuracy for what concerns the numerical results, even though its computational cost is higher, as it has been highlighted in Table 8.

Table 8 Summary of solvers' results (first example).

	Rate Density		Summation Density
Kernel Function	Cubic Spline	Poly6	Spiky
Numerical Accuracy	Excellent	Excellent	Excellent
Graphical Accuracy	Good	Good	Satisfactory
Splashing Particle Generation	Excellent	Satisfactory	Poor
Velocity Deviation	Good	Good	Satisfactory
Computational Effort	Satisfactory	Good	Good

Table 9 Accuracy Legend.

Accuracy	Grade
Poor	$60\% \leq Acc$
Satisfactory	$60\% < Acc \leq 75\%$
Good	$75\% < Acc \leq 90\%$
Excellent	$90\% < Acc \leq 100\%$

4.4.4 Mesh Sensitivity

In this section, a sensitivity analysis was conducted, as depicted in Figure 56, to explore how varying mesh density affects our numerical outcomes. The graph illustrates the investigation of three distinct mesh sizes:

- 4 cm: This configuration results in 1024 particles. Despite its low particle count, this mesh proves to be the least computationally demanding among the three. However, its lower particle count compromises the accuracy of our numerical predictions, particularly in velocity.
- 3 cm: Utilizing a finer mesh with 2000 particles enhances result accuracy compared to the 4 cm mesh. Notably, velocity prediction sees a significant improvement. Yet, doubling the particle count from the previous mesh leads to a substantial increase in computational time, escalating from 2.5 hours to 10 hours.
- 2 cm: Employing an even finer mesh with 4394 particles substantially impacts computational time, rising to 28.5 hours. Nevertheless, this mesh yields the most precise results among the three simulations, especially evident in velocity prediction.

Upon analyzing all three mesh sizes, we can discern the optimal choice. Considering both accuracy and computational efficiency, the 3 cm mesh emerges as the optimal selection. It offers results closely resembling the most refined mesh while maintaining a reasonable computational duration.

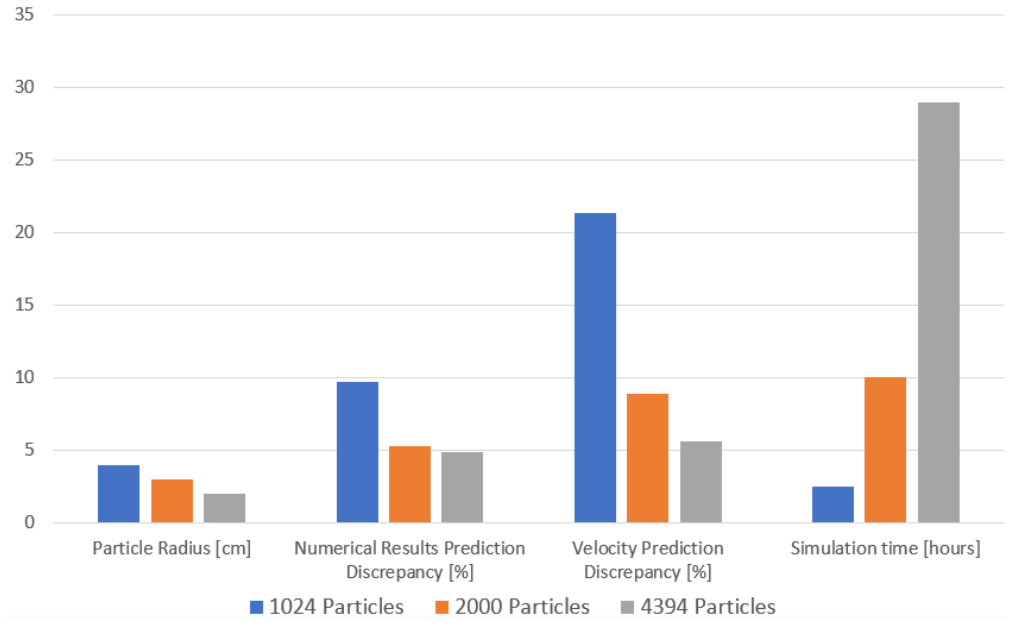


Figure 56 Mesh sensitivity analysis.

4.5 Dam break with a ramp.

In the previous section a simple dam break problem has been analyzed. Here through the addition of an obstacle, more specifically a ramp, it will be presented and investigated a more complex dam break problem. The presence of the obstacle allows to evaluate the quality of the developed SPH solver when the particles are not impacting against a vertical wall.

Looking at the schematic of the problem, which is shown in Figure 57, the model is a cube, the side length of the cube, L, is equal to 25 m. This means that the ramp is located at $x = 50\ m$. From the side view the ramp is an isosceles right triangle, with a cathetus length of 2.5 m. Moreover, with respect to the previous example, the right vertical wall has been removed; therefore, there is not an upper limit value for the x coordinate. Finally, the simulation time is equal to 6 seconds.

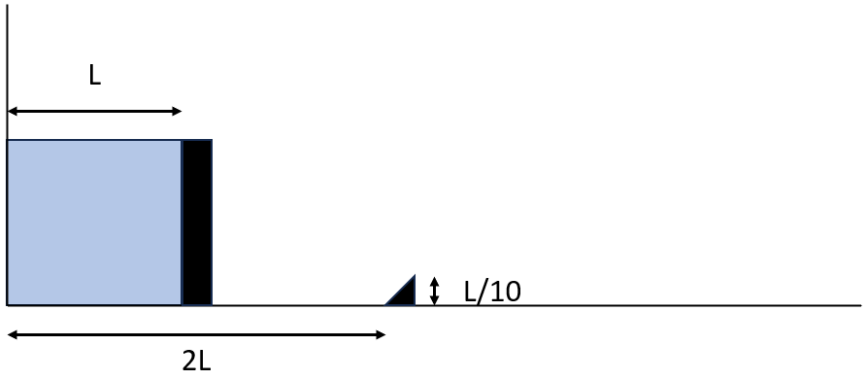


Figure 57 System representation (side view).

To perform the simulation, the system has been modeled in LS Pre- Post-, through the option "SPH Generation", obtaining the discretized model which is represented in Figure 58. It presents 8000 particles which are distributed as follows: 20 in x direction, 20 in y direction and 20 in z direction. From the number of the particles and the model dimensions it is possible to identify the particle length. The maximum velocity of particles throughout the simulation is capped at 40 m/s, resulting in an artificial speed of sound (Equation (2-60)) of 400 m/s. Utilizing the Lewy-Courant condition, the time step for the Rate Density solver can be determined, resulting in a value of $1.5511 \cdot 10^{-4}$ s, thereby establishing a total of 38682 time steps for the entire simulation, as outlined in Table 10. In contrast, the Density Summation solver employs the Spiky kernel, which features a support domain half that of the Cubic Spline utilized in the Rate Density approach. To ensure equitable comparison, a reduction factor (*f*) has been applied to the time step of the Density Summation solver, resulting in a time step of $3.1023 \cdot 10^{-5}$ s, necessitating a total of 193400 time steps for the entire simulation.

The ramp has been discretized through the utilization of the repulsive particles. They have the same size of the model particles (refer to "Boundary Conditions" section).

To reduce the required computational time of the simulation, it has been decided to exploit the symmetry condition of the problem in the y direction. This means that in $y = 12.5\,m$, a symmetry plane has been positioned (displayed in white in Figure 58), so, the particles that are above this plane will not be taken into account during the simulation. Therefore, it allows to perform the calculations for the half number of the particles, which reduces from 8000 to 4000.

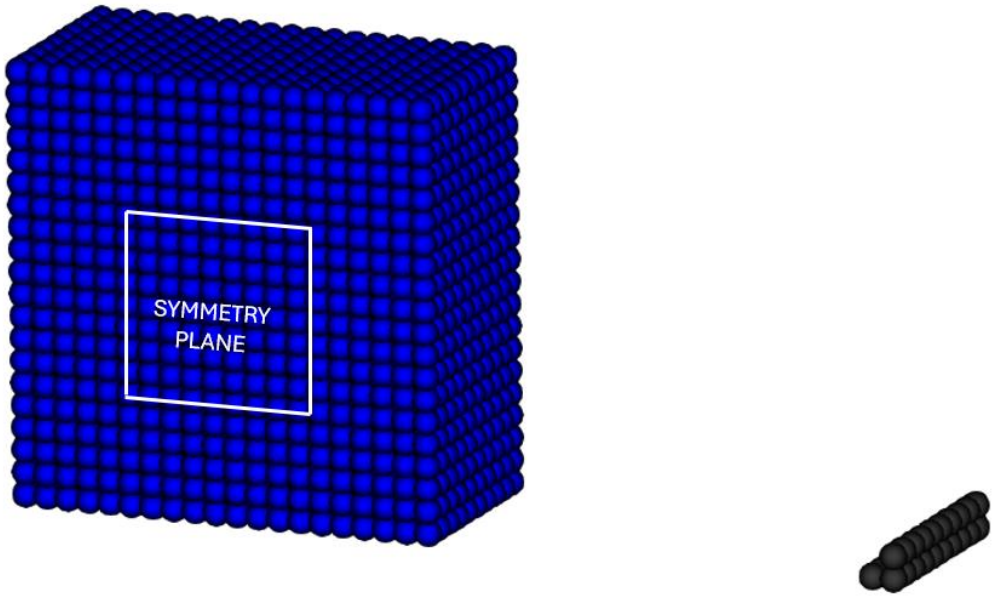


Figure 58 System discretized (isometric view).

Table 10 Simulation Parameters.

	Rate Density	Summation Density	Unit
Num of Particles	4000	4000	-
Kernel Function	Cubic Spline	Spiky	-
L	25	25	m
Simulation Duration	23	22.5	h
Time Step	0.15511	0.031023	ms
Num of Time Steps	38682	193400	-

4.5.1 Numerical Results

In this section the numerical results related to the second validation example are presented and discussed.

Starting the analysis from the mass conservation of the model, looking at Figure 59, it can be pointed out that the highest variation of the mass is at 1.85 s with a peak value of 7819191 kg, which defines a percentage variation with respect to the initial value of 0.086 %. This outlines that the mass is conserved along the simulation.

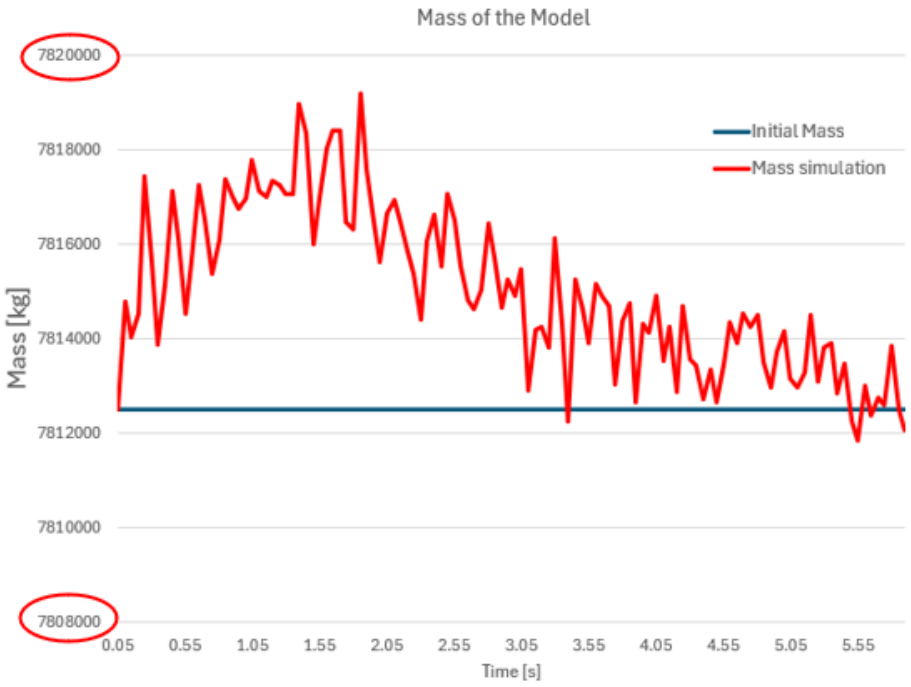


Figure 59 Evolution of the mass model along the simulation.

The analysis continues with the analytical results shown at Figure 60, where the parameter that evaluates the ratio between the actual height of the water with respect to the initial one is presented (HT/HT_0).

For what concerns the Rate Density solver outcomes, the first look at the graph reveals that the obtained values are compliant and close to Monaghan's simulation results and to the experimental data (J. Monaghan 1994). In particular, the first point at normalized time 0.71 matches exactly the experimental data. Furthermore, the slope between the two curves differs due the point at normalized time 1.39, which, being slightly further from the experimental data, causes the curve to be slightly shifted up. In the subsequent section, the slope is the same as Monaghan's simulation and therefore that slight difference between the two curves is kept.

Moving on the results related to the Summation Density solver, the first observation that can be highlighted from the graph is that the slope of the curve is not constant. In particular, between the time instants 0.71 and 1.39, the curve slope reduces, this means that the particles belonging to the upper layers of the model are not falling down with a sufficient velocity. Moreover, this is also underlined by the intersection between the two simulation curves, outlining a worse behavior of our solver with respect to Monaghan one. Furthermore, it can be stated that in general, the results present a significant discrepancy with respect to the experimental data (J. Monaghan 1994), especially in the last quarter of the simulation, where it is also possible to highlight that there is a considerable difference in the slope between the curves obtained by Monaghan and the present simulation, showing a diverging behavior. Therefore, in z direction the solver demonstrates to struggle to follow the evolution of the experimental data along the normalized time.

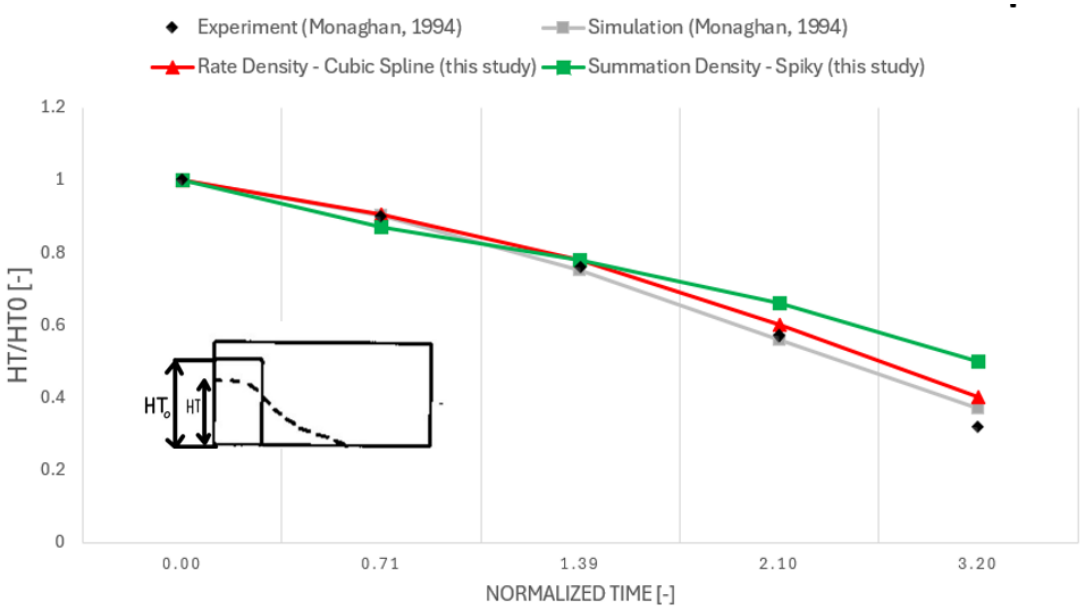


Figure 60 Simulation Results: ratio between the actual height of water and the initial one.

Consider the ratio between the actual front surge of water and the initial one (SR/SR_0), which is presented in Figure 61.

Starting from the Rate Density solver, it can be outlined that each point of the results of this study present a low discrepancy with respect to the experimental data. In particular, the highest difference is noted at normalized time 3.20, where the simulation retrieves a value of 4.48 against 3.80 detected by the experiment. Moreover, making a comparison with Monaghan simulation, it is possible to state that each result, defined in this study simulation, is more accurate and closer to the experimental data.

Focusing on the slope of the curve between the second and third points, located at normalized times 1.39 and 2.10 respectively, reveals a discrepancy with the experimental data. This suggests a slowdown in the advancement of the front surge. The primary cause is attributed to the coarse discretization of the ramp. In contrast, Monaghan's simulation doesn't exhibit this phenomenon due to its finer discretization with 50 particles in both the x and z directions, taking advantage of the simulation's 2-D nature for precise discretization. Consequently, even though the ramp profile is discretized with repulsive particles, it closely resembles a triangular profile, facilitating particle traversal over the obstacle. However, considering the particle motion in the third dimension significantly increases computational time, imposing limitations on the maximum particle count.

Furthermore, as highlighted by (J. Monaghan 1994), the little discrepancy at normalized time 3.20, between the simulation and the experimental results, is likely generated by the lack of the implementation of friction between the particles and the ground, leading to velocity values that are slightly higher with respect to the experiment.

Moving on the Summation Density solver results, it can be pointed out that for this second analytical result the solver is able to retrieve values that are compliant with the experimental data. In particular each point belonging to the present simulation curve is closer to the experimental data than the ones belonging to Monaghan simulation curve. A more detailed look at the graph outlines that for the time instant 0.71 the simulation is able to match exactly the experimental data. Finally, the last consideration can regard the slope of the curve, which is quite constant until the 2.10 time instant, because in the last quarter of the simulation the curves deviate a bit with respect to the evolution of the experimental data. This means that the velocity of the particles is a little higher than the velocity of water in the experiment, as for the Rate Density solver. However, it is possible to assert that for what concerns the results in the x direction, the solver works correctly retrieving satisfying values.

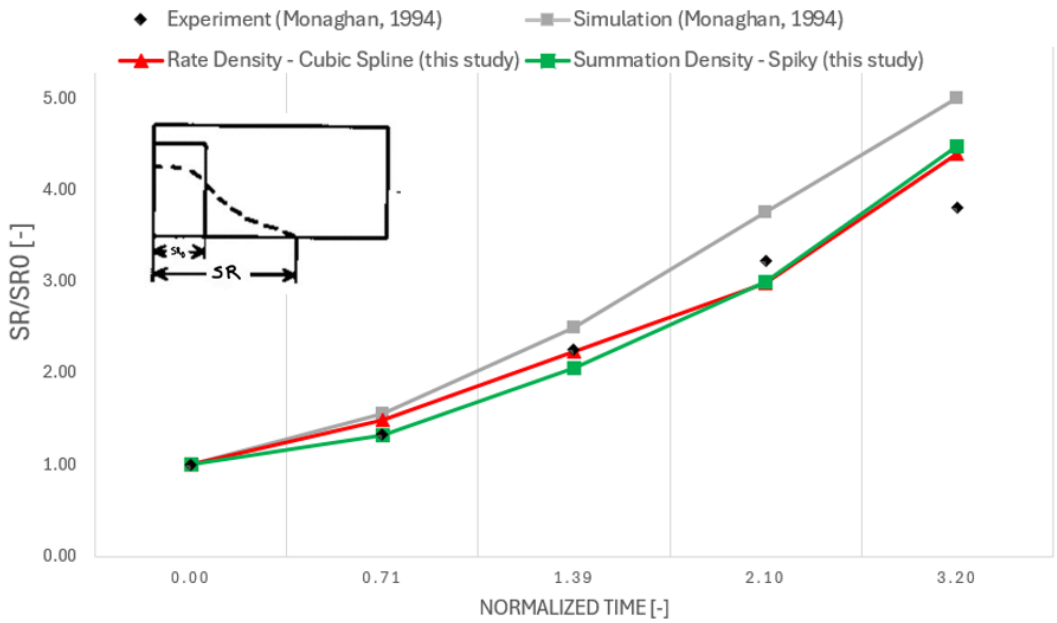


Figure 61 Simulation Results: ratio between the actual front surge of water and the initial one.

4.5.2 Graphical Results

In this section the graphical results, from Figure 62 to Figure 66, related to the second validation example are presented and discussed.

The analysis begins with the graphical results belonging to the Rate Density solver simulation. Starting from Figure 62-A, the particles present a correct descending movement, the shape is smooth and there are no oscillations on the free surface. Then, in Figure 63-A, it is possible to observe what has been explained in the previous section: from the side view the triangular ramp is discretized with 3 boundary particles. Therefore, the first water particles, that are approaching the ramp, struggle to overcome the obstacle. However, considering Figure 64-A, when the particles, belonging to the higher layers of the front surge, are overcoming the ramp, they are able to bring with them the particles of the lower layers. This results in a correct reproduction of the fluid motion against the ramp, even if the discretization is quite rough, compared to Monaghan simulation. Furthermore, the correctness of the SPH solver is also outlined by the fact that, during the entire simulation all the particles are moving together, this means that is no separation in the particle flow, but especially, there is no splashing particle generation after the impact against the obstacle. Finally, looking at the particles close to the left boundary in Figure 64-A and Figure 65-A, it can be highlighted a phenomenon that was also present in the simulation of the previous example. The particles of the higher layers of the model, when they are close to the boundary, they struggle to fall; in fact, even if

this involves only ten/fifteen particles, it defines a flow profile that is not totally smooth. However, looking all the other screenshots of the simulation, where this phenomenon is not present, it can be pointed out that it will appear only in the first instants when the last particles of the upper layer are starting to fall down.

Focusing on the graphical results related to the Summation density solver, from Figure 62-B, it can be highlighted that, even if the particles, in the lower-right corner of the model, are moving in the correct direction, defining a correct profile of the model, they show a low capacity to move together. This is an observation that is quite important, because if the particles present this behavior, when they will move through the obstacle, it can be expected that there will be a consistent splash generation. Advancing to the next simulation frame, showed in Figure 63-B, here it possible to observe that the model shape is correct and close to the expected one, even though few particles are oscillating on the free surface. Focusing on Figure 64-B, here it is displayed the first instants after the impact of the fluid particles against the ramp. The first thing that can be pointed out is that, as expected, after the impact there is a remarkable splash generation, with water particles that are moving basically in all the directions. Nevertheless, the majority of the particles are able to overcome the obstacle staying together, defining a concreate particle flow. In fact, in the continuing of the simulation (Figure 65-B and Figure 66-B), this particle flow is still recognizable, proving the effectiveness of the solver, even if this approach leads to a consistent generation of splashes. Furthermore, considering the last three simulation frames (from Figure 64- B to Figure 66-B), it is possible to observe two important aspects:

- The particles are bond on the left vertical boundary, which is the same phenomenon analyzed for the previous example.
- Taking into account the particles before the ramp, they define a model shape which is not so smooth, because of the fact that the particles on the free surface present an oscillating behavior.

Concluding, the solver is able to reproduce the motion of a water flow against a ramp, even though the outcomes are influenced by a consistent generation of splashes.

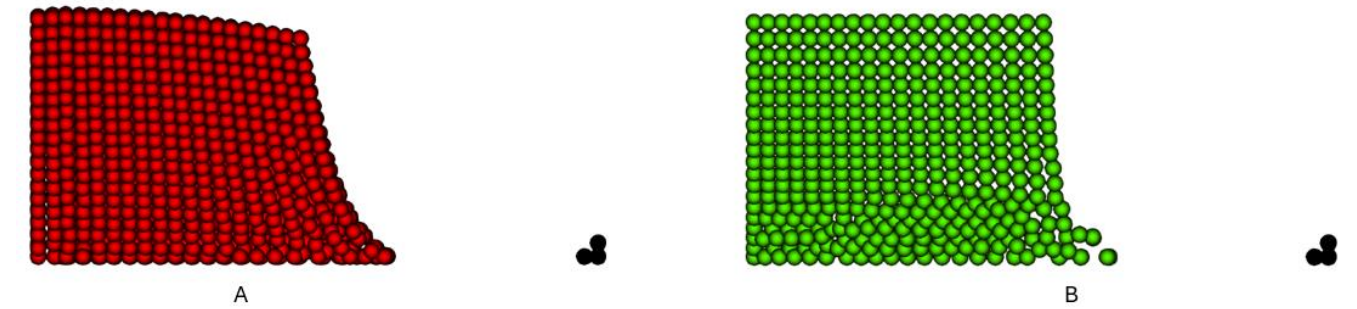


Figure 62 Simulation at 1 s (A- Rate Density solver, B- Summation Density solver).

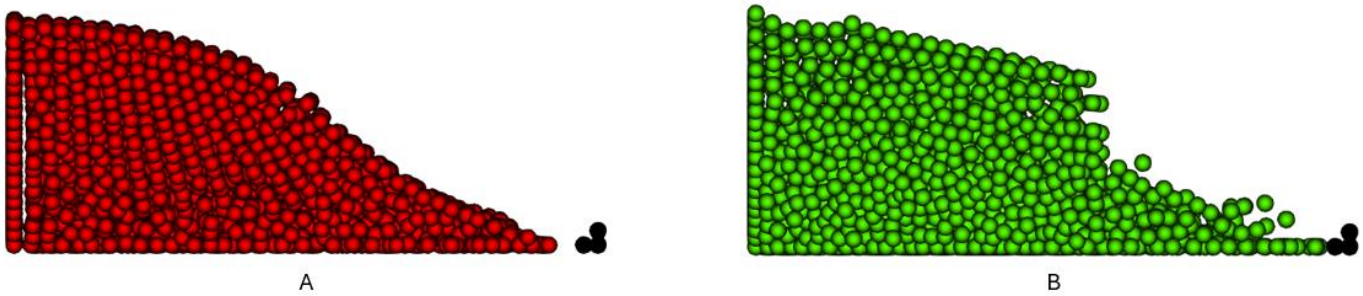


Figure 63 Simulation at 2 s (A- Rate Density solver, B- Summation Density solver).

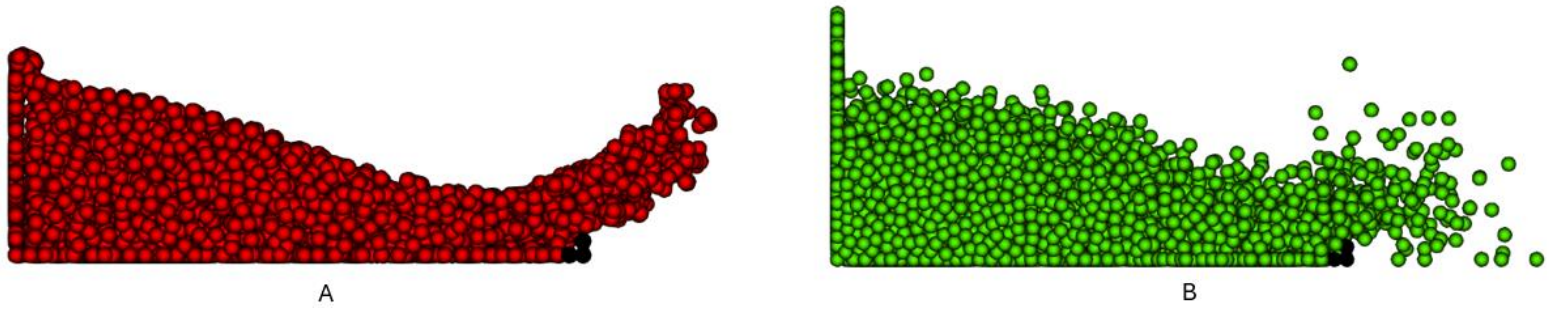


Figure 64 Simulation at 3 s (A- Rate Density solver, B- Summation Density solver).

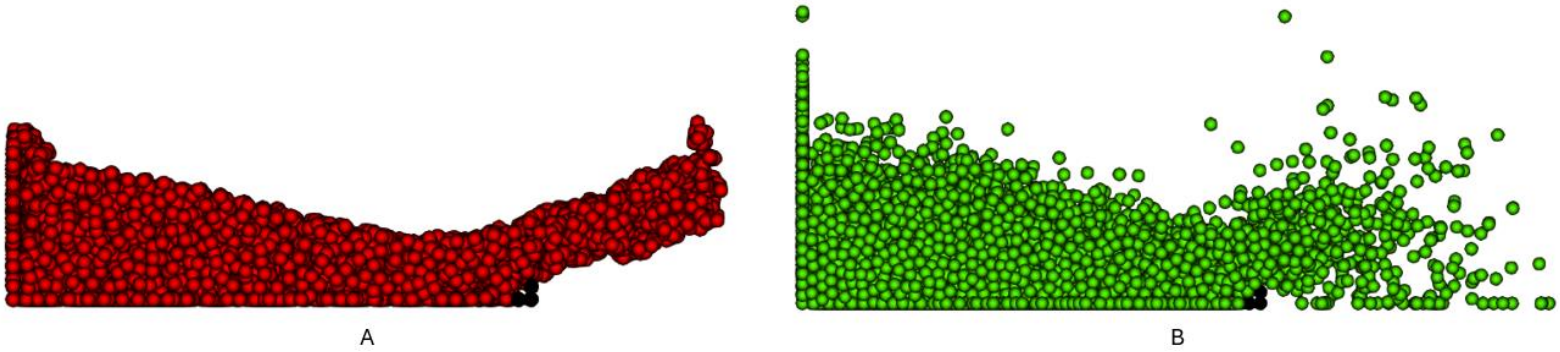


Figure 65 Simulation at 4 s (A- Rate Density solver, B- Summation Density solver).

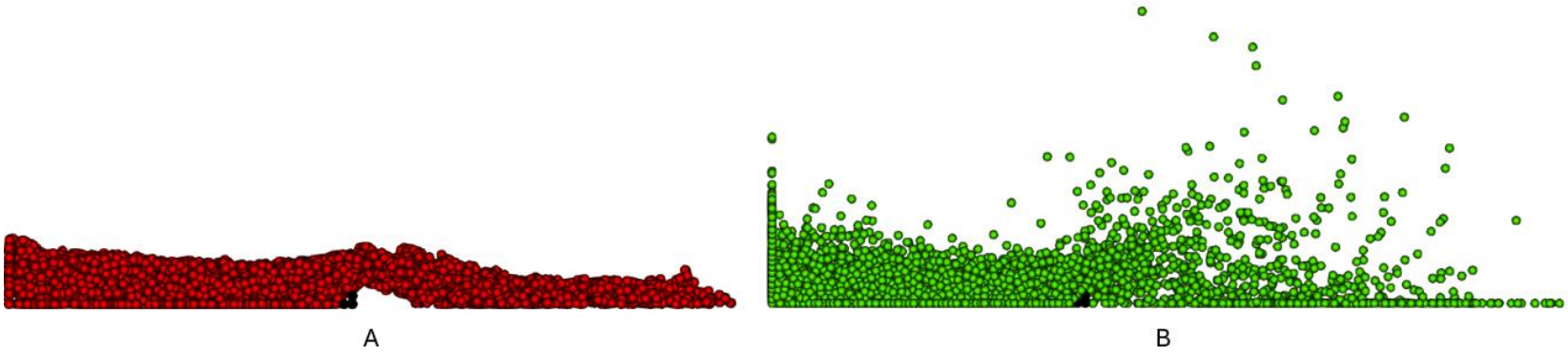


Figure 66 Simulation at 5 s (A- Rate Density solver, B- Summation Density solver).

4.5.3 Comparison of the solvers

To evaluate which solver is able to simulate in the most faithful way the water motion against the ramp, it is necessary to compare the analytical and the graphical results.

Starting from the analytical outcomes, it appears evident that, for what concerns the evolution of the height of the water along the simulation (HT/HT_0), which is reported in Figure 60, the solver that retrieves the best results is the one based on the Rate Density approach, because the slope of curve results is constant and for all the time instants the discrepancy with respect to the experimental data is very low. For the second parameter, the one that shows the advancing of the front surge of the water (SR/SR_0), noted in Figure 61, the results obtained from both simulations are very similar: both are able to match twice the experimental data and the slope is nearly constant along the entire simulation. For both in the last quarter of the simulation, there is a rise in the values, defining a little but tangible discrepancy with the last experimental data. Therefore, according to this parameter the solvers perform at the same level in reproducing the water flow motion.

Moving on the graphical results, the comparison appears more complex. Firstly, before the impact, both solvers are able to define a correct particle motion, that outlines a precise evolution of the model shape from the side view, showed in Figure 62 - Figure 63. The only slight difference is in the smoother free surface particle flow with the implementation of the Rate Density approach. Advancing to the frame showing the instants after the impact with the ramp (Figure 64), here it is possible to outline a more compact movement behavior with the implementation of the Rate Density solver, even though, the motion defined by the Summation Density solver is also compliant with the physical movement of the water. For what concerns the last two frames (Figure 65 - Figure 66), here a more significant difference is outlined between the two solvers. In fact, the Rate Density solver allows to obtain a particle stream that is still compact, so the particles are effectively moving together, and at the same time it totally neglects the generation of splashes.

Furthermore, taking into account the interaction of the particles with the left vertical boundary, the implementation of the Rate Density solver leads to a more physical motion of the particles, which are still influenced by the presence of the boundary, but they are able to move far from it avoiding to remain bond on it for the entire duration of the simulation, that is the issue of the Summation Density Solver.

Finally, considering the comparison of the analytical and graphical results, as outlined in Table 11, it is possible to point out that the solver that retrieves the best outcomes for this second example is the Rate Density Solver, due to its capability to define a very low difference between the simulation results and the experimental data, and at the same time it outlines a smooth and compact particle motion for the entire duration of the simulation.

Table 11 Summary of solvers' results (second example).

Solver	Rate Density	Summation Density
Numerical Accuracy	Excellent	Satisfactory
Splashing Particle Generation	Excellent	Poor
Velocity Deviation	Good	Satisfactory
Computational Effort	Satisfactory	Good

Table 12 Accuracy Legend.

Accuracy	Grade
Poor	$60\% \leq Acc$
Satisfactory	$60\% < Acc \leq 75\%$
Good	$75\% < Acc \leq 90\%$
Excellent	$90\% < Acc \leq 100\%$

4.6 Fall of a droplet in water.

The third and last validation example regards the fall of a drop into a glass of water.

The set up of the example is shown at Figure 67. The glass, from the side view, presents a rectangular shape whose length and height are respectively 0.06 m and 0.04 m. The drop at the initial instant is located 5 mm above the free surface of the water, in the exact middle of the glass. It presents a diameter of 5 mm. It has an initial velocity which is equal to 2.0 m/s. For what concerns the maximum fluid velocity along the simulation, this value is set equal to 2.1 m/s. The duration of the experiment performed by DaMing (DaMing 2011), is 0.12 s, which also corresponds to the duration of the simulation.

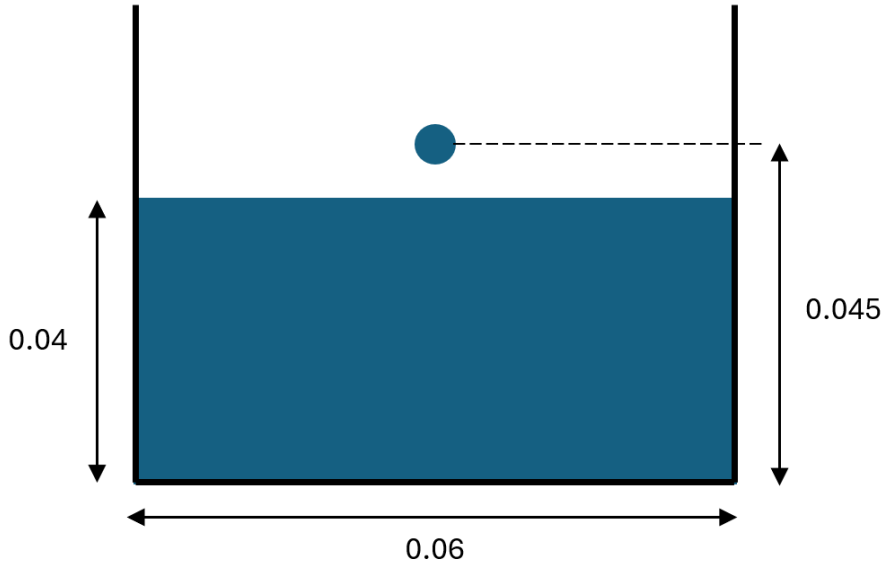


Figure 67 System Representation (dimensions expressed in meters).

To perform the simulation, the system has been modeled in LS Pre- Post-, through the option "SPH Generation", obtaining the discretized model which is represented in Figure 68. In this example for the Summation Density Solver, a model with the implementation of boundary particles has been generated (Figure 68-B), because in this case the density has not been normalized (refer to "Mass Conservation" section). Thus, the two models have the same number of moving particles. However, to reduce the computational time required by the simulation two symmetry conditions, in the x and y directions, have been exploited: at $x = 0.03\ m$ and at $y = 0.03\ m$, two symmetry planes have been set. In this way, the number of particles in both directions will be the half, leading to a total number of moving particles of 36034. For what concerns the Summation Density Solver, the additional 7224 boundary particles define a total number of particles equal to 43258.

The maximum velocity of particles throughout the simulation is capped at 2.1 m/s, resulting in an artificial speed of sound (Equation (2-60)) of 21 m/s. Utilizing the Lewy-Courant condition, the time step for the Rate Density solver can be determined, resulting in a value of $1.318 \cdot 10^{-5}$ s, thereby establishing a total of 10154 time steps for the entire simulation, as outlined in Table 13. In contrast, the Density Summation presents a greater number of particles (due to the presence of the boundary particles), therefore the time step has been increased to get a similar duration for the simulation, resulting in a time step of $1.757 \cdot 10^{-5}$ s, necessitating a total of 7616 time steps for the entire simulation.

The validation of the solvers will be performed trough the comparison between the graphical results and the experiment pictures provided by (DaMing 2011). To facilitate this comparison, a section plane has been established at $y = 0.024\, m$ to provide a clearer depiction of the water's movement.

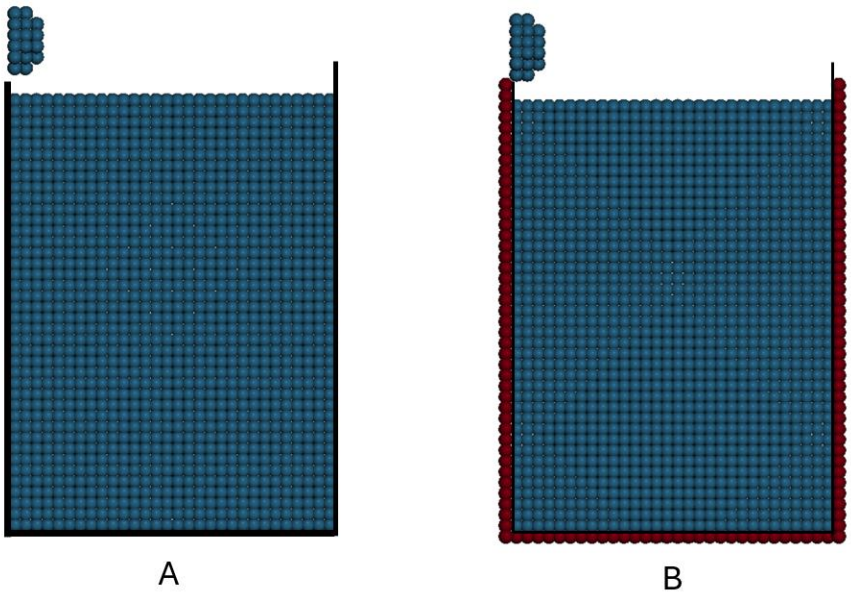


Figure 68 System Discretization (A without boundary particle implementation, B with boundary particle implementation).

Table 13 Simulation Parameters.

	Rate Density	Density Summation	Unit
Num of Moving Particles	36034	36034	-
Num of Fixed Particles	-	7224	-
Kernel Function	Cubic Spline	Poly6	-
Simulation Duration	57	53	h
Time Step	0.01318	0.01757	ms
Num of Time Steps	10154	7616	-

4.6.1 Numerical Results

For this third and last validation example, the only numerical result that will be presented and discussed is the evolution of the mass model along the simulation, shown in Figure 69. It can be stated that the mass is conserved, because the highest variation with respect to the initial mass is at time 0.012 s, with a percentage difference of 0.027 %.

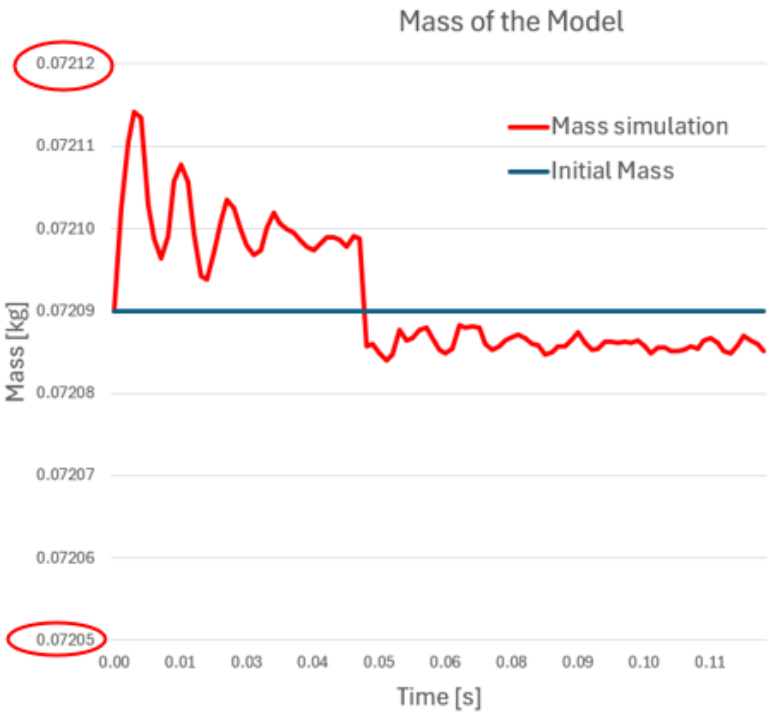


Figure 69 Evolution of the mass model along the simulation.

4.6.2 Graphical Results

In this section the graphical results related to the third validation example are presented and discussed.

Starting the analysis from the first-time step, illustrated in Figure 70, for both solvers, the impact of the drop on the free surface accurately initiates particle motion, clearly outlining a valley. The calculated discrepancies compared to the experimental images are 10.15% for the Rate Density solver and 5.96% for the Summation Density solver. However, a closer examination of Figure 70-C shows that while the Summation Density solver is more accurate along the entire free surface, the impact of the drop produces a few splashing particles. In contrast, the Rate Density simulation, depicted in Figure 70-B, displays a slightly uneven profile at the sides of the depression, caused by some particles oscillating on the water's surface.

Considering the next time step, shown in Figure 71, we start with the Summation Density outcomes (Figure 71-C). Initially, we observe significant generation of splashing particles on the free surface. However, a

more detailed look at the picture reveals that the particles closely replicate the profile captured in the experimental image: descending in height from the side walls towards the center, forming gentle valleys. Additionally, at the center of the glass, a cluster of particles is moving upward along the z-axis, effectively simulating the formation of a water gush. The quality of the graphical result is highlighted by the calculated discrepancy of 7.38% for the Summation Density solver.

On the other hand, the Rate Density solver, as shown in Figure 71-B, produces qualitatively poorer results compared to the Summation Density solver. Although the particles at the bottom of the semicircular valley manage to reverse their motion and ascend, the shapes of the two lateral valleys are not completely smooth, despite their lengths being comparable to those in the experimental image. This leads to a higher discrepancy of 18.71%, more than double that of the Summation Density solver.

Advancing to the third time step, illustrated in Figure 72, both solvers demonstrate that the cluster of particles accurately represents the water gush, now reaching its peak height. Concurrently, two distinct valleys are observable from the sides to the base of the water gush. These features bring the simulation results remarkably close to the experimental data.

Examining in more detail, the quality of the Summation Density (Figure 72-C) lies in its reproduction of the water gush, with a height that closely matches the experimental data. However, near the glass's side and close to the water gush, there is still the generation of splashing particles. In contrast, the Rate Density solver (Figure 72-B) shows lower accuracy in reproducing the water gush but excels in accurately reproducing the lateral valleys, whose shape and extent perfectly match the experimental outcomes (Figure 72-A).

Calculating the discrepancy, the Rate Density solver produces results closer to the experimental picture, with a discrepancy of 7.47%, compared to the Summation Density's discrepancy of 9.24%.

Finally, examining the last time step in Figure 73, both solvers show that the particles are still settling into their rest positions, revealing a slight discrepancy compared to the experimental result (Figure 73-A). However, the Summation Density solver (Figure 73-C) is closer to the experimental result, producing a relatively uniform free surface. This frame, though, is affected by splashing particles generated in the previous time step, which are still falling. The calculated discrepancy for the Summation Density solver is 9.02%.

In contrast, the Rate Density solver, despite not generating splashes in this frame, produces a rougher simulation outcome. This is due to the difficulty of the particles in the middle of the glass in reaching their rest positions. This issue significantly impacts the calculation of the discrepancy, which is 15.63%.

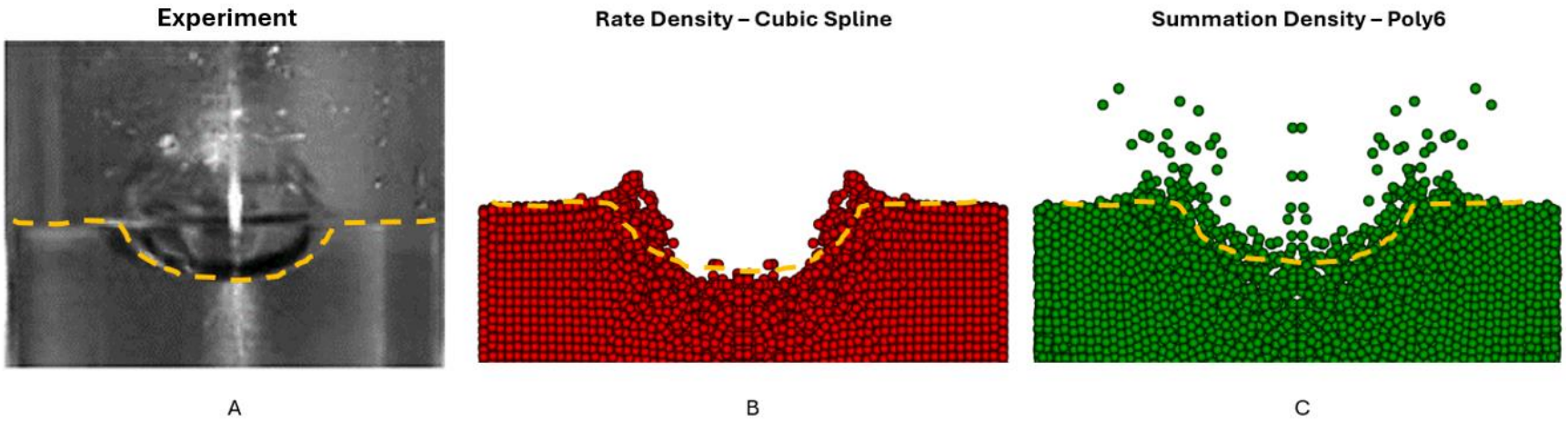


Figure 70 Results at 0.015 s (-A Experiment frame (DaMing 2011), -B Simulation frame (Rate Density), -C Simulation frame (Summation Density)).

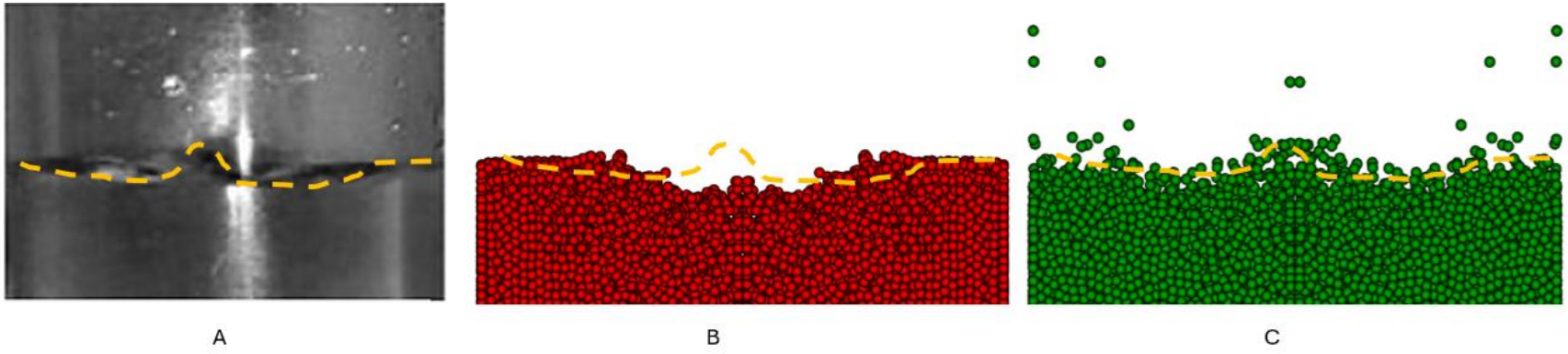


Figure 71 Results at 0.05 s (-A Experiment frame (DaMing 2011), -B Simulation frame (Rate Density), -C Simulation frame (Summation Density)).

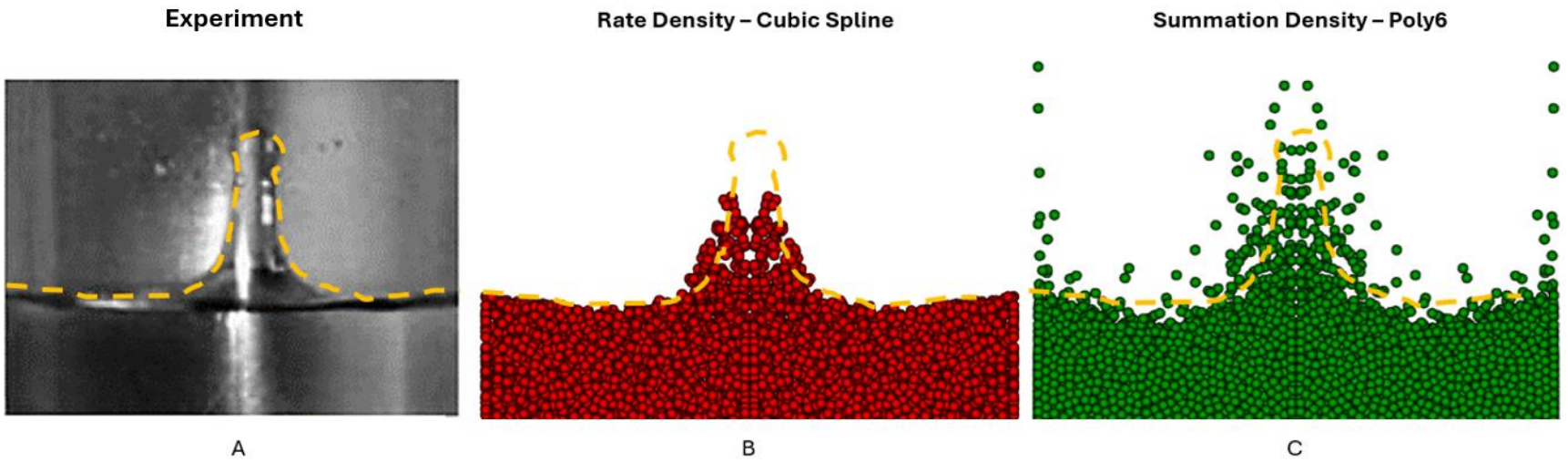


Figure 72 Results at 0.08 s (-A Experiment frame (DaMing 2011), -B Simulation frame (Rate Density), -C Simulation frame (Summation Density)).

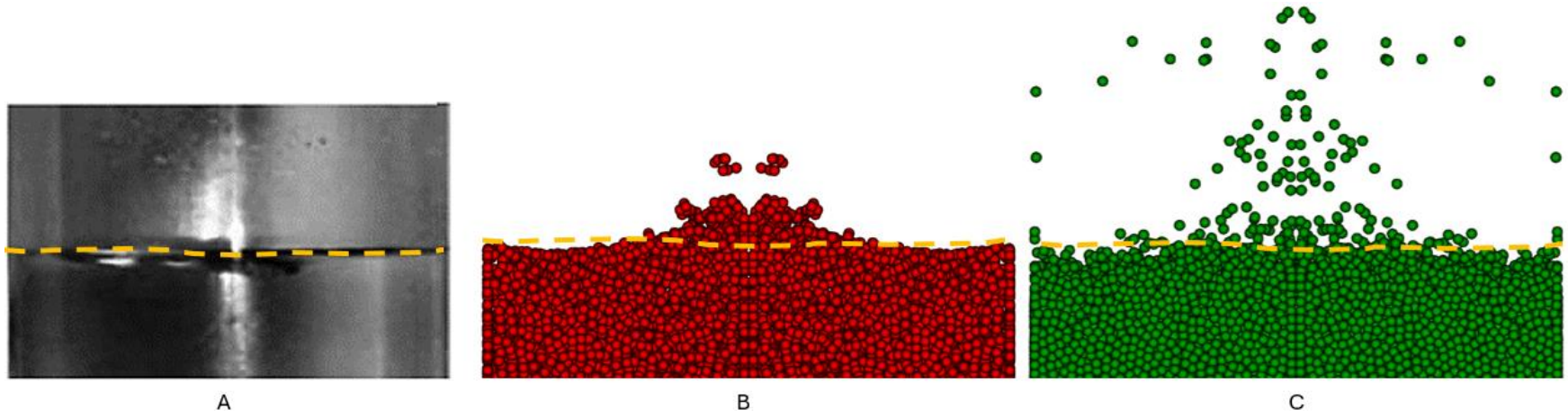


Figure 73 Results at 0.12 s (-A Experiment frame (DaMing 2011), -B Simulation frame (Rate Density), -C Simulation frame (Summation Density)).

4.6.3 Comparison of the solvers

When comparing the simulation outcomes obtained by implementing both solvers in this third example, it becomes evident that the results are similar and comparable. More specifically, the Summation Density solver generally shows higher accuracy, with an accuracy rate of 92.10%, as reported in Figure 74. It better reproduces the water gush, achieving a height close to the experimental one. However, it also generates a considerable number of splashing particles.

The Rate Density solver, on the other hand, perfectly reproduces the semicircular valley following the drop impact and the two valleys on the sides of the water gush. Despite this, the height of the water gush is lower than expected, and there is a notable issue with the particles in the water gush having difficulty reaching their rest position, but it is able to completely neglect any splash generation. Consequently, the overall accuracy is 87.01%, as highlighted in Figure 74.

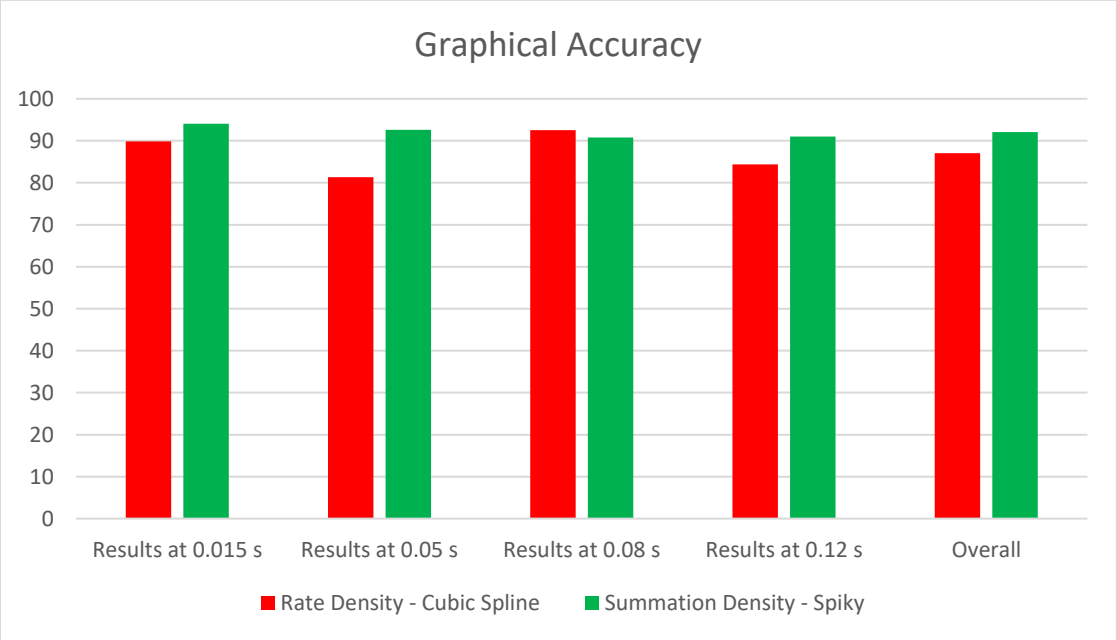


Figure 74 Graphical Accuracy for the third validation example.

In terms of computational effort, the Summation Density Solver requires over 7224 boundary particles, necessitating a reduced time step to achieve simulations of comparable duration.

Considering all these aspects, for the third and last validation example, the Rate Density solver retrieves the best outcomes. It shows accuracy close to that of the Summation Density solver while completely avoiding the generation of splashing particles.

Table 14 Summary of solvers' results (third example).

Solver	Rate Density	Summation Density
Graphical Accuracy	Good	Excellent
Splashing Particle Generation	Excellent	Poor
Computational Effort	Satisfactory	Good

Table 15 Accuracy Legend.

Accuracy	Grade
Poor	$60\% \leq Acc$
Satisfactory	$60\% < Acc \leq 75\%$
Good	$75\% < Acc \leq 90\%$
Excellent	$90\% < Acc \leq 100\%$

4.7 Application to an industrial problem

In this section the solver will be applied to investigate a problem strictly related to the automotive industry, which is the fuel sloshing in a vehicle tank (refer to the "The Sloshing Problem" section).

The set up of the problem (Rajagounder 2016), is shown in Figure 75. The tank has a length of 0.370 meters, the height is equal to 0.140 meters and the fuel inside occupies the 25% of the total available volume of the tank. In this analysis the fuel is replaced by the water and the tank undergoes a uniform acceleration in the negative direction of the x axis, this means that the liquid will perceive the acceleration in the opposite direction. This uniform acceleration is equal to 1.6 m/s and the experiment lasts for 1 s, as reported in Table 16.

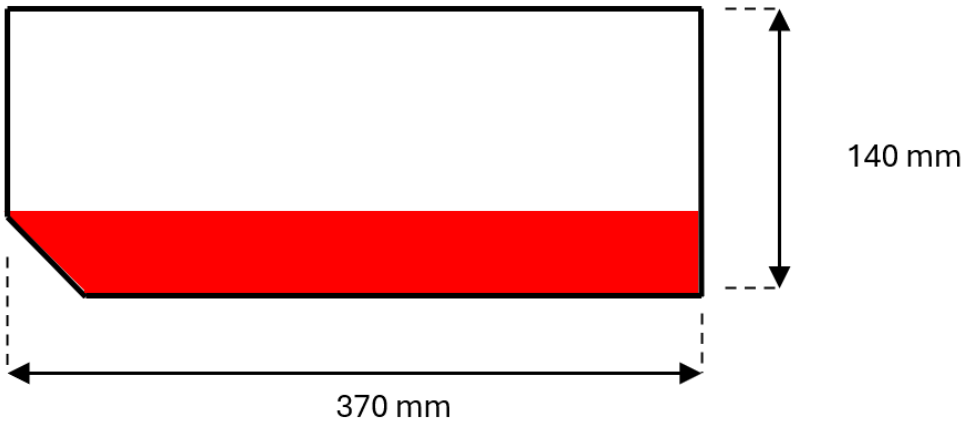


Figure 75 Tank model dimensions.

Table 16 Experiment Parameters.

		Unit
Liquid	Water	-
Liquid Volume	25%	-
Tank length	0.37	m
Acceleration	1.6	m/s^2
Experiment duration	1	s

In this study, two different set-ups (Rajagounder 2016), will be analyzed and discussed through the use of the developed SPH solver with the implementation of the Rate Density approach (Cubic Spline kernel):

- The first case represents the tank with no baffle implementation.
- The second case introduces two vertical baffles to reduce the fuel sloshing.

Finally, the developed SPH solver will be exploited to model an improved solution to obtain a further reduction of the fuel sloshing.

For what concerns the numerical results the analyzed factor is the ratio between the height of the fuel at the right boundary of the tank and the length of the tank (H/L). It will be evaluated along the entire simulation, where the time is normalized through the following normalization factor:

$$Normalization\ factor\ = \sqrt{\frac{a}{L}}$$

Where: a, is the acceleration to which the liquid is undergone; and L is the length of the tank.

4.8 Without baffles

The first case represents the tank in its initial configuration, without any wall implementation, as shown in Figure 76. This design, as discussed in the Literature Review in "The Sloshing Problem" paragraph, is expected to lead to a significant movement of the fuel inside of the tank.

Figure 76 Tank without walls real set up.

To perform the simulation, the system has been modeled in LS Pre- Post-, through the option "SPH Generation", obtaining the discretized model which is represented in Figure 77. Looking at Table 17, it is possible to outline that, the liquid inside of the tank has been discretized with 38656 moving particles. The walls have been designed exploiting the boundary particles, thus they are fixed in their position for the entire duration of the simulation, in particular to model the left and right boundaries 6769 boundary particles have been implemented.

Thus, the total number of particles reaches the value of 45245, defining a remarkable computational cost which sets the simulation time to 43 hours.

Finally, from the volume of the liquid and the number of particles has been possible to define their length. The maximum speed of the fluid inside of the tank has been set to 1.5 m/s, outlining an artificial speed of

sound (Equation (2-60)) equal to 15 m/s. Therefore, it has been possible to define the time step, through the CFL condition, which is 0.15305 milliseconds.

Table 17 Simulation parameters (no wall implementation).

Rate Density Approach		Unit
Moving Particles	38656	-
Fixed Particles	6769	-
Kernel function	Cubic Spline	-
Simulation time	43	hours
Time step	0.15305	ms

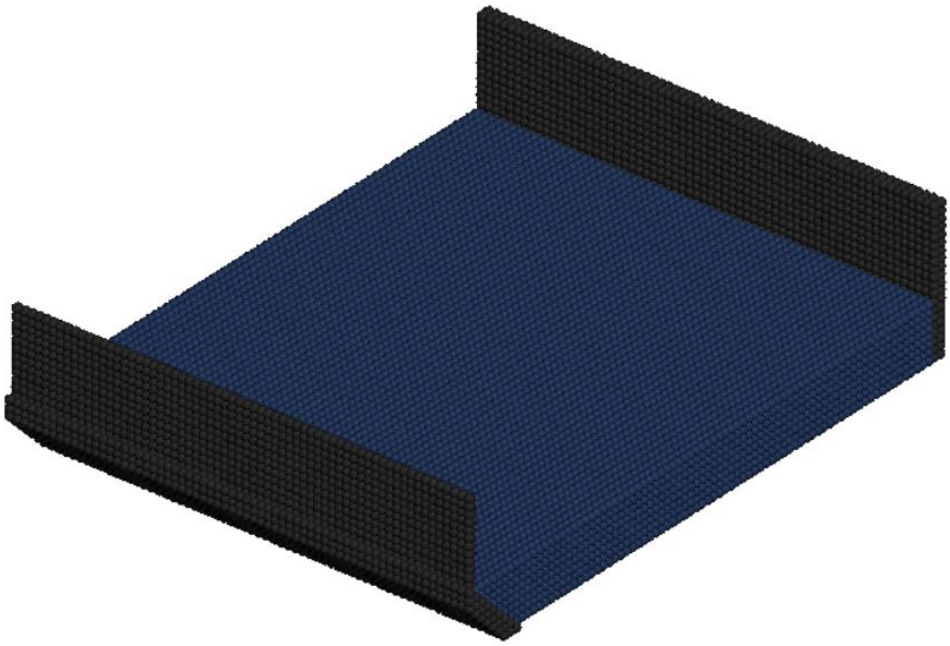


Figure 77 Tank without walls discretized set up (isometric view).

4.8.1 Numerical Results

In this section the numerical results related to the simulation involving the initial set up of the tank are presented and discussed.

The first result that is analyzed is the evolution of the mass model along the simulation, Figure 78. The highest variation is outlined at normalized time 0.42, where the difference with respect the initial value is equal to 0.070%. This ensures that the mass is conserved along the simulation.

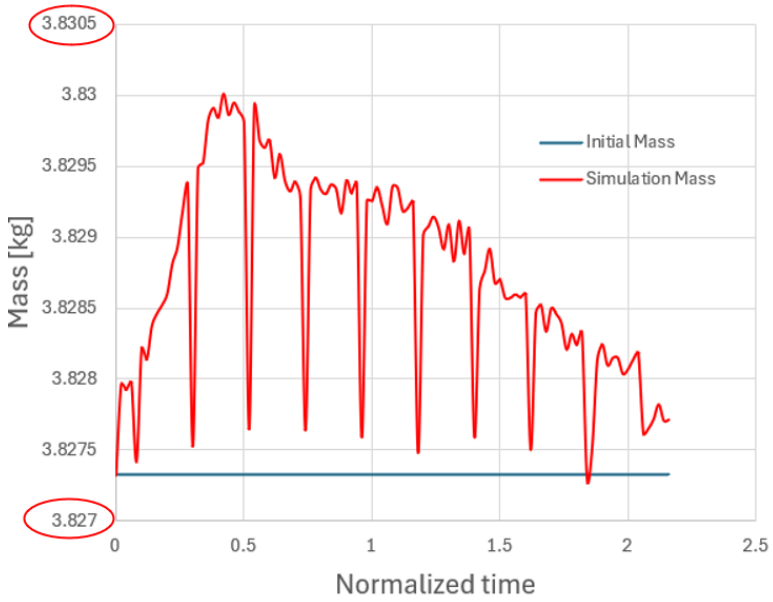


Figure 78 Evolution of the mass model along the simulation.

The analysis continues with the evolution of the ratio between the height of the free surface at the right boundary and the length of the tank (H/L), as shown in Figure 79. The graph demonstrates that the simulation results align with both the experimental data and the simulations conducted by Rajagounder (Rajagounder 2016). The greatest variation from the experimental data occurs at a normalized time of 0.927, with a discrepancy of 17%.

However, the graph shows that for more than half of the simulation time, the results are closer to the experimental data than those obtained by Rajagounder. Averaging the discrepancy over the entire simulation duration, the value reduces to 9.26%, highlighting the excellent accuracy of the solver in replicating the fluid movement within the tank.

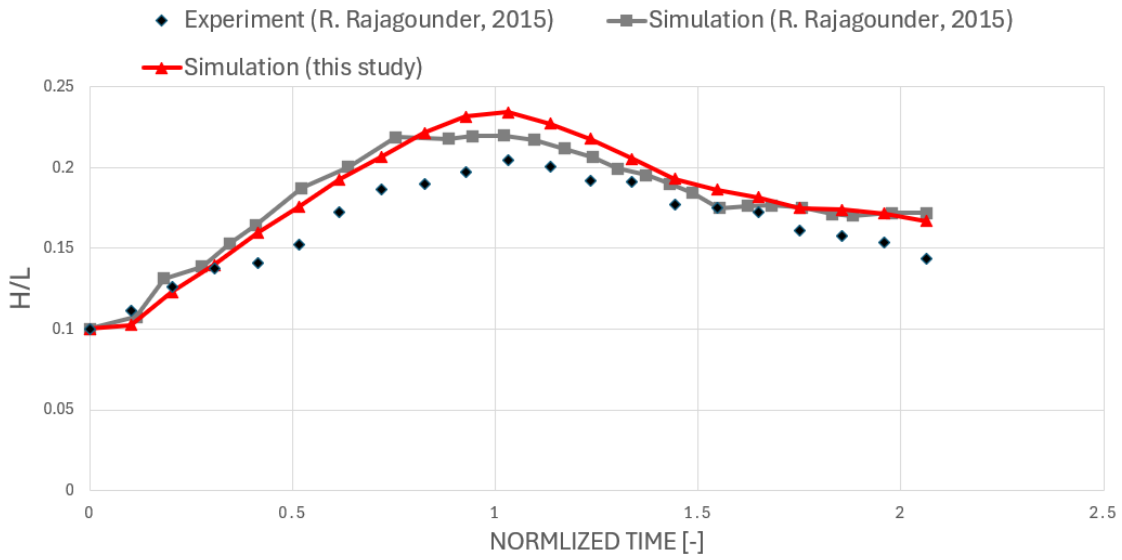


Figure 79 Evolution of the ratio between water height at the right boundary and the tank length (without baffle implementation).

4.8.2 Graphical Results

In this section the graphical results, Figure 80, are presented and discussed.

The comparison between the experiment (Figure 80-A) and the simulation (Figure 80-B) clearly shows the correct motion of the particles, which have moved from their initial rest position under uniform acceleration. Notably, no splashing particles are generated. Additionally, Figure 80-B indicates that there are no oscillations on the free surface, demonstrating smooth interaction between the particles.

Examining the discrepancy between the experimental free surface, marked by the yellow dotted line, and the simulated free surface reveals further insights. The two surfaces intersect around two-thirds of the tank's length. In the first two-thirds of the tank, the particles are below the yellow line, resulting in a discrepancy of 16.67%. In the last third, the particles are closer to or above the line, with a discrepancy of 7.91%. The weighted average of these discrepancies yields an overall value of 13.54%, indicating good accuracy of the solver from a graphical perspective.

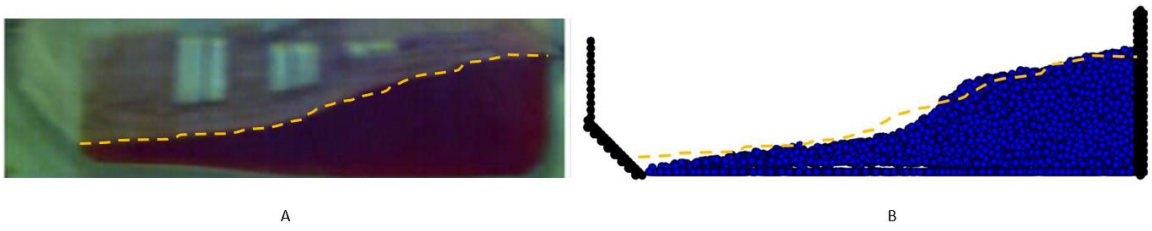


Figure 80 Results at 0.5s (A- Experiment (Rajagounder 2016), B- Simulation Rate Density solver).

4.9 Two baffles implementation

In this second scenario, two walls are placed equally spaced inside the tank, depicted in Figure 81. As shown in Table 18, the total number of particles has slightly increased to 50,728 due to the presence of these walls extending to the top of the tank. Despite this increase, the simulation time remains unchanged at 43 hours.

Regarding the time step evaluation, it remains consistent with the previous case. Although reducing the speed of sound could be considered due to the expected decrease in liquid sloshing, it was decided to keep it constant.

Table 18 Simulation parameters (two baffles implementation).

Rate Density Approach		Unit
Moving Particles	36608	-
Fixed Particles	13652	-
Kernel function	Cubic Spline	-
Simulation time	43	hours
Time step	0.15305	ms

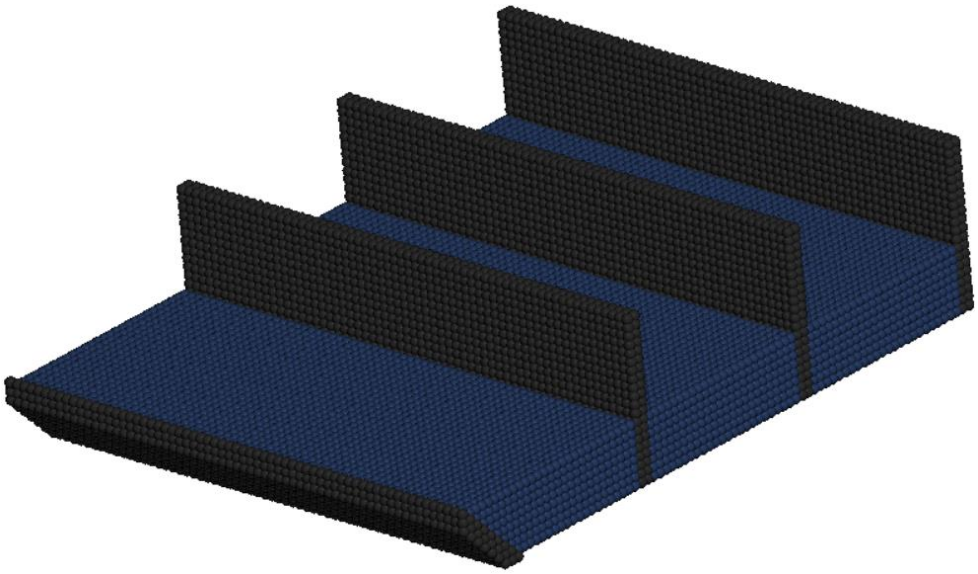


Figure 81 Tank with baffles discretized set up (isometric view).

4.9.1 Numerical Results

In the following the numerical results, related to the simulation with implementation of two baffles, are presented and discussed.

Starting from Figure 82, where is displayed the evolution of the model's mass along the simulation, the highest variation with respect the initial value is at normalized time 1.44, where the difference is 0.0326%. This ensures that the mass is conserved along the simulation.

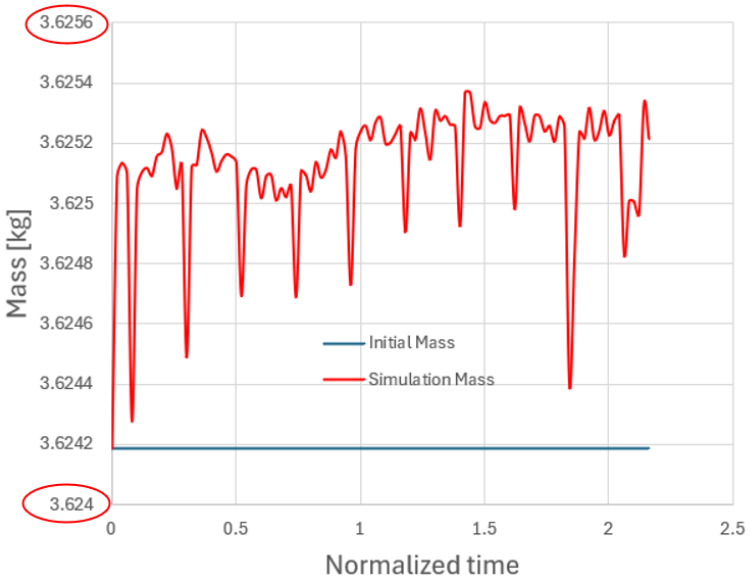


Figure 82 Evolution of the mass model along the simulation.

Advancing to Figure 83, which shows the evolution of the free surface height at the right boundary over the tank length (H/T), we observe that the simulation values align well with the experimental data. Furthermore, comparing these simulation outcomes with those from (Rajagounder 2016) reveals that in the first half of the simulation, the values from this study closely match Rajagounder's results, demonstrating a similar discrepancy relative to the experimental data, more specifically at normalized time 0.373 it is possible to outline the highest discrepancy which is equal to 15.13 %. In the second half, however, the values belonging to this study exhibit much higher accuracy, closely matching the experimental data.

Overall, the discrepancy with the experimental data is 5.49%, highlighting the excellent numerical accuracy of the Rate Density solver.

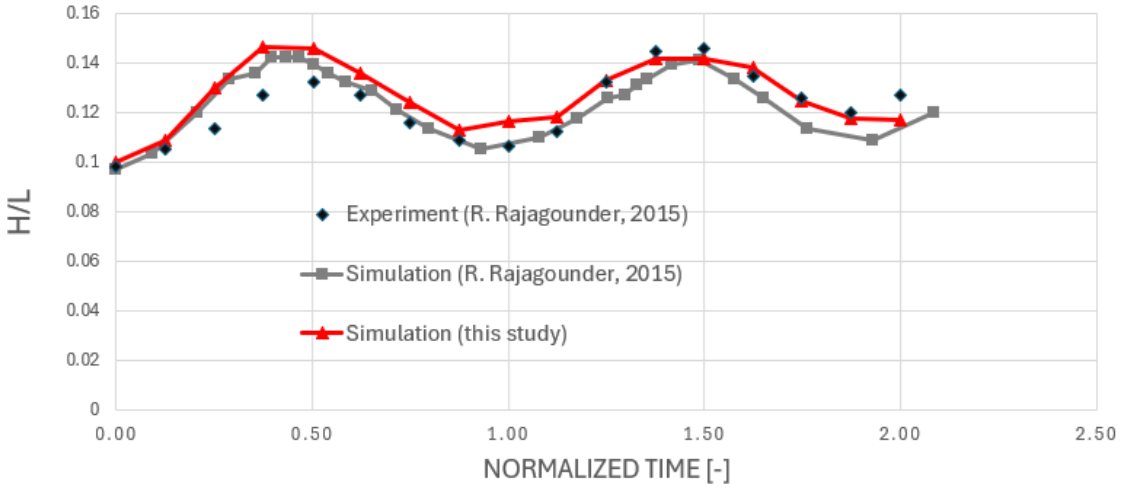


Figure 83 Evolution of the ratio between water height at the right boundary and the tank length (two baffles implementation).

4.9.2 Graphical Results

A first look at Figure 84, shows that the particles move correctly in the right direction under the influence of uniform acceleration. No splashes are generated, and there are no oscillations on the free surface. Thus, from a qualitative perspective, the solver accurately reproduces the motion of the particles.

The next step involves computing the discrepancy between the free surface observed in the experiment and the one simulated by the solver. For the three sections of the tank, starting from the left and moving to the right, the calculated discrepancies are 15.71%, 11.96%, and 11.54%, respectively. The overall discrepancy is 13.07%, highlighting the solver's good graphical accuracy and ensuring its correctness.

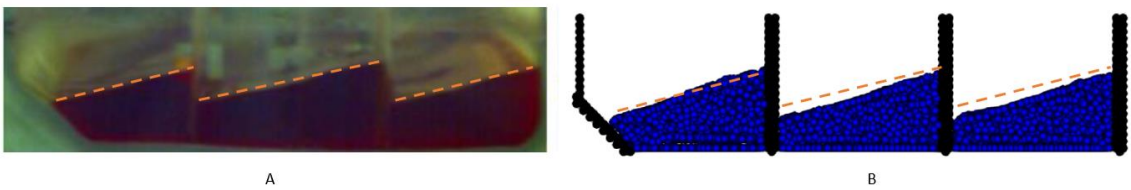


Figure 84 Results at 0.22 s (A- Experiment (Rajagounder 2016), B- Simulation Rate Density solver).

4.10 Proposed Solution

In this section is presented a new design solution to ensure a further suppression of the fuel sloshing.

Based on the results obtained with the solutions proposed by (Rajagounder 2016) and considering the baffle design analysis previously conducted (refer to “The Sloshing Problem” section), the proposed solution to suppress fuel sloshing in a rectangular tank is shown at Figure 85.

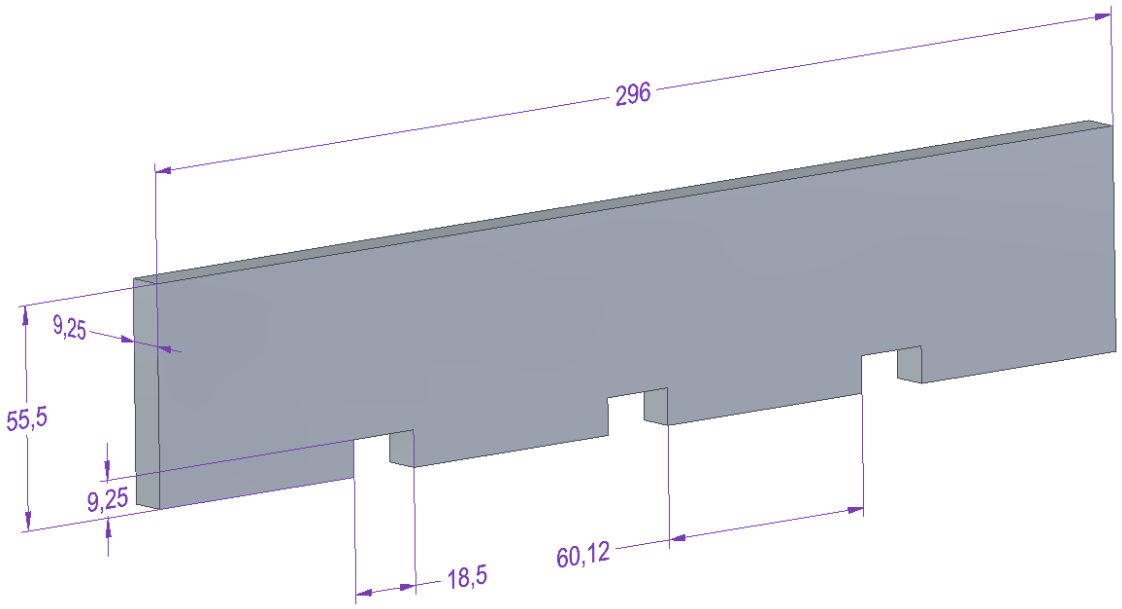


Figure 85 Baffle design in 3-D (all dimensions reported in mm).

The 3 baffles are equally spaced inside the tank and have a height of 55.5 mm and has a width of 9.25 mm, which is a consequence of the discretization of the model through the particles, Figure 86. In fact, to ensure an effective repulsion by the baffle a two layers of boundary particles must be realized, because in this way no penetration can occur, but especially, two particles that are approaching the wall from the two different sides, will not interact with each other, because the baffle thickness is large as their smoothing length.

For what concerns the position and dimensions of the holes, these have been determined after comparing the performance of different model designs with respect to the solution proposed by (Rajagounder 2016), as it is discussed in the next section.

Three different solutions have been designed and analyzed:

- Two holes, with a length of 55.5 mm and a height of 18.5 mm.
- Three holes, with a length of 18.5 mm and a height of 13.875 mm.
- Three holes, with a length of 18.5 mm and a height of 9.25 mm.

Table 19

Rate Density Approach		Unit
Moving Particles	35728	-
Fixed Particles	12860	-
Kernel function	Cubic Spline	-
Simulation time	43	hours
Time step	0.15305	ms

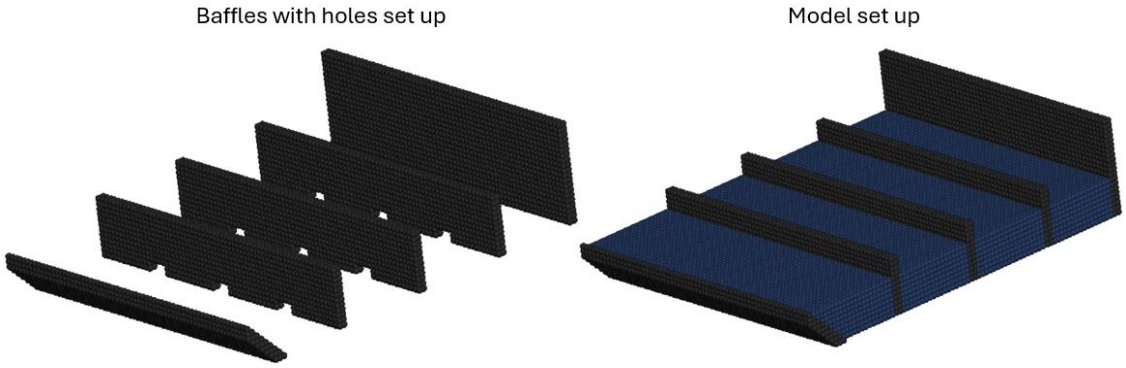


Figure 86 Discretized model set up (isometric view).

4.10.1 Numerical Results

Starting from Figure 87, where is displayed the evolution of the model's mass along the simulation, the highest variation with respect the initial value is at normalized time 2.06, where the difference is 0.0424%. This ensures that the mass is conserved along the simulation.

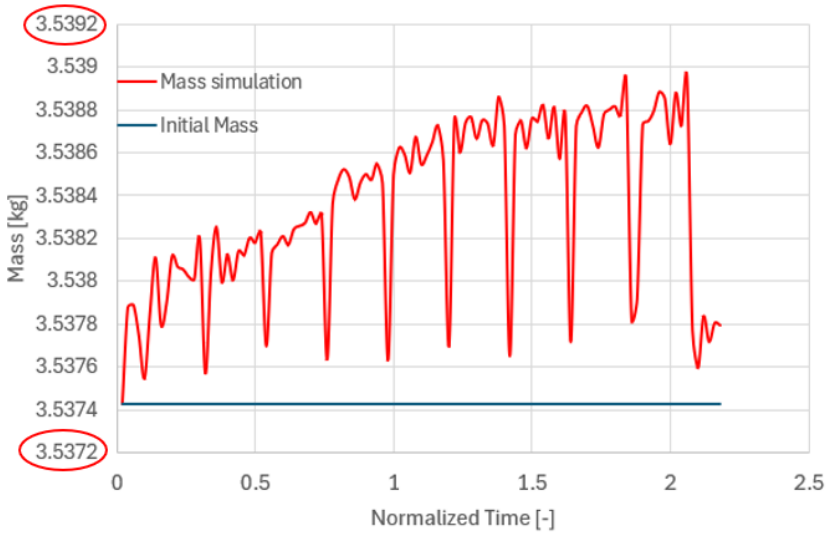


Figure 87 Evolution of the mass model along the simulation.

Moving to the analysis of Figure 88, which reports the evolution of the free surface height at the right boundary over the tank length (H/L), where the three different designs have been compared.

The initial solution involved the use of two holes, each measuring 55.5 x 18.5 mm. This configuration successfully mitigated the first peak at a normalized time of 0.50. However, the hole dimensions proved too large, failing to provide adequate blockage against fluid motion in subsequent stages. Consequently, the baffles did not perform effectively in sustaining the fluid control.

To address this, the second solution entailed reducing the size of the existing holes and incorporating a third hole while maintaining symmetry. This modification, with the three smaller holes, slightly decreased the maximum variation of H/L. Nevertheless, the overall sloshing performance was 1% worse compared to the original setup. Despite this minor setback, the results were close to an improvement. Therefore, a further reduction in the hole size was proposed to achieve a sloshing level lower than that of the original configuration.

By incrementally adjusting the hole dimensions and configurations, the objective is to find an optimal setup that effectively minimizes fluid motion while maintaining the structural integrity and performance of the baffles. The optimal configuration involves three baffles with three holes whose length and height are respectively 18.5 mm and 9.25 mm, represented by the redline.

This solution significantly improves upon the one proposed by Rajagounder, based on two parameters that measure the baffle's effectiveness in suppressing fuel motion inside the tank:

- Maximum variation of H/L during the simulation: for Rajagounder's solution, the highest variation occurs at normalized time 0.37 with a value of 0.1464. Implementing the new baffle, the highest variation still occurs at normalized time 0.37 but is reduced to 0.1367, indicating a 6.63% reduction in the maximum variation of H/L.
- Overall discrepancy of the H/L during the simulation: this is evaluated by calculating the areas between the curves and the dotted line showing the rest position. The area under the grey curve, representing the original configuration, has a value of 0.054. The area under the red curve (the new solution) is 0.0521, resulting in a 3.52% reduction.

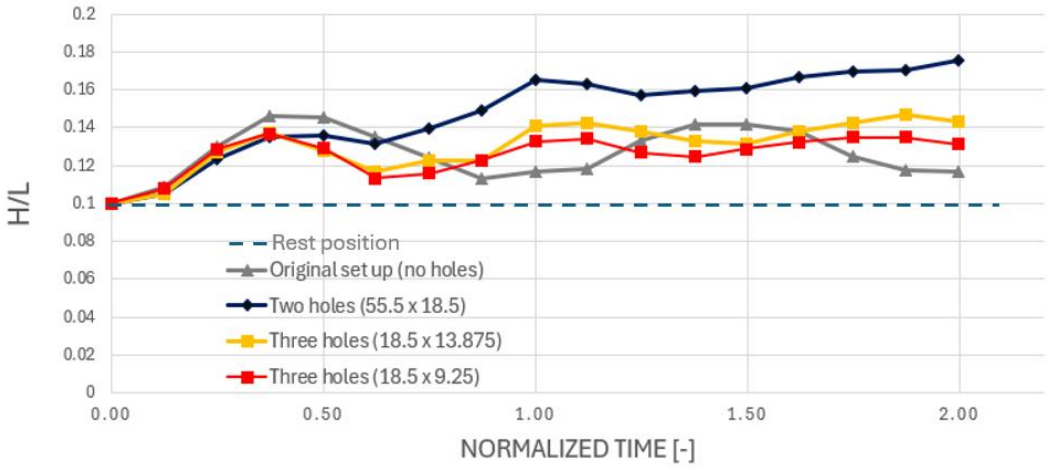


Figure 88 evolution of the free surface height at the right boundary over the tank length.

Moreover, examining Table 20, it is important to highlight that the new solution also reduces the volume required for designing the baffles. Despite introducing an additional baffle in the new configurations, their height is now less than half the height of the tank, unlike the previous design where the baffles were as tall as the tank. The new solution features three small holes (18.5 x 9.25) in the baffles. These holes are able to reduce the baffle volume further and, at the same time, enhance their effectiveness, because they potentially enable the baffles to reduce fuel motion under convective sloshing conditions (refer to "The Sloshing Problem" section).

Thus, it is possible to state that the baffle design, shown at Figure 85, is able to further reduce the fuel sloshing inside of the tank and at the same time it requires less material to produce the baffles.

Table 20 Design baffle volume.

Hole Configuartion (Length x Height)	Number of baffles	Volume	Unit
Original set up (No holes)	2	0.7345	dm^3
Two holes (55.5 x 18.5)	3	0.4319	dm^3
Three holes (18.5 x 13.9)	3	0.4469	dm^3
Three holes (18.5 x 9.25)	3	0.4499	dm^3

4.10.2 Final Discussion

The proposed solution involves reducing the height of the baffles from 140 mm to 55 mm, allowing for the addition of a third baffle. Each baffle includes three holes measuring 18.5 x 9.25 mm. These adjustments result in a lower overall volume compared to the original design. According to numerical simulations, the

new design reduces the peak value reached by the free surface to 0.1367 and decreases overall sloshing inside the tank by 3.52%. Therefore, the new solution not only improves fuel sloshing suppression but also reduces the volume of material required for baffle implementation, resulting in mass saving.

5 CONCLUSIONS

This thesis focuses on developing a Smoothed Particle Hydrodynamics (SPH) solver to investigate the motion of incompressible fluids, emphasizing comprehensive pre-processing and post-processing techniques for practical engineering applications.

Pre-Post-processing

The pre-processing phase has been completed and validated. The solver correctly reads and extracts the necessary information (particle mass and coordinates) from the .k file generated in LS Pre-Post, outlining the initial matrix.

The post-processing phase has also been completed and validated. The results are saved into a .vtk file (ASCII file format) and visualized in ParaView, the post-processing software chosen for this study.

Validation

The solver script, developed in MATLAB, has been validated through three benchmark examples from the literature:

- Dam break within a box.
- Dam break with a ramp.
- Fall of a droplet in water.

Comparison

Two solvers were developed, following different approaches:

Rate Density, which proved to be the most accurate in terms of numerical and graphical results. However, this method exhibited lower stability during simulations, leading to excessively high particle velocities that exceeded CFL conditions, resulting in unfeasible solutions under certain conditions.

Summation Density, which demonstrated perfect stability. None of the simulations produced unfeasible values. However, the accuracy was lower, particularly in graphical results, which were affected by oscillations on the free surface and significant splashing particle generation.

Additionally, the influence of the time step on the results was investigated, revealing its significant impact on numerical outcomes.

Based on the results obtained in the validation examples, the Rate Density solver with the implementation of the Cubic Spline kernel was found to be the most suitable for investigating the motion of incompressible fluids. It ensures high accuracy from both numerical and graphical outcomes, completely avoiding particle oscillations on the free surface and entirely preventing the generation of splashing particles.

Application to sloshing phenomena

Finally, the Rate Density solver was applied to investigate a sloshing problem in an automotive fuel tank. The two solutions proposed in the referenced paper were analyzed, and the solver's results were compared with those from the paper. A new solution was designed, reducing fuel sloshing by 3.52% and decreasing the highest variation in the height of the free surface relative to the rest position by 6.63%, while also reducing the volume required by the new baffles.

Future work

This study provides a reliable tool for simulating incompressible fluid dynamics, with potential applications in the automotive, aerospace, and maritime industries. Future research could focus on parallelizing the code, extending its application to more complex and computationally intensive fluid dynamics problems, and integrating more advanced post-processing techniques, such as binary files.

REFERENCES

Allen, MP. 1989. *Computer simulation of liquids.* Oxford, UK: Oxford Univ. Press.

Bagheri, Mohammadreza. 2023. "A review of smoothed particle hydrodynamics." *Computational Particle Mechanics.*

Batchelor, GK. 1967. *An introduction to fluid dynamics.* Cambbridge, UK: Cambridge Univ. Press.

Belytschko, T. 1996. "Meshless methods: an overview and recent developments." *Elsevier.*

Bhavikatti, SS. 2005. *Finite element analysis.* New Age International.

Bindel, David. 2011. "The SPH equations Applications of Parallel Computers (CS 5220)." *612 Rhodes Hall Dept of Computer Science Cornell University Ithaca, NY 14853-5169.*

Birdsall, C.K. 1991. *Plasma Physics via Computer Simulation (1st ed.).* Boca Raton: CRC Press.

Bistafa, Sylvio R. 2018. "On the development of the Navier–Stokes equation by Navier." *Revista Brasileira de Ensino de F´ısica, vol. 40, nº 2, e2603.*

Bui, Ha H. 2008. "SPH-Based numerical simulations for large deformation of geomaterial considering soil-structure interaction." *The 12th International Conference of International Association for Computer Methods and Advances in Geomechanics (IACMAG).*

Cai, Zhemin. 2021. "Evaluation of rigid body force in liquid sloshing problems of a partially filled tank: Traditional CFD/SPH/ALE comparative study." *The University of New South Wales, School of Mechanical and Manufacturing Engineering, Sydney, Australia.*

Calderon-Sanchez, J. 2019. "A SPH simulation of the sloshing phenomenon inside fuel tanks of the aircraft wings." *Universidad Politecnica de Madrid (UPM),Madrid, Spain.*

Cercos-Pita, J.L. 2015. "AQUAgpusph, a new free 3D SPH solver accelerated with OpenCL." *Computer Physics Communications.*

Chen, Jun. 2009. "SPH-based Visual Simulation of Fluid." *Proceedings of 2009 4th International Conference on Computer Science & Education.*

Chen, S. 2023. "Mesh-Free Methods with Special Focus on SPH. In: Advanced Computational Methods and Geomechanics." *Springer Tracts in Civil Engineering . Springer, Singapore.*

Cherniaev, Aleksandr. 2023. "The equation of motion for incompressible fluids." *(unpublished manuscript; received from the author December 12).*

Chhabra, R.P. 2010. "Non-Newtonian Fluids: An Introduction." *Rheology of complex fluids.*

Childs, Peter R.N. 2011. "Chapter 2 - Laws of Motion." In *Rotating Flow*, by Peter R.N. Childs, 17-52. Butterworth-Heinemann: Peter R.N. Childs.

Coleman, Neal. 2010. "A Derivation of the Navier-Stokes Equations." *B.S. Undergraduate Mathematics Exchange, Vol. 7, No. 1.*

2009. *Computational Fluid Dynamics.* 1640 Rhode-Saint-Genese: Prof. Dr. John F. Wendt.

Corless, RM. 2013. *A graduate introduction to numerical methods.* London, ON, Canada: Springer.

Crespo, A. J. C. 2007. "Boundary Conditions Generated by Dynamic Particles in SPH Methods." *CMC, vol.5, no.3* 173-184 .

Cundall, P. A. 1979. "A discrete numerical model for granular assemblies." *Géotechnique Volume 29 Issue 1* 47-65.

Dahlquist, G. 2003. *Numerical methods.* Linkpoing University, Sweden.

DaMing, LI. 2011. "Numerical simulation of droplet impacting liquid surface by SPH." *School of Civil Engineering &Key Laboratory of Harbor &Ocean Engineering Ministry of Education, Tianjin University,Tianjin 300072, China.*

Deissler, R. G. 1976. "Derivation of the Navier–Stokes equation." *Am. J. Phys. 44 (11)* 1128–1130.

Desbrun, Mathieu. 1996. "Smoothed Particles: A new paradigm for animating highly deformable bodies." *Eurographics Workshop on Computer Animation and Simulation (EGCAS), Poitiers, France.*

Diehl, S. 2008. "NuGrid: Toward High Precision Double-Degenerate Merger Simulations with SPH in 3D." *", in 10th Symposium on Nuclei in the Cosmos (NIC X), July 27–August 1, 2008, Mackinac Island, Michigan, USA, PoS(NIC X)155, SISSA, Trieste.* 58.

Domínguez, J. M. 2010. "Neighbour lists in smoothed particle hydrodynamics." *EPHYSLAB Environmental Physics Laboratory, Universidad de Vigo, Ourense, Spain.*

Encyclopedia Britannica. 2021. *"Fluid | Definition, Models, Newtonian Fluids, Non-Newtonian Fluids, & Facts".*

Fernández-Fernández, JA. 2022. "Fast Octree Neighborhood Search for SPH Simulations." *ACM Trans. Graph. 41, 6, Article 242.*

Gao, David Y. 2016. "Derivation of the Equations of Motion." In *Navier–stokes equations*, by David Y. Gao, 11-15. Virginia Polytechnic Institute and State University: David Y. Gao.

Garg, Sahil. 2017. "Meshfree Methods: A Comprehensive Review of Applications." *International Journal of Computational Methods Vol. 15, No. 3 (2018) 1830001.*

Goffin, Louis. 2013. "Development of a didactic SPH." *Universiity of Liège, Faculty of Applied Science, France.*

Gomez-Gesteira, MONCHO. 2010. "State-of-the-art of classical SPH for free-surface flows." *EPHYSLAB (Environmental Physics Laboratory), University of Vigo, Spain.*

Gui, Y. 2014. "Development of a family of explicit algorithms for structural dynamics with unconditional stability." *Nonlinear Dyn.* 1157–1170.

Hack, Henri Robert. 2018. "Viscosity." In *Encyclopedia of Engineering Geology*, by Henri Robert Hack, 926-928. Springer.

Hosain, M. L. 2018. "Numerical Investigation of Liquid Sloshing in Carrier Ship Fuel Tanks." *Department of Energy, Building and Environment, Mälardalen University, Sweded.*

Hosseini, Mahmood. 2012. "Simplified Dynamic Analysis of Sloshing in Rectangular Tanks with Multiple Vertical Baffles." *Associate Professor, International Institute of Earthquake Engineering and Seismology, Tehran, Iran, and Part Time Faculty Memner in Graduate School, Civil Engineering Deptment, Islamic Azad University- South Tehran Branch, Tehran, Iran.*

Huerta, Antonio. 2004. "Meshfree Methods." *Department of Mechanical Engineering, Northwestern University, 2145 Sheridan Road, Evanston, IL, USA.*

Ji, Z. 2018. "Numerical simulations of oil flow inside a gearbox by Smoothed Particle Hydrodynamics (SPH) method." In *Tribology International*, 47-58.

John R. Fanchi. 2002. "Chapter 6 - Fluid Properties." In *Shared Earth Modeling*, by John R. Fanchi, 87-107. Butterworth-Heinemann: John R. Fanchi.

Kelley, Scott M. 2018. "A Simple Leapfrog Integration Scheme to Find Optimal Interplanetary Trajectories." *Oregon State University Department of Physics.*

Kestin, J. 1978. "Viscosity of liquid water in the range− 8 C to 150 C." *J. Phys. Chem. Ref. Data 7* 941–948.

Khorasanizade, Shahab. 2018. "Improving Linked-Lists Using Tree Search Algorithms for Neighbor Finding in Variable-Resolution Smoothed Particle Hydrodynamics." *IDMEC, Instituto Superior T´ecnico, Universidade de Lisboa, Av. Rovisco Pais,1049-001 Lisboa, Portugal.*

Knupp, Patrick M. 2007. "Remarks on Mesh Quality." *Sandia National Laboratories , P. O. Box 5800 Albuquerque, NM 87185.*

Korson, Lawrence. 1968. "Viscosity of Water at Various Temperatures." *The Journal of Physical Chemistry.*

Li, Shaofan. 2002. "Meshfree and particle methods and their applications." *Department of Civil & Environmental Engineering, University of California, Berkeley CA 94720.*

Liu, G.R. 2003. *Mesh free methods: moving beyond the finite element method.* CRC Press.

Liu, Liu G.R. and M.B. 2003. ". Smoothed particle hydrodynamics: a meshfree particle method. World Scientific Publishing Company Incorporated." *World Scientific Publishing Company Incorporated.*

Liu, M.B. 2005. "Modeling incompressible flows using a finite particle method." *Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260, Singapore.*

Liu, M.B. 2010. "Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments." In *Archives of Computational Methods in Engineering*, by Michal Kleiber, Volume 17, pages 25–76,.

Lobovsky, L. 2007. "Smoothed particle hydrodynamics modelling of fluids and solids." *Applied and Computational Mechanics 1* 521 - 530.

Lombardi, JC. 1998. *Smoothed particle hydrodynamic simulations of stellar collisions.* Cornell University ProQuest Dissertations Publishing.

Lucy, L.B. 1977. "A numerical approach to the testing of the fission hypothesis." *Institute of Astronomy, Cambridge, United Kingdom.*

Maghbouli, A. 2015. "Effects of grid alignment on modeling the spray and mixing process in direct injection diesel engines under non-reacting operating conditions." In *Applied Thermal Engineering*, 901-912.

Marchioli, Cristian. 2020. "Lecture 4 - Conservation Equation." In *Fluid Dynamics*, by Cristian Marchioli. Udine, Italy: University of Udine.

Marri. 2002. "Smoothed particle hydrodynamics for galaxy-formation simulations: improved treatments of multiphase gas, of star formation and of supernovae feedback." *Monthly Notices of the Royal Astronomical.*

Marrone, S. 2011. "δ-SPH model for simulating violent impact flows." *Computer Methods in Applied Mechanics and Engineering,Volume 200, Issues 13–16,* 1526-1542.

Mehra. 2005. "High velocity impact of metal sphere on thin metallic plates: A comparative smooth particle hydrodynamics study." *Journal of Computational Physics.*

Milan Toma, Rosalyn Chan-Akeley. 2021. "Fluid–Structure Interaction Analyses of Biological Systems Using Smoothed-Particle Hydrodynamics." (https://doi.org/10.3390/biology10030185) 1.

Mokos,] Athanasios. 2016. "A multi-phase particle shifting algorithm for SPH simulations of violent hydrodynamics with a large number of particles."

Molteni, Diego. 2009. "A simple procedure to improve the pressure evaluation in hydrodynamic context using the SPH." *Computer Physics Communications* 861-872.

Monaco, Antonio Di. 2014. "SPH Modeling of Solid Boundaries Through a SemiAnalytic Approach." *Engineering Applications of Computational Fluid Mechanics* 1-15.

Monaghan, J. J. 1992. "Smoothed Particle Hydrodynamics." *Annu. Rev. Astron. Astrophys* 543-574.

Monaghan, J. J. 1999. "SOLITARY WAVES ON A CRETAN BEACH." *JOURNAL OF WATERWAY, PORT, COASTAL, AND OCEAN ENGINEERING.*

Monaghan, J.J. 1977. "Smoothed particle hydrodynamics: theory and application to non-spherical stars." *Monthly Notices of the Royal Astronomical Society, vol. 181, Nov. 1977, p. 375-389.*

Monaghan, J.J. 1994. "Simulating free surface flow with SPH."

Moretti, Felipe. 2014. "Numerical and Experimental Evaluation of Sloshing in Automotive Fuel Tanks." *SAE Technical Paper 2014-36-0122.*

Moura, Carlos A. de. 2013. *The Courant–Friedrichs–Lewy (CFL) Condition.* London: Springer New York Heidelberg Dordrecht .

Müller, Matthias. 2003. "Particle-Based Fluid Simulation for Interactive Applications." (D. Breen, M. Lin (Editors)) 2-3.

Nair, Prapanch. 2014. "An improved free surface modeling for incompressible SPH." *Department of Mechanical Engineering, Indian Institute of Science, Bangalore, India.*

Nimisha, P. 2022. "Slosh Damping in Rectangular Liquid Tank With Additional Blockage Effects Under Pitch Excitation ." *ASME. J. Fluids Eng. December 2022; 144(12): 121403.*

Paggi, Marco. 2020. "SPH modelling of hydrodynamic lubrication: laminar fluid flow–structure interaction with no-slip conditions for slider bearings." *Springer.*

Peng. 2019. "LOQUAT: an open-source GPU-accelerated SPH solver for geotechnical modeling." *Acta Geotechnica. 14. 10.1007/s11440-019-00839-1. .*

Purkayastha, Subhrangshu. 2022. "Review of Smooth Particle Hydrodynamics and its Applications for Environmental Flows." *Journal of The Institution of Engineers (India): Series A.*

Rajagounder, Rajamani. 2016. "A Study of Liquid Sloshing in an Automotive Fuel Tank under Uniform Acceleration." *Department of Production Engineering, PSG College of Technology, Coimbatore - 641 004, India.*

Randles, P.W. 1996. "Smoothed Particle Hydrodynamics – some recent improvements and applications." *Comput. Methods Appl. Mech. Eng., 138* 375- 408.

Rio, Gérard. 2005. "Comparative study of numerical explicit time integration algorithms." *Advances in Engineering Software Volume 36, Issue 4.*

S.Koshizuka. 1995. "Moving Partcile semi implicit method for fragmentation of incomressible fluids." *University of Tokyo, Nuclear Engineering Research Laboratory, Japan.*

Schneiderbauer, Simon. 2014. "What do the Navier–Stokes equations mean?" *Eur. J. Phys. 35 015020.*

Singal, Vaibhav. 2013. "CFD Analysis of a Kerosene Fuel Tank to Reduce Liquid Sloshing." *Energy Division, School Of Mechanical and Building Sciences, VIT University, Vellore - 632014, Tamil Nadu, India.*

Springel, Volker. 2002. "Cosmological smoothed particle hydrodynamics simulations: a hybrid multiphase model for star formation." *Monthly Notices of the Royal Astronomical.*

Succi, S. 2001. *The lattice Boltzmann equation: for fluid dynamics and beyond.* Oxford Univ. Press.

Takamiya, Hiroki. 2011. "Smoothed particle hydrodynamics analysis on semi-solid metal forming process." *Japan Journal of Industrial and Applied Mathematics.*

Vesenjak, M. 2007. "Application Aspects of the Meshless SPH Method." *University of Maribor, Faculty of Mechanical Engineering, Smetanova 31, SI-2000 Maribor, Slovenia.*

Viccione, G. 2008. "Defining and optimizing algorithms for neighbouring particle identification in SPH fluid simulations." *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS.*

Vrh, Marko. 2009. "Improved explicit integration in plasticity." *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING.*

Wang, Zhi-Bin. 2016. "An overview of smoothed particle hydrodynamics for simulating multiphase flow." *Applied Mathematical Modelling, Volume 40, Issues 23–24* 9625-9655.

Wetzstein, M. 2009. "VINE—A NUMERICAL CODE FOR SIMULATING ASTROPHYSICAL SYSTEMS USING PARTICLES. I. DESCRIPTION OF THE PHYSICS AND THE NUMERICAL METHODS." *The Astrophysical Journal Supplement Series.*

Yagawa, Genki. 1999. "Recent developments of free mesh method." *Int. J. Numer. Meth. Engng. 2000; 47:1419}1443.*

Yang, X.F. 2013. "A new kernel function for SPH with applications to free surface flows." *Key Laboratory for Mechanics in Fluid Solid Coupling Systems (LMFS), Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China.*

Ye, Ting. 2019. "Smoothed particle hydrodynamics (SPH) for complex fluid flows: Recent developments in methodology and applications." *Phys. Fluids 31, 011301* .

Yerro, Alba. 2015. "MPM modelling of landslides in brittle and unsaturated soils."

Yosef. 2023. "Modeling Dynamics of Laterally Impacted Piles in Gravel Using Erosion Method." *Geotechnics 3, no. 4* 1251-1278.

Zhang, Enhui. 2020. "Influencing analysis of different baffle factors on oil liquid sloshing in automobile fuel tank." *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering.*

Zheng, Mianlun. 2017. "A novel unconditionally stable explicit integration method for finite element method." *Springer.*

Zheng, S. 2021. "Topology optimization on fuel tank rib structures for fuel sloshing suppression based on hybrid fluid–solid SPH simulation." In *Thin-Walled Structures*, by N. Silvestre. Elsevier.

Zheng, Shuai. 2021. "Topology optimization on fuel tank rib structures for fuel sloshing suppression based on hybrid fluid–solid SPH simulation." *School of Software Engineering, Xi'an Jiaotong University, China.*