Simulating Bending Failure of Ice using Smoothed Particle Hydrodynamics

A re-usable framework for simulating of fluid problems using SPH

Jasper Koning
4209079

July 2019

C. Keijdener A. Metrikine

Offshore Engineering
Delft University of Technology

Acknowledgments

This thesis would have been impossible without the help of a couple important people, that have worked tirelessly to help and guide me through this process.

Firstly, I would like to thank my thesis supervisor for suggesting the topic, in the first place, helping with the development of the ice-model, and for supervising, advising and critiquing my work.

Secondly, I would like to thank Marielle for all the talk about particles and physics she has put up with over the past few months. All the time she has put into reviewing my work and giving me the motivation to finish on time.

Abstract

The offshore industry is a conservative industry, sticking to rigid best-practices and reluctant to try new techniques. A promising new technique that hasn't found much adoption in offshore industry is SPH. This thesis aims to improve the reach and use of SPH within the offshore industry. With an abundance of world's unexplored hydrocarbons located in the Arctic Region, 18%, ice-structure interactions (ISI) are set to increase. Modeling these ISI requires complex dynamics, SPH can model these ISI dynamics without extra treatment.

SPH works by interpolation of a set of neighbouring particles using a weighing function, see Figure 1. The particle being interpolated is shown in red, the neighbouring particles in black, and the kernel (weighing) function in green.

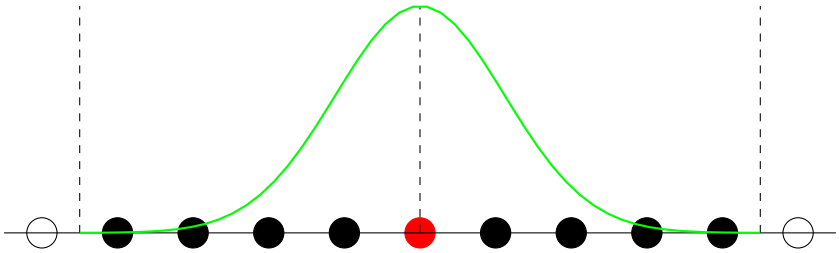


Figure 1: 1D SPH

This modeling technique offers many advantages over conventional rules-of-thumb, Finite Element Modeling (FEM), and Finite Volume (FV) methods. SPH is a particle based method, thus, uniquely suited for problems with large displacements, or discontinuities.

However, in the standard, weakly-compressible (WCSPH) method spurious pressure fluctuations and particle clustering can occur. By implementing the plethora of correction methods and equations present in literature, such as kernel corrections, incompressible variants, or density corrections, these drawbacks can be circumvented and a robust framework can be formed.

An implementation of WCSPH that focuses on adaptability and flexibility is presented in this work. The flexibility of the implementation allows future researchers to focus on the core of their research, only changing the equations they are interested in, instead of implementing all the required equations for a full SPH simulation. A validation study of the implementation assures that it matches closely with existing implementations and real-world results.

The mathematical model developed in Keijdener, Hendrikse, and A. Metrikine 2018 and Keijdener and A. V. Metrikine 2014 is implemented in the proposed WCSPH implementation. Comparing the results shows close agreement for the breaking length. However, significant differences are present in the failure time results. Despite this it shows the possibility of combining solid mechanics with SPH, and validates the integration method, solver, and SPH model.

In the future I expect the use of SPH to grow in all industries, the offshore industry included. SPH is an extremely promising technique with many advantages over conventional techniques. As shown by the comparison with Keijdener, Hendrikse, and A. Metrikine 2018, SPH can approximate dynamics without extra treatment and is, therefore, a valuable tool for the offshore industry.

Abbreviations

CFD . . . . . Computational Fluid Dynamics
CFL . . . . . Courant-Friedrichs-Lewy
DoF . . . . . Degree of Freedom
EoM . . . . . Equation of Motions
FD . . . . . Finite Difference
FE . . . . . Finite Element
FEM . . . . . Finite Element Modeling
FV . . . . . Finite Volume
ICSPH . . . . . Incompressible SPH
IQR . . . . . Interquartile range
ISI . . . . . Ice structure interactions
PDE . . . . . Partial Differential Equation
SPH . . . . . Smoothed Particle Hydrodynamics
TU . . . . . Technical University
WCSPH . . . . . Weakly-Compressible SPH

Contents

Chapter 1

Introduction

A significant amount (18%) of the worlds unexplored hydrocarbons are expected to be located offshore in the Arctic Region (Bird et al. 2008; Gautier et al. 2009). Floating structures are essential for the extraction of these offshore hydrocarbons (Keijdener, Hendrikse, and A. Metrikine 2018). With ice being a frequent occurrence in the Arctic Region, understanding, predicting, and modeling of ice structure interactions (ISI) is, therefore, essential to sustainable and safe extraction of hydrocarbons.

The offshore industry has traditionally been an extremely conservative industry, reluctant to try new methods and ways, instead opting to stay with the tried and true methods. This has also been the case with the adoption of smoothed particle hydrodynamics (SPH), even-though small steps are being made in the simulation of offshore problems using SPH (Stansby 2018; Gotoh and Khayyer 2016).

This thesis aims to apply this new modeling technique to the ISI, to increase awareness of SPH in general and specifically in the offshore sector, see Section 1.2 for detailed objectives. The necessary background information for this thesis is presented in Section 1.1.

1.1 Background Information

Computational modeling is a powerful tool for any engineering discipline. Using a computational model extremely complex simulations can be performed to assess the quality of the design or test principles. Thanks to massive increases in computational power over the past decades, these simulations can be many orders of magnitudes more complex and accurate than back-of-the-envelope calculations commonly used in engineering.

A plethora of computational modeling techniques, methods and applications exist. The most common being: Finite Element Modeling (FEM), Finite Volume (FV), Finite Difference (FD). A (relative) newcomer in the modeling toolbox is smoothed particle hydrodynamics (SPH), the modeling technique central to this thesis.

Finite Difference

Finite difference is an approximation method for derivatives on computers. This method is required since any derivative in an partial differential equation (PDE), see Equation 1.1, involves infinite small steps, which can not be computed on a computer.

$$\rho A \frac{\partial^2 u}{\partial t^2} + EA \frac{\partial^2 u}{\partial x^2} = 0 \tag{1.1}$$

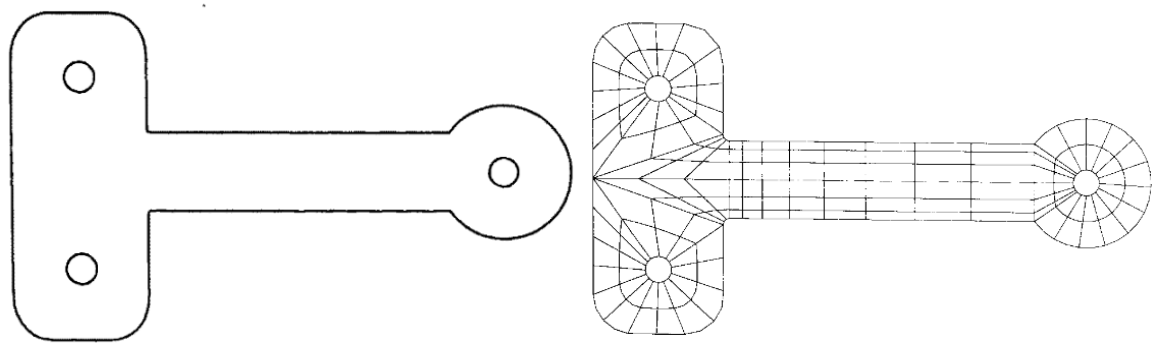
This Equation of Motion (EoM), for a rod, shows the relation between the acceleration of the rod, and the internal force in the rod. The second order derivative in this equation can be approximated using a finite difference scheme. A nearly infinite number of FD schemes exist, however, central, forward, or backward difference are commonly used schemes. A central difference scheme for for the first-order derivative is shown in Equation 1.2, this scheme has a second-order error $\mathcal{O}(\Delta x^2)$.

$$\frac{w(a + \frac{\Delta x}{2}) - w(a - \frac{\Delta x}{2})}{\Delta x} = w'(a) \tag{1.2}$$

FD schemes require grid points to operate on, therefore, they are generally only used for simple geometries. When complex geometries are required Finite Element Modeling (FEM) is generally a more suitable option. However, when a geometry can be defined using a regular grid and boundary conditions are relatively simple, FD is highly accurate.

Finite Element Modeling

Perhaps the most well known of the modeling techniques is Finite Element Modeling (FEM). This method sub-divides the geometry in many small simple finite elements (FE), this collection of FEs is called a finite-element mesh, see Figure 1.1. This meshing procedure can be time-consuming and difficult, especially around circular shapes, as shown in the bracket, requiring many elements to approximate the circular shapes.



(a) Bracket before meshing.

(b) Bracket meshed into FEs.

Figure 1.1: Finite element meshing of bracket (Bathe 2014)

For each of the simple finite-elements the PDEs can be formulated and discretized into several degrees of freedom (DoFs). This results in a large system of algebraic equations which can be solved using a matrix solver. The selection of the correct mathematical model and corresponding PDEs is essential for FEM.

Finite Volume

Similarly to FEM, the object in FV is divided into small geometrically simple but finite-sized elements. FV is based on the conservation of flux, the flux entering one volume has to be equal to the flux leaving the neighbouring cells. This allows complex geometries and unstructured meshes. This is especially relevant for fluids which can deform heavily when flowing.

FV is the natural choice for computational fluid dynamic (CFD) simulations since the PDE involved in fluid flow are conservation laws.

Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics, as the name suggests, is a particle method. These particle methods are inherently suited for problems with large deformations, or discontinuities, and can solve these problems without numerical diffusion (Khayyer, Gotoh, and Shao 2009).

This particle discretization approach is different than the one used in FV and FEM which divides the domain in cells or shapes, instead it divides the domain into (regular) points called particles. These particles are somewhat analogous to actual molecular particles, they interact which each other and influence each other. This influence on each other is determined using a weighting (kernel) function. An more in-depth description of the mechanics of SPH can be found in Chapter 2.

1.2 Objectives

As ISI is predicted to increase, due the amount of unexplored hydrocarbons in the Arctic Region, modeling this becomes critical. Previous work on ISI and the effect of hydrodynamics and hydrostatics on these interactions has been carried out by Valanto 1992 and Keijdener, Hendrikse, and A. Metrikine 2018. However, these only constitute experiments and mathematical models. The goal of this thesis is to apply SPH to these ISI. Furthermore, a plethora of correction methods and modifications exist in SPH, understanding these is the second goal of this thesis. The final goal of this thesis is to develop a re-usable program, specifically aimed at students and faculty staff, that enables them to easily simulate 2D problems. Based on these goals the following research question has been formed:

> *How can Smoothed Particle Hydrodynamics be used to effectively model hydrodynamic bending failure of level ice during ice-structure interactions?*

Since this question is complex and requires implementation of a model, several research objectives have been defined to simplify the process and break it down into several sequential steps.

1. Gathering of literature with regards to SPH that can be applicable to offshore industry.
2. Define problems with SPH and corresponding solutions provided in literature.
3. Create a re-usable program specifically aimed at students, and university personal that allows them to solve 'simple' 2D SPH problems in an efficient and understandable manner.
4. Verification of the program using a dam break scenario, and a fluid containment scenario.
5. Implementation of the ISI model as defined by Keijdener, Hendrikse, and A. Metrikine 2018.

1.3 Methodology

As mentioned in Section 1.2 the first objective is the gathering of existing knowledge of SPH. In order to achieve this objective an extensive literature study is carried out to assess the past and present state of the art with regards to SPH. Based on this literature, the solutions provided by the present state-of-the-art for various problems with regards to SPH will be analysed, fulfilling the second goal of this thesis as listed in Section 1.2. Then, a re-usable computer program will be created. Since it is focused on current students and professors at the TU two programming languages are considered for the creation of the program namely, Python and MATLAB. The baseline (WC)SPH as described in (Monaghan 1994) will be implemented. This program will be verified using known values and simulations. Lastly, the implementation of the ISI model as defined by Keijdener, Hendrikse, and A. Metrikine 2018 into the developed program allows the answering of the overall research question of this Thesis.

1.4 Contribution

This research will provide a contribution to ongoing research efforts into SPH by supplying future researchers with a framework to evaluate equations and changes in. Furthermore, it highlights the competencies of SPH as a method, and underlines the need and possibilities for it in the offshore engineering field.

1.5 Motivation

My personal motivation for selecting this subject comes from my long held interest in programming. I've started programming from an early age, and continue to do so to this day. This subject attracted me since it is at the fulcrum of programming and engineering, combining the two disciplines. Furthermore, it method intuitively makes sense, raising the question why it has not be utilised more in the past.

1.6 Scope

The scope of the research will include an implementation of WCSPH as defined in Monaghan 1994. It will also include a literature study into SPH in general and specifically it applications to hydraulic flow. Lastly, it will study common problems and solutions for SPH applications to fluid flow problems. These problems and solutions will be limited to commonly accepted, having been referenced, mentioned and examined in different sources. Therefore, the sourcing will be biased towards review papers and articles. The ice-structure interactions will be limited to the model described by Keijdener, Hendrikse, and A. Metrikine 2018.

The following equations, and methods are implemented in the WCSPH implementation and fall within the scope of this research, see Table 1.1.

Table 1.1: Implemented equations, and methods.

Component	**Implemented equations**
Kernels	Cubic Spline (M. Liu and G. R. Liu 2010) Gaussian (Monaghan 1994) Wendland (Wendland 1995)
Time-stepping	Diffusion (Violeau and Rogers 2016) CFL (Violeau and Rogers 2016) Force (Violeau and Rogers 2016) Constant
Methods	WCSPH (Monaghan 1994)
Integration	Euler leap-frog (Monaghan 2005) Newmark-Beta[1](Newmark 1959)
SPH	Tait (Monaghan 1994) Momentum (Monaghan 1994) Continuity (Monaghan 1994) Boundary Force (Monaghan 1994)
Equation of State	Tait (Batchelor 1967)
Miscellaneous	Kinetic Energy Smoothing-Length (Monaghan 2005) Summation-Density (Monaghan 1994) Shepard Filter (Bonet and Lok 1999)

[1]Not usable for SPH particles, only coupling particles

Chapter 2

SPH

SPH is a mesh-free method that models a fluid, solid or gas as a discrete number of particles interpolated using a so called kernel function, see Section 2.1. It offers significant advantages over rivaling methods such as computational fluid dynamics (CFD) and the finite volume method (FVM). However, it is susceptible to spurious pressure changes, boundary penetration, and tensile instability, these issues are covered in more detail in Chapter 3.

The main advantage of SPH is that it is a truly mesh-free method. This offers a significant advantage over rivaling methods such as CFD. The fluid is only discretized in nodes or particles and the connections between those particles are generated automatically by virtue of being neighbouring particles. In meshed methods such as CFD and FVM the connections between neighbouring nodes are fixed and can not be changed. However, changes have recently been made to try to work around this using overset meshes (Ma et al. 2018).

Boundary conditions are often challenging in meshed methods. The boundary has to be divided into smaller cube or triangular shapes, this can be time consuming and difficult (Bathe 2007). SPH allows easy creation of boundary conditions by numerous methods such as ghost particles, mirror particles or fixed boundaries, see Sections 2.5 and 3.5.

Due to its massively parallel scale, SPH can easily be scaled by increasing the amount of computational power, for example using GPUs (Gomez-Gesteira et al. 2010). This allows for significant reductions in computation time.

Lastly, it allows the capture of complex interaction phenomenon easily by interpolation.

2.1 Interpolation

As described in Section 1.1, SPH can approximate any function using a two-step process. Firstly, a continuous interpolation is created, then a discrete approximation of this continuous interpolation is performed, (Violeau and Rogers 2016).

This interpolation is an integral interpolation and forms the basis of SPH. Using this integral interpolation the smoothed version of any scalar or vector field $f_s(\vec{r})$ can be found based on the non-smoothed $f(\vec{r})$ field, (Monaghan and Gingold 1977). The interpolation of an arbitrary function is shown in Equation 2.1:

$$f_s(\vec{r}) = \int_\Omega W(\vec{r} - \vec{r'}, h)\ f(\vec{r'})\ d\vec{r'} \tag{2.1}$$

With $W(\vec{r} - \vec{r'}, h)$ being the smoothing kernel and Ω the space containing all particles. This smoothing kernel is the single most important function used in SPH and dictates the entire SPH interpolation. Any function can be used as an interpolation function as long as it adheres to the following criteria, (M. Liu and G. R. Liu 2010).

1. Unity over its domain; The interpolating kernel should not increase or decrease the function that is interpolated, therefore, unity is required, see Equation 2.2.

$$\int_{\Omega} W(x, h)dx = 1 \tag{2.2}$$

2. Compact support area, since the kernel function dictates the area of support this area should be compact as to limit the number of neighbours that need to be taken into account, see Section 2.2.

3. Positive for any point within its support domain. The kernel function should not cause any repulsive forces.

4. Decay monotonically; The further the distance from the origin, the lower the kernel value.

5. Symmetric, left and right of the y-axis should be equal for the kernel function.

6. Sufficiently smooth and continuous to obtain gradients and derivatives.

Any arbitrary function that follows these criteria can be used as a smoothing kernel in the SPH interpolation. The basic identity of the kernel function then looks as follows, see Equation 2.3.

$$f(x) = \int_{\Omega} f(x') \cdot \delta(x - x') \; dx' \tag{2.3}$$

In which the $\delta(x - x')$ is the Dirac delta function, see Equation 2.4.

$$\delta(x - x') = \begin{cases} 1, & x = x' \\ 0, & x \neq x' \end{cases} \tag{2.4}$$

The Dirac function $\delta(x - x')$ can not be used for establishing a 'point' support and therefore is replaced with a so called *Smoothing Function* $W(R, h)$ with a smoothing length h and a particle distance of $R = x - x'$. This replacement results in the following final form of the kernel function, see Equation 2.5.

$$f(x) = \int_{\Omega} f(x') \cdot W(x - x', h) \; dx' \tag{2.5}$$

The smoothing particle length (h) represents the length of influence of the smoothing function W. The larger this length the larger the influence of neighbouring particles. It provides the cut-off point where distant particles no longer influence the particle.

Many different smoothing kernels have been used in literature each with their own advantages, disadvantages, and complexities. A non exhaustive list of commonly used kernel functions is listed below. All kernels are shown with the unit-less distance q, see Equation 2.6.

$$q(r, h) = \frac{r}{h} \tag{2.6}$$

Gaussian Kernel

The Gaussian Kernel is the original kernel used by (Monaghan and Gingold 1977) and is a single continuous function, see Equation 2.7 and Figure 2.1.

$$W(q) = \alpha_d \cdot e^{-q^2} \tag{2.7}$$

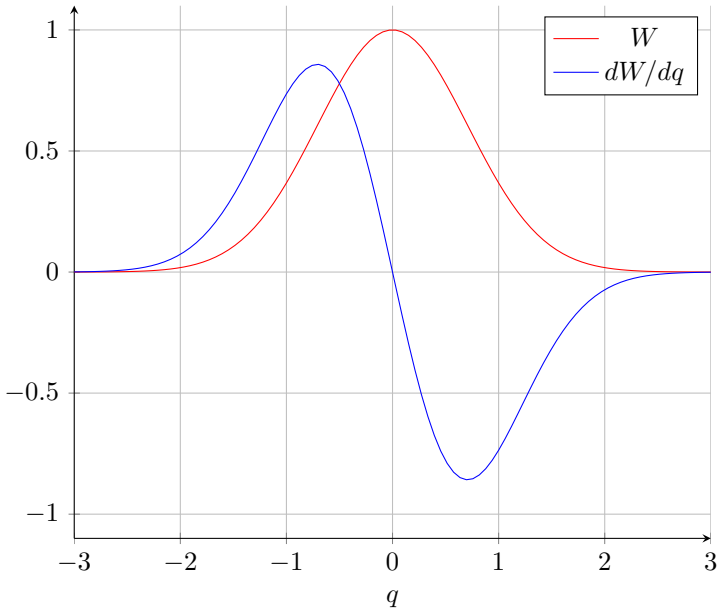


Figure 2.1: Gaussian Kernel and its derivative.

However, this continuous function is an asymptotic function and only tends to zero at infinity, see Figure 2.1. Therefore, the function is assumed to be zero at values larger then 3, resulting in the following actual implementation, see Equation 2.8.

$$W(q) = \begin{cases} \alpha_d \cdot e^{-q^2}, & q \leq 3 \\ 0, & q > 3 \end{cases} \tag{2.8}$$

Because of this discontinuity, this function is not in wide use anymore, and has been superseded by a so-called super Gaussian kernel or other kernel functions.

Super Gaussian

The super Gaussian kernel has mostly superseded the use of the (regular) Gaussian kernel in SPH. This kernel was introduced by Monaghan 1992 as a more accurate Gaussian kernel.

$$W(q) = \begin{cases} \frac{1}{\pi^{d/2} h} \cdot e^{-q^2} \cdot \left(\frac{d}{2} + 1 - q^2\right), & q \leq 3 \\ 0, & q > 3 \end{cases} \tag{2.9}$$

with d being the number of dimensions.

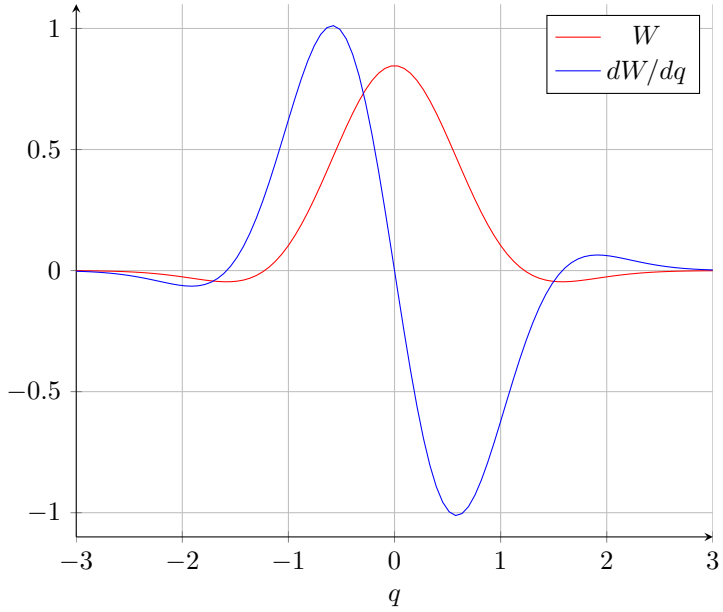


Figure 2.2: Super Gaussian Kernel and its derivative for 1D.

The super Gaussian kernel still suffers from the same draw-backs as the Gaussian kernel; having a discontinuity at $q = 3$. Therefore, even-though the super Gaussian kernel has superseded the Gaussian kernel it, in turn, has been superseded by different kernel functions such as the Cubic Spline.

Cubic Spline

The cubic spline is a piece-wise continuous function with continuous derivatives up to the second derivative. It is based on the M-spline group of functions, and has seen wide use in (fluid) SPH since its introduction (Monaghan 1994; Morris, Fox, and Zhu 1997; M. Liu and G. R. Liu 2010). The function consists of three sub-equations and conditions and is shown in Equation 2.10.

$$M_4(q) = \alpha \times \begin{cases} 1 - \frac{3}{2}q^2 + \frac{3}{4}q^3, & 0 \leq q \leq 1 \\ \frac{1}{4}(2-q)^3, & 1 \leq q \leq 2 \\ 0, & q > 2 \end{cases} \tag{2.10}$$

The function and the three distinct sections are displayed in Figure 2.3. In this figure, the distinct sections are drawn with $\alpha = 1$.

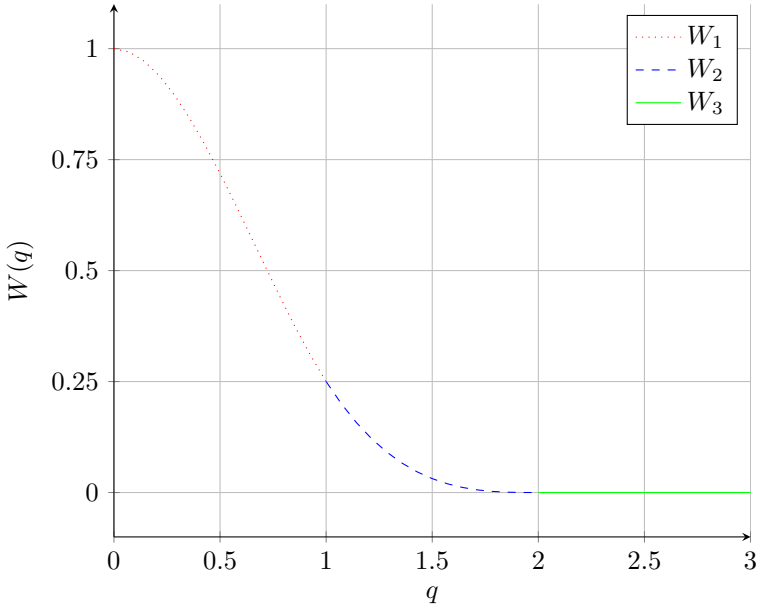


Figure 2.3: The three sections of the cubic spline

This cubic spline function is then scaled based on the number of dimensions to ensure the function stays normalised in space. These are as follows see Table 2.1.

Table 2.1: Cubic M-spline scaling factors

Dimension	**Factor** (α)
2D	$10/(7\pi \cdot h^2)$
3D	$1/(\pi \cdot h^3)$

The derivative of the cubic spline function can be taken and results in the following set of equations for the boundaries.

$$M_4'(q) = \alpha \times \begin{cases} -3q + \frac{9}{4}q^2, & 0 \leq q \leq 1 \\ -\frac{3}{4}(2-q)^2, & 1 \leq q \leq 2 \\ 0, & q > 2 \end{cases} \tag{2.11}$$

A M_4 spline with a α value of 1 leads to the following function values and its derivative, see Figure 2.4.

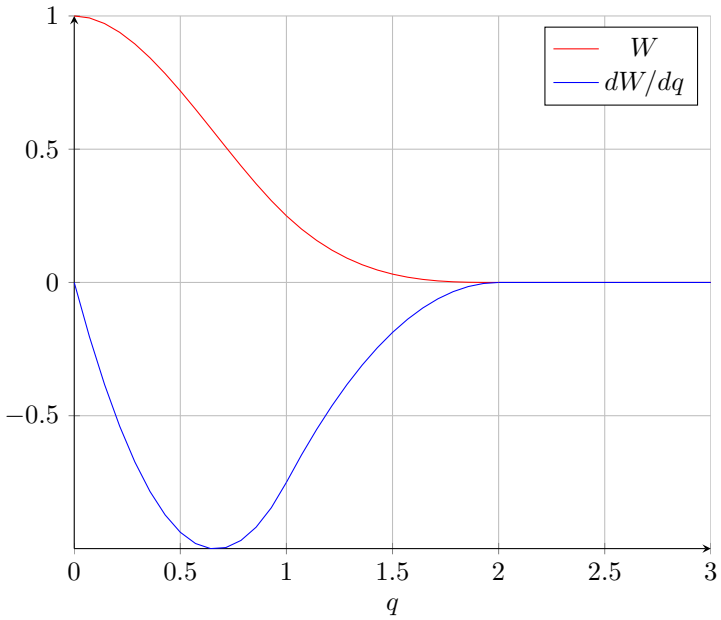


Figure 2.4: The cubic spline and its derivative.

As can be seen from the figure, the first derivative of the M-spline function is piece-wise continuous, resulting in smooth derivatives over the domain.

Quintic Spline

The derivatives of the cubic spline (M_4) are only continuous until the $(n-2)$ or second derivative. Therefore, if higher order derivatives are required for a certain problem a higher order function is required. A quintic spline is a fifth order spline that has continuous derivatives until the third order. This spline has been described by (M. Liu and G. R. Liu 2010), and has the following Equation 2.12.

$$W(q) = \alpha \times \begin{cases} (1 - \frac{1}{2}q)^4 \cdot (2q+1), & 0 \leq q \leq 2 \\ 0, & q > 2 \end{cases} \tag{2.12}$$

This quintic spline only consists of two equations instead of the three required for the cubic spline, see Equation 2.10. Its first derivative is shown in Equation 2.13 and Figure 2.5.

$$W'(q) = \alpha \times \begin{cases} -\frac{5}{8}(2-q)^3 q, & 0 \leq q \leq 2 \\ 0, & q > 2 \end{cases} \tag{2.13}$$

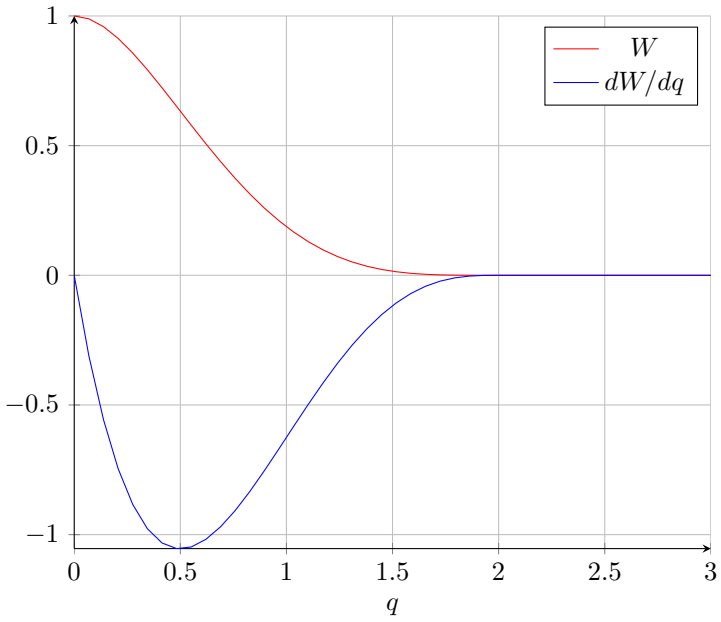


Figure 2.5: The quintic spline and its derivative.

Wendland Kernel

In 1995 Wendland developed an alternative set of quintic kernels, commonly referred to as Wendland Kernels, (Wendland 1995). Wendland kernels are denoted by their smoothness C^n value. Commonly, three possible Wendland kernels are used: C^2, C^4 and C^6. Following (Monaghan and Kajtar 2009), errors are significantly reduced when using the Wendland quintic (C^4) kernel. Furthermore, accuracy is improved and computational time significantly reduced for quintic kernels (Panizzo, Capone, and Dalrymple 2007).

The 2D Wendland quintic (C4) spline is given as, see Equation 2.14 and Figure 2.6.

$$W(q) = \frac{9}{4\pi h^2} \times \begin{cases} \left(1 - \frac{q}{2}\right)^6 \left(\frac{35}{12}q^2 + 3q + 1\right), & 0 \leq q \leq 2 \\ 0, & q > 2 \end{cases} \tag{2.14}$$

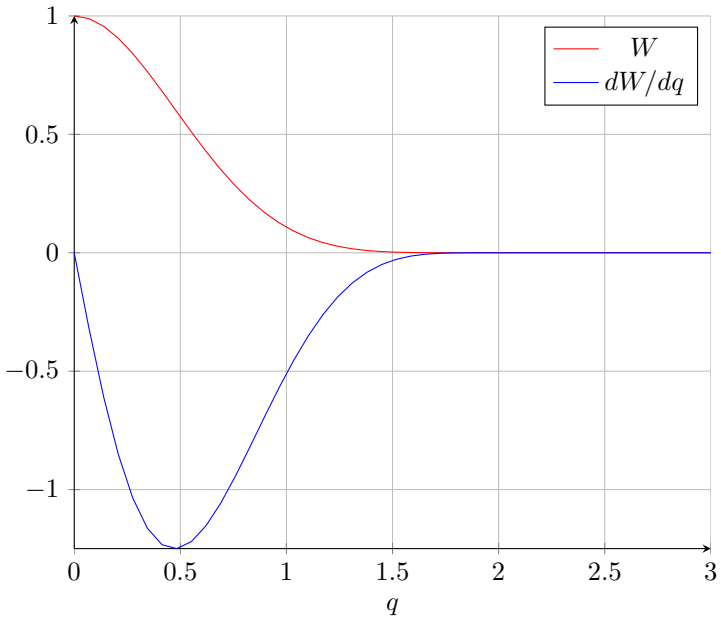


Figure 2.6: Value and derivative of Wendland C^4 Kernel.

2.2 Discretization

Since infinite integrals such as described in Section 2.1 are difficult to execute on computers, these need to be discretized.

In the discretization, any solid, liquid, or gas is approximated as a set of, so called, particles. These particles are somewhat analogous to actual molecular particles. However, they occur on a macro scale and possess properties not displayed by single particles such as pressure and density.

The discretization in SPH for an arbitrary function (f) is shown in Equation 2.15.

$$f(\vec{r}) = \sum_b m_b \frac{f_a}{\rho_b} W_{ab} \tag{2.15}$$

Where b is the set of all other particles, m_b being the mass of the other particle, and $W_a b$ the kernel function for particles a and b. Using this generic discretization, any function can be discretized using the integral interpolation and kernel function described in Section 2.1.

For a single degree of freedom (1D), using SPH interpolation and the Gaussian kernel function, see Equation 2.8, results in the following schematic:

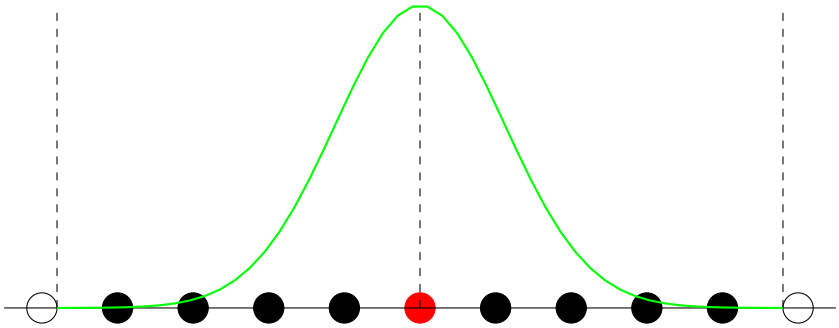


Figure 2.7: 1D SPH using SPH interpolation and Gaussian kernel function.

In this figure, the particle being interpolated is shown in the middle in red, the neighbouring particles in black, and the kernel function in green. Based on the properties of the neighbouring particles, any arbitrary function can then be calculated.

For example consider a strip of fluid with a length (L) of 10 m, a width (B) of 1 m, a depth (D) of 1 m, and a density of 1000 kg/m^3. The mass of the particles is based on $m = \frac{L \cdot B \cdot D \cdot \rho}{N-1}$, which results in a mass of 1000 kg for each of the particles. With a smoothing length (h) of 1.6, the following example values are found, see Table 2.2.

Table 2.2: Example data

Particle	x	r_5	q	W_{5b}	f_b
0	0	5	3.125	-	35
1	1	4	2.500	0.001	35
2	2	3	1.875	0.010	35
3	3	2	1.250	0.074	35
4	4	1	0.625	0.239	35
5	5	0	0	0.353	35
6	6	1	0.625	0.239	35
7	7	2	1.250	0.074	35
8	8	3	1.875	0.010	35
9	9	4	2.500	0.001	35
10	10	5	3.125	-	35

Using Equation 2.15, the value of 34.65 for f is found. This is slightly lower than the actual value 35. Due to rounding and discretization of the SPH. However, it is extremely close to the actual value. In this discretization, the particles own value was required, in most actual fluid functions, see Section 2.4, this is not required because the gradient of the kernel is zero at the origin.

2.2.1 Density

The density of the fluid, solid or gas is an important parameter that determines the eventual properties of the studied object. The density is commonly expressed using the following SPH formulation

(Monaghan and Gingold 1977):

$$\rho_a = \sum_b m_b \cdot W(\vec{r_{ab}}, h) \tag{2.16}$$

where $\vec{r_{ab}}$ is the distance between the two nodes, or $|\vec{r_a} - \vec{r_b}|$. In this equation, the density of one particle (a) is expressed as the sum of all other particles and the interpolation kernel (W). In practice not all other particles are used but only the neighbouring particles since the kernel functions tend to zero for larger values of q, see Section 2.1, due to their compact support requirement.

However, as described by Liu in (M. Liu and G. R. Liu 2010), this equation conserves mass exactly but has issues with low densities at the edges of the boundaries due to truncation. An alternative is the following normalization equation 2.17.

$$\rho_a = \frac{\sum_b m_b W_{ab}}{\sum_b \left(\frac{m_b}{\rho_b}\right) W_{ab}} \tag{2.17}$$

Another alternative is suggested by (Monaghan 1989), see equation 2.18. This is based on the time derivative of the density based on the gradient of the kernel and the velocity, instead of directly calculating the density.

$$\frac{d\rho_a}{dt} = \sum_b m_b \vec{v_{ab}} \cdot \nabla W_{ab} \tag{2.18}$$

where $\vec{v_{ab}}$ represents the difference in velocity between particle a and b or $\vec{v_{ab}} = \vec{v_a} - \vec{v_b}$. The expression ∇W_{ab} is the gradient of the kernel function with respect to particle a and b.

2.3 Integration

Various integration and stepping schemes have been used in literature. Many authors prefer simple first-order schemes. These simplistic schemes, such as an first-order forward Euler scheme, see Equations 2.19, are efficient to evaluate and thus allow for smaller time-steps.

$$\begin{aligned} \vec{v}_{n+1} &= \vec{v}_n + \delta t \vec{f}_n \\ \vec{r}_{n+1} &= \vec{r}_n + \vec{v}_n \delta t + \tfrac{1}{2}\delta t^2 \vec{f}_n \end{aligned} \tag{2.19}$$

With $\vec{v_n}$ the initial velocity at time-step n, δt the time-step, and $\vec{f_n}$ the acceleration at time-step n. This is an explicit scheme, requiring no iterative procedure to compute the state at the next time-step. Higher order, such as Runge-Kutta fourth-order, schemes have also been utilised for SPH (Antuono et al. 2015; Cherfils, Pinon, and Rivoalen 2012). While these higher-order schemes provide increased accuracy they also require multiple evaluations, resulting in longer compute times. Therefore, a balance between the order and speed is required, the leapfrog method (2.20) provides this balance (Monaghan 2005; Violeau and Rogers 2016).

$$\begin{aligned} \vec{v}_{\frac{1}{2}} &= v_0 + \tfrac{1}{2}\delta t f_0 \\ \vec{r}_1 &= \vec{r}_0 + \delta t v_{\frac{1}{2}} \\ \vec{v}_1 &= \vec{v}_{\frac{1}{2}} + \tfrac{1}{2}\delta t f_1 \end{aligned} \tag{2.20}$$

This type of leapfrog scheme is often described as a *drift-kick-drift* form. The kick is the change in velocity due to the force, and the drift is the change in position due to the velocity.

2.4 Hydrodynamics

Even though SPH was originally developed for use in astrophysics by (Monaghan and Gingold 1977), it has seen wide use in free-surface flow since it is original free-surface flow application by (Monaghan 1994).

This free-surface flow variant of SPH is a set of equations where the interpolation, see Section 2.1, and the discretization, see Section 2.2, have been applied to hydrodynamic equations, or Navier-Stokes equations.

In general, two (main) base sets of equations for SPH in fluids exist namely, Weakly-Compressible SPH (WCSPH) and Incompressible SPH (ICSPH). In this section only these two methods will be described, however, many more fluid methods exist in literature such as cSPH (Colagrossi and Landrini 2003), δ-SPH (Moltini and Colagrossi 2009), and CSKSPH (Frontiere, Raskin, and Owen 2017).

2.4.1 WCSPH

Weakly-Compressible SPH or *WCSPH*, as the name suggests, assumes that the fluid to be slightly (weakly) compressible. This SPH implementation or method is the original free-surface method using SPH (Monaghan 1994). Many extensions, corrections and modifications have been suggested in literature, see Chapter 3. In this section, the original implementation as described in Monaghan 1994 will be presented.

Governing Equations

For any free-surface flow the governing equations are the Navier-Stokes equations. The flow is assumed to be inviscid, without viscosity, $\text{Re} \to \infty$. The Navier-Stokes equations for inviscid flow is split into two parts. The first, the continuity equation, see Equation 2.21, governs the rate of change in the density of the fluid.

$$\frac{d\rho}{dt} = -\rho \nabla \cdot \vec{v} \tag{2.21}$$

The second equation is the momentum equation, see Equation 2.22. This momentum equation expresses the acceleration of the fluid based on the pressure gradient.

$$\frac{d\vec{v}}{dt} = -\frac{1}{\rho}\nabla P + \vec{g} \tag{2.22}$$

Equation of State

An equation of state (EOS) relates the density of a fluid to the pressure of a fluid. In SPH, the Tait EOS was originally suggested by Batchelor 1967 and has been in common use ever since, Monaghan 1994; Cummins and Rudman 1999; M. Liu and G. R. Liu 2010; Becker and Teschner 2007 among others. This equation of state is shown in Equation 2.23.

$$P = B\left[\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right] \tag{2.23}$$

Monaghan suggested using $\gamma = 7$ and this value has been maintained in most subsequent papers Monaghan and Gingold 1977; Stansby 2018; Violeau and Rogers 2016; Lee et al. 2008; Crespo 2008 among others. This value for γ causes the pressure to respond strongly to small variations in density. However, this can cause large fluctuations in pressure estimates using SPH. Furthermore, it is well established that attractive forces between particles can lead to instability in SPH (Morris, Fox, and Zhu 1997). Therefore, the subtraction can be omitted to ensure only positive pressures can occur in the simulation.

Continuity

As described in Section 2.2.1 the density in SPH is commonly taken as the summation of the mass of the surrounding particles multiplied by the kernel function. This equation conserves mass exactly, however, in free surface scenarios the surface particles have less neighbours which results in a lower and erroneous density near the surface (Becker and Teschner 2007). Therefore, an alternative approach is suggested by Monaghan (Monaghan 1994) by solving the continuity equation, see Section 2.4.1, which results in the following equation for the rate of change of mass of particle a.

$$\frac{d\rho_a}{dt} = \sum_b m_b \vec{v_{ab}} \cdot \nabla_a W_{ab} \tag{2.24}$$

where the rate of change of the mass of particle a is dependent on the mass of particle b, the velocity difference between particle a and b, and the gradient of the kernel with respect to particle a.

Momentum

The acceleration of the particle is determined using the momentum equation, see Section 2.4.1. This leads to the following SPH approximation for that momentum equation, see Equation 2.25 (Monaghan 1994).

$$\frac{d\vec{v_a}}{dt} = -\sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} \right) \nabla_a W_{ab} + \vec{g} \tag{2.25}$$

where, $\vec{g}$ the external (gravity) force, P_a and P_b the pressure at particle a and b, respectively. Again this function depends on the gradient of the kernel, and returns a vector.

Following Price 2012, this continuity equation is more stable when applied in its anti-symmetric form, see Equation 2.26. This can be done since the gradient of an even function is an odd function (Violeau and Rogers 2016).

$$\frac{d\vec{v_a}}{dt} = \sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} \right) \nabla_b W_{ba} + \vec{g} \tag{2.26}$$

In Equation 2.26 the gradient is not taken with respect to particle a but with respect to particle b and the kernel is taken from b to a (W_{ba}).

Artificial Viscosity

The flow modeled by Monaghan 1994 is an inviscid flow, having no viscosity (Re $\to \infty$). However, this also means there is no damping in the entire system, which can lead to instability. Therefore, an artificial viscosity is introduced, this artificial viscosity is the only dispersive quantity in the whole WCSPH set of equations. The artificial viscosity is expressed as follows, see Equation 2.27.

$$\Pi_{ab} = \begin{cases} -\frac{\alpha h (c_a + c_b)}{\rho_a + \rho_b} \left(\frac{\vec{v_{ab}} \cdot \vec{r_{ab}}}{\vec{r_{ab}^2} + \eta^2} \right), & \vec{v_{ab}} \cdot \vec{r_{ab}} < 0 \\ 0, & \vec{v_{ab}} \cdot \vec{r_{ab}} \geq 0 \end{cases} \tag{2.27}$$

with $\eta^2 = 0.01h^2$ and α a problem dependent quantity, often taken as 0.5 (Monaghan 1994; Monaghan 2005; Violeau and Rogers 2016). The small η factor is added to prevent singularities in the fraction. Monaghan 1994 proposed for this to be added to the momentum equation, resulting in the following (complete) momentum equation.

$$\frac{d\vec{v_a}}{dt} = -\sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} + \Pi_{ab} \right) \nabla_a W_{ab} + \vec{g} \tag{2.28}$$

2.4.2 ICSPH

In-Compressible SPH or *ICSPH* opposed to WCSPH assumes the fluid to be in-compressible and the density fixed. This can be a valid approximation since water or any other fluid is highly in-compressible, requiring extremely high pressures to even compress slightly. It has been developed in Cummins and Rudman 1999 and offers several advantages over 'traditional' weakly-compressible SPH. It reduces random motion of particles due to large pressure fluctuations (Lee et al. 2008) and results in smoother pressure and velocity fields. Lastly, it is shown to reduce CPU time by a factor of 2-20 (Lee et al. 2008) due to the elimination of the restrictive CFL timestep condition,

see Section 3.4 (Cummins and Rudman 1999). However, in contrast to WCSPH, the density is not obtained directly instead it requires the solving of the Pressure Poisson Equation (PPE) using a system of linear equations. Following Cummins and Rudman 1999 the PPE can be solved exactly or approximately. However, when solving exactly the pressure displays a checkerboard pattern, therefore, the approximate solution is required, see Equations 2.29 (Cummins and Rudman 1999; Lee et al. 2008).

$$\nabla^2 p_a \approx \frac{2}{\rho_a} \sum_b m_b \frac{(p_a - p_b)\vec{r}_a \cdot \nabla_a W_{ab}}{r_{ab}^2 + \eta^2} \tag{2.29}$$

While this linear system can be solved using traditional methods such as GMRES or Bi-CGSTAB (Violeau and Rogers 2016), it introduces significant complexity in the implementation of SPH. Furthermore, the pressure at the surface needs to be set to zero, therefore, surface particle tracking is required. This surface particle tracking is complicated and introduces an additional component of complexity in the implementation. In order to prevent surface tracking and reduce complexity Skillen et al. 2013 developed a method without the required separate surface particle tracking, see Equation 2.30. This system of linear equations can also be solved using the traditional methods such a Bi-CGSTAB.

$$a_{ii} P_i + \sum_b \alpha_i a_{ij} P_j = \alpha_i b_i \tag{2.30}$$

with a_{ij} being the coefficients in the PPE. The summation is performed for the neighbouring particles, excluding particle i itself. The coefficient α exists for the free-surface boundary condition, this coefficient is described by the following Equation 2.31 (Skillen et al. 2013).

$$\alpha = \begin{cases} 0, & \nabla \cdot \vec{r} \leq 1.4 \\ \frac{1}{2}\left(1 - cos\left(\frac{\nabla \cdot \vec{r} - 1.4}{1.6 - 1.4}\right)\right), & 1.4 < \nabla \cdot \vec{r} < 1.6 \\ 1, & \nabla \cdot \vec{r} \geq 1.6 \end{cases} \tag{2.31}$$

2.5 Boundary Conditions

Simulating and modeling fluids requires containment by solid or dynamic boundaries. These boundaries are formed by introducing particles that exhibit repulsive forces. These repulsive forces contain the fluid particles and simulate a solid wall by introducing an extra force that repels particles as they get close to the boundary.

Monaghan 1994 suggested the use of a boundary force that increases as distance between the boundary and SPH-particles decreases. This equation is based on the distance between the particle and the boundary (r) and the initial spacing between particles (r_0). This boundary force is a repelling force as described in Section 2.5.

$$f(r) = \begin{cases} D\left[\left(\frac{r_0}{r}\right)^{p_1} - \left(\frac{r_0}{r}\right)^{p2}\right]\frac{\vec{r}}{r^2}, & r \leq r_0 \\ 0, & r > r_0 \end{cases} \tag{2.32}$$

Since this force only exists when $r \leq r_0$, this force is purely repellent. Furthermore, the constants p_1 and p_2 should be chosen such that $p_1 > p_2$. Following Monaghan 1994, the value $p_1 = 4$ and $p_2 = 2$ result in satisfying results. However, this introduces (extra) forces in the SPH equation which can lead to instabilities (Violeau and Rogers 2016).

Chapter 3

Problems and Solutions

A plethora of corrections, modifications, extensions and parameter values exist for SPH. In this section, common problems and corrections for SPH, more specifically the WCSPH implementation of SPH, will be discussed, and solutions provided in literature will be presented for these common problems.

3.1 Kernel Correction

As described in Section 2.1, a nearly infinite number of kernel functions exist. These kernels can suffer from truncation near boundaries and free surfaces due to the absence of neighbouring particles. Following Bonet and Lok 1999, the kernels can be corrected for this truncation. The corrected kernel ($\tilde{w}_{ab}$) can either be corrected using a first-order moving-least-squares correction (MLS), or a zeroth-order Shepard filter, see Equation 3.1.

$$\tilde{W}_{ab} = \frac{W_{ab}}{\sum_b W_{ab} \frac{m_b}{\rho_b}} \tag{3.1}$$

A (slightly) more computationally intensive and higher-order correction is the MLS corrected kernel (Dilts 1999). Here, the kernel is corrected using a moving-least-squares approximation, and is, therefore, first-order correct. This corrected form of SPH, using an MLS correction, is often referred to as cSPH (Colagrossi, Lugni, and Brocchini 2010).

Not only the kernel should be corrected, also its gradient should be corrected (Bonet and Lok 1999). The gradient can be corrected using a correction matrix L, this matrix is determined using the following Equation 3.2.

$$L_a = \left(\sum_b \frac{m_b}{\rho_b} \nabla_a W_{ab} \otimes (\vec{r}_b - \vec{r}_a) \right) \tag{3.2}$$

This gradient correction ensures that any linear velocity field is evaluated exactly, and angular momentum preserved (Crespo 2008; Bonet and Lok 1999). However, this approach of kernel corrections results in a deviance of the required kernel symmetry (Vaughan et al. 2008), see Section 2.1 kernel requirement 5.

3.2 Pressure

The pressure field in WCSPH exhibits large oscillations and can be noisy and inaccurate (Crespo 2008; Lee et al. 2008; Violeau and Rogers 2016). These spurious pressure oscillations have several different sources. Firstly, the stiff equation of state causes small difference in density to lead to large differences in pressure, see Section 2.4.1. Secondly, inaccuracies in the SPH interpolation procedure, and, finally, inaccuracies due to evaluation of all equations in the same computational points (Violeau and Rogers 2016).

A possible solution to these density fluctuation is the use of an in-compressible method, such as ICSPH (Lee et al. 2008), see Section 2.4.2. However, this is not the only solution to the pressure fluctuations, density filters can be applied to the WCSPH method.

Density Filtering

Density filter, or also called density re-initialisation, re-assigns a density to each particle. This filtering is applied every 20-30 time-steps to ensure correct densities of the particles, see Equation 3.3 (Violeau and Rogers 2016; Crespo 2008).

$$\rho_a^{new} = \sum_b \rho_b \tilde{w}_{ab} \frac{m_b}{\rho_b f} = \sum_b m_b \tilde{w}_{ab} \tag{3.3}$$

With $\tilde{w}_{ab}$ being the corrected kernel, see Section 3.1. This is equivalent to taking the summation density, see Equation 2.16, with a corrected kernel.

3.3 Clustering

SPH particles often show clustering behaviour, especially when negative pressure occurs, this leads to instability in tension (Morris, Fox, and Zhu 1997). This is caused by the kernel function's instability in tension, this instability in tension causes the clustering of particles. This clustering is especially apparent in particles that utilise an equation of state, which results in negative pressure (Crespo 2008). In real particles this clustering would be prevented by repulsive molecular forces between atoms.

The first possible solution to prevent clustering issues is the usage of Wendland or quintic kernels, see Section 2.1. These Wendland kernels do not suffer from the instability in tension, depicted by other kernels (Violeau and Rogers 2016; Dehnen and Aly 2012; M. Liu and G. R. Liu 2010). This instability is caused by a negative second derivative of the kernel in these situations.

Following Monaghan 2000 this tensile instability can also be corrected by introducing an artificial pressure. This artificial pressure should increase when the distance between the particles decreases and be based on the kernel function, see Equation 3.4.

$$f_{ab} = \frac{W(q)}{W(\Delta p)} \tag{3.4}$$

with Δp being the average particle spacing divided by the smoothing length h. Based on this function a tensile correction factor (cf_{ab}^n) is created and added to the momentum equation, see Equation 3.5.

$$\frac{d\vec{v_a}}{dt} = -\sum_b m_b \left(\frac{P_a + cf_{ab}^n}{\rho_a^2} + \frac{P_b + cf_{ab}^n}{\rho_b^2} + \Pi_{ab} \right) \nabla_a W_{ab} + \vec{g} \tag{3.5}$$

The dimensionless parameter c is typically 0.2 for $P_a < 0$ and 0.01 for $P > 0$. This slightly positive values for c when the pressure is positive causes a slightly higher pressure, which prevents fluid particles from forming local linear structures, with a pressure difference of less than 1% (Monaghan 2000).

3.4 Time-stepping

In order to prevent instability due to time-stepping a variable time-step is commonly used that is based on three criteria (Monaghan 1994; Violeau and Rogers 2016). For each particle the conditions are evaluated and the minimum of all the criteria is used to find the maximum time-step for a given state. Further modification to these time-step controls are presented in D. Violeau and Leroy 2014, but are outside the scope of this thesis. Because time-stepping is such a crucial factor in the overall computational cost other avenues have also been explored such as regional time-stepping (RTS), where different time-steps are used for individual cells (Goswami and Batty 2014).

3.4.1 CFL

The traditional Courant-Friedrichs-Lewy (CFL) stability condition for SPH is captured in the following equation.

$$\Delta t_{cfl} = 0.25 \frac{h_a}{c_a} \tag{3.6}$$

The CFL condition enforces that no information traveling at the speed of sound (c_a), the maximum speed in the fluid, can travel further than the discretization distance (h). For each particle (a) this CFL condition has to be evaluated and the minimum taken. This CFL condition does not take particle motion into account, therefore, additional time-step constraints are required.

3.4.2 Diffusion

The maximum allowable time-step due to diffusion of particles is expressed in Equation 3.7.

$$\Delta t_d = 0.125 \frac{h_a^2}{\max v} \tag{3.7}$$

This expression has to be evaluated for each single particle and the minimum taken.

3.4.3 External Force

Particle almost always have at least one external force, such as gravity, acting upon them. This causes the particles to accelerate and move quicker than anticipated by either CFL or diffusion criterion, therefore, the following equation is introduced to capture the maximum time-step based on external force.

$$\Delta t_f = 0.25\sqrt{\frac{h_a}{\max|\vec{F}_a|}} \tag{3.8}$$

The factor for this equation has been obtained experimentally. Therefore, different suggested values have been published. A factor of 0.25 has been used by Morris, Fox, and Zhu 1997; Violeau and Rogers 2016, a factor of 0.30 has been used by Lattanzio et al. 1985.

3.5 Boundary Penetration

While boundaries are still an open problem in SPH, several methods have been suggested for the creation of (rigid) boundaries in SPH (Violeau and Rogers 2016; Crespo 2008). For rigid walls, three basic concepts exists:

1. Repulsive Forces
2. Fictitious Particles
3. Ghost Particles

The repulsive forces try to prevent boundary penetration and simulate a solid wall by creating an (extra) force that repels particles as they get close to the boundary. Fictitious particles are regular fluid particles that are fixed in space. Ghost particles are particles that are only created when fluid particles get close to the rigid wall.

3.5.1 Boundary Force

As described in Section 2.5, boundary forces prevent solid wall penetration by introduction of a (strictly) repulsive force that depends on the distance between the particles and the boundary.

3.5.2 Fictitious particles

These particles adhere to the same criteria and equations as the 'regular' fluid particles, however, their position is prescribed. These particles provide the simplest implementation of a boundary since no extra equations are necessary. However, boundary penetrations can occur when an existing velocity is placed on the fluid particles.

3.5.3 Ghost Particles

Ghost particles are only created when the fluid particles get near the boundary, which is defined by a distance less than the smoothing length (h). These ghost particles are equal to the neighbouring fluid particle, however, with an opposing velocity. This results in a changing number of particles at every time-step, creating extra complexity, however, only the required number of particles are evaluated.

3.6 Group Velocity

In order to promote stability Monaghan 1989 introduced a velocity correction, called XSPH. It dampens the velocity using a small factor (ε), this small factor is typically equal to 0.5 (Monaghan 2005).

$$\frac{d\vec{r}_a}{dt} = \vec{v}_a - \varepsilon \sum_b \frac{m_b \vec{v}_{ab}}{\frac{1}{2} \cdot (\rho_a + \rho_b)} \tag{3.9}$$

This correction, see Equation 3.9, keeps particles more orderly and prevents penetration (Monaghan 1994). However, it has also been criticized for introducing another (fictional) damping term without addressing the root cause of the instability (Violeau and Rogers 2016). Furthermore, it introduces an extra complexity since each particle now has two velocities: the uncorrected velocity from the momentum equation, and the corrected velocity for XSPH (Monaghan 1994).

3.7 Smoothing Length

The kernel smoothing lengths are generally experimentally determined constants, and range from around 1.0 to 1.6 (Shadloo et al. 2012; Monaghan 2005). For the cubic spline kernel, see Section 2.1, this smoothing length has to be in the range of 1.0 to 1.6 to ensure stability (Balsara 1995). These smoothing lengths are generally kept constant for the entire simulation. However, as particles get closer and move these smoothing lengths should be adjusted, which can be accomplished by implementing a non-constant smoothing length, see Equation 3.10 (Monaghan 2005).

$$h_a = \sigma \left(\frac{m_a}{\rho_a} \right)^{\frac{1}{d}} \tag{3.10}$$

with $\sigma = 1.3$ and d being the number of dimensions, either 1, 2, or 3 for 1D, 2D and 3D, respectively.

Chapter 4

Implementation

4.1 Introduction

As described in Section 1.2 and 1.3, the goal is to implement a re-usable framework for students and university staff members. Therefore, the code will be built in Python and use as many generic methods as possible. These generic methods are a contract or a template that allow the user to switch the implementation of the method itself, without changing anything where the method is used. They only dictate the input and output of a specific method without specifying the actual implementation.

Only the WCSPH method, see Section 2.4.1, will be implemented. WCSPH is the most mature SPH method. It has been in use since the original paper and has seen constant development since (Monaghan 1994). The goal of this framework is not to be an exhaustive SPH simulator but to serve as a framework where users can implement their own methods and implementation, therefore, it is sufficient to implement the WCSPH method.

The implementation is organized as follows, see Figure 4.1, several important files are highlighted.

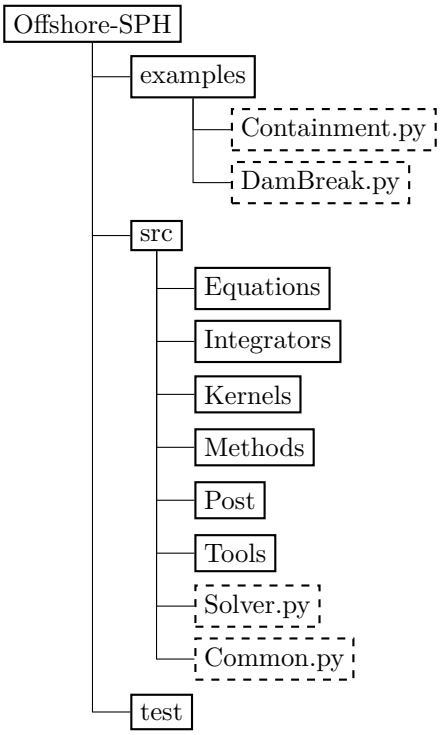


Figure 4.1: Organization of the implementation.

The main code of the program is contained in the *src* or source folder. The equations, such as the momentum equation, continuity equation, summation density equation, etc. are contained in the *Equations* folder. These equations are then utilized in the *Methods*. The solver (*Solver.py*) is the main executor, this combines the method, equations, tools, and integrator to run the simulation. *Common.py* contains important information about the structuring of particle arrays, and different particle types. The *test* folder contains tests to ensure a proper functioning of the code. This is a crucial part of any code base. Two test-cases have been created and are stored in the *examples* folder, these test-cases are further discussed in Section 5.2.

The actual implementation is publicly accessible on github[1]. It is available under the MIT License, allowing any user to use the software as-is.

4.2 Code

The goal of the implementation is to provide a shared code base that can be easily extend and modified by students and staff members at the TU Delft. Therefore, Python has been chosen as language of implementation.

The main loop of the solver is shown in Figure 4.2. As can be seen the main-loop consist of seven different phases or activities. These activities need to be carried out sequentially, and the content of the activity can be changed by the user. For example the integration method (Predict and Correct activities) can be changed by the user. Then this different integration method will be utilized when

[1] https://github.com/KoningJasper/Offshore-SPH

calling the predict activity. This is possible due to the use of templates or stensils, these dictate the entries and outputs of a function. As long as the function adheres to these requirements they can be swapped, regardless of the actual content of the function.

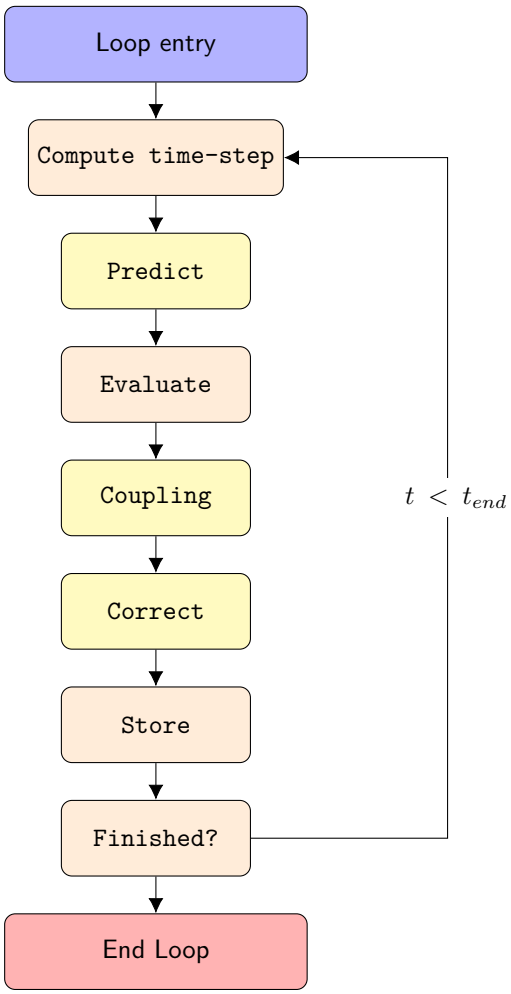


Figure 4.2: Schematic main-loop.

The highlighted (yellow) blocks displayed in the figure are user changeable depending on the implemented method and the following of the template.

4.2.1 Time-Step

The time-step largely dictates the time required for the computation of a single simulation. This time-step is implemented following the general SPH time-step rules set out in Section 3.4.

Besides the dynamic time-step condition, a possibility for a fixed time-step is also implemented. When the time-step is fixed, the dynamic time-step is not computed and instead the fixed time-step is used.

4.2.2 Evaluation

The evaluation of the particles is the main component of the implementation. As described in Section 4.2, several activities can be changed depending on the method, see Section 4.3, that has been used when calling the solver, these activities are highlighted in yellow. The entire evaluation procedure is schematically shown in Figure 4.3.

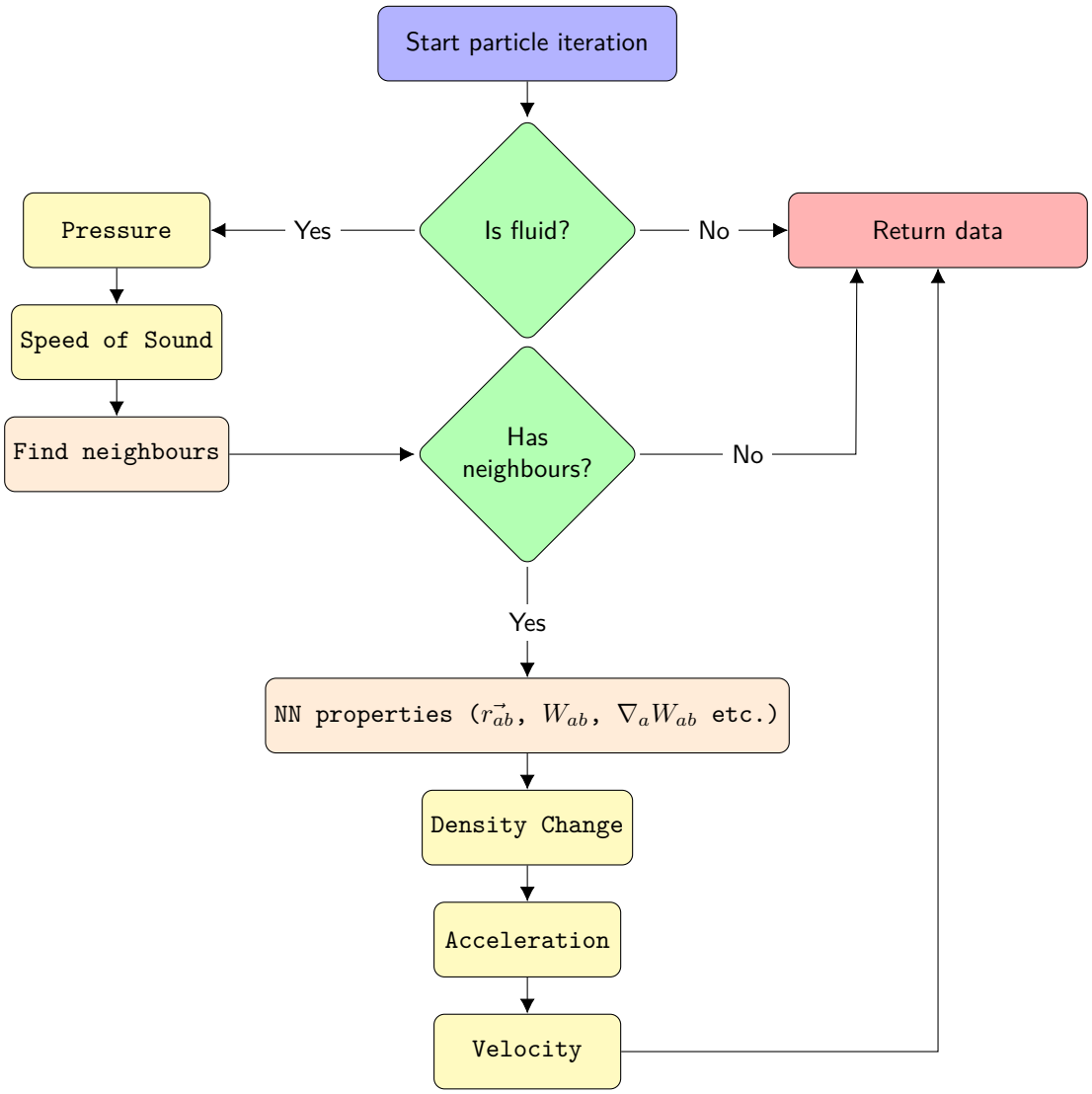


Figure 4.3: Schematic evaluation-loop.

4.2.3 Predict & Correct

The predict & correction processes, see Figure 4.2, predict a time-step and correct that prediction. They do not have to be implemented necessarily, for example an first-order euler integration method does not implement a predict stage only a correct stage, see Section 2.3.

For this implementation two integrators are implemented: a first-order Euler integrator, and the leapfrog integrator as described by Monaghan 2005, see Section 2.3. For actual usage it is advisable to utilise the leapfrog integrator since it is of a higher-order (Monaghan 2005).

4.3 Method

The WCSPH method, see Section 2.4.1, has been implemented since the goal of this implementation and program is to provide a starting point for the users. This method is the oldest and most mature SPH method. Furthermore, it should only serve as a starting point for future users. Therefore, the WCSPH method is sufficient for this application.

The method binds together several equations to calculate the required properties. The template for the method has six required functions (see Figure 4.3): pressure calculation, speed of sound computation, density change, acceleration computation, and velocity change. Each of these components can be switched by creating a new method.

By default the WCSPH method uses continuity to calculate the change in density for a given particle, by simply changing the density change function to the summation density function the method can be changed without altering any other equations of configurations.

This allows for extreme flexibility when testing or creating equations for use in SPH without having to change any other parts of the simulation. This is possible due to the implementation of the templates.

4.4 Neighbourhood

A crucial component of the SPH implementation is the determination of the neighbouring particles (neighbourhood), often shown as b in equations. A naive implementation would be to calculate the distance between each particle and only take ones that fall within the kernel's range. However, as the number of particles increase, this becomes prohibitive since N^2 calculations are required at every time-step to compute the full distance matrix. Therefore, a so called neighbourhood search is required.

Following Monaghan 2005, the domain is divided into cells, for each particle the cell is then determined. Based on a particle's cell, all particles in the surrounding eight cells are evaluated for distance, and only the ones that fall within the kernel's range are kept, see Figure 4.4. This reduces the number of operations required to $N\log N$.

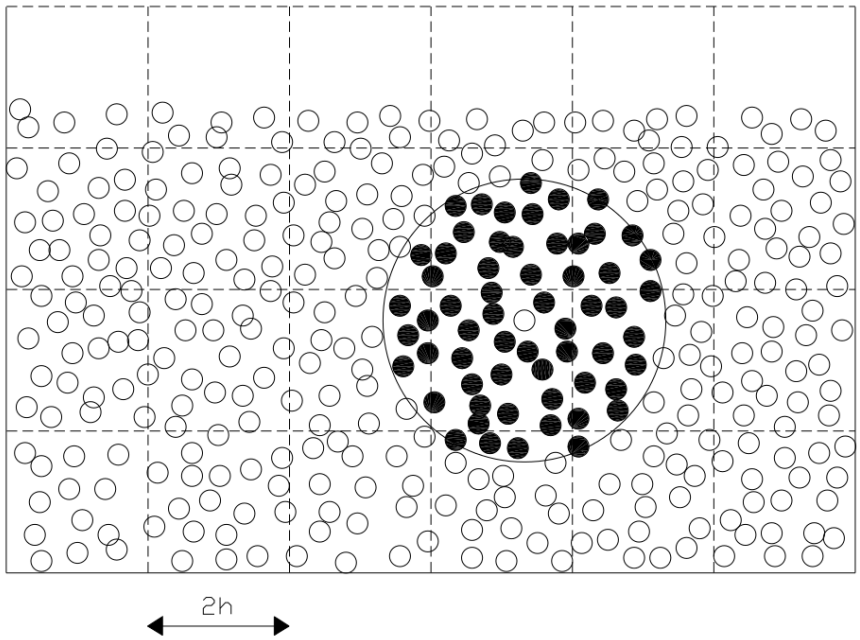


Figure 4.4: 2D neighbouring particles and kernel's range (Crespo 2008).

The internal bookkeeping of these cells is done using a cell linked-list (CLL) (Domínguez et al. 2010). This is an efficient method for the retrieval of neigbouring particles. When the neighbouring particles are retrieved, the distance is checked, since not all neighbouring particles are actually within the radius, see Figure 4.4.

4.5 Numba

Python is an high-level language, this means that it allows the user to express complicated instructions with only a few lines of code. Allowing the user to be more productive. However, due to it's high level nature and lack of compilation, python is also a relatively slow language. Being several orders of magnitude slower than C++ in computationally intensive operations.

Numba[2] tries to overcome these issues by compiling python in C++ to offer the same flexibility but with the speed of C++. Besides offering compiled speed-ups Numba also offers easy parallelization. Since SPH can be described as a 'stupidly parallel' class of software, it is trivially parallelizable making nearly linear speed gains on parallelization.

4.6 Coupling

In-order to improve relevance of the implementation and extend use to not only fluid interaction but also fluid-structure interaction, a coupling is implemented. This is essential for the modeling of problems such as ice-fluid breakage as modeled in Keijdener, Hendrikse, and A. Metrikine 2018. The coupling is achieved by creating of a new type of particle: coupled particles. These coupled

[2]http://numba.pydata.org/

particles behave like boundary particles in the main loop, they repel fluid particles to ensure the no-penetration constraint. Furthermore, no acceleration is computed for these boundary particles, instead these accelerations are computed by the user-defined coupling. To enable the user-defined coupling to compute these accelerations the entire state is transferred, where the acceleration is computed and the complete state returned. After-which the main loop is resumed, see Figure 4.5 for a schematic overview.

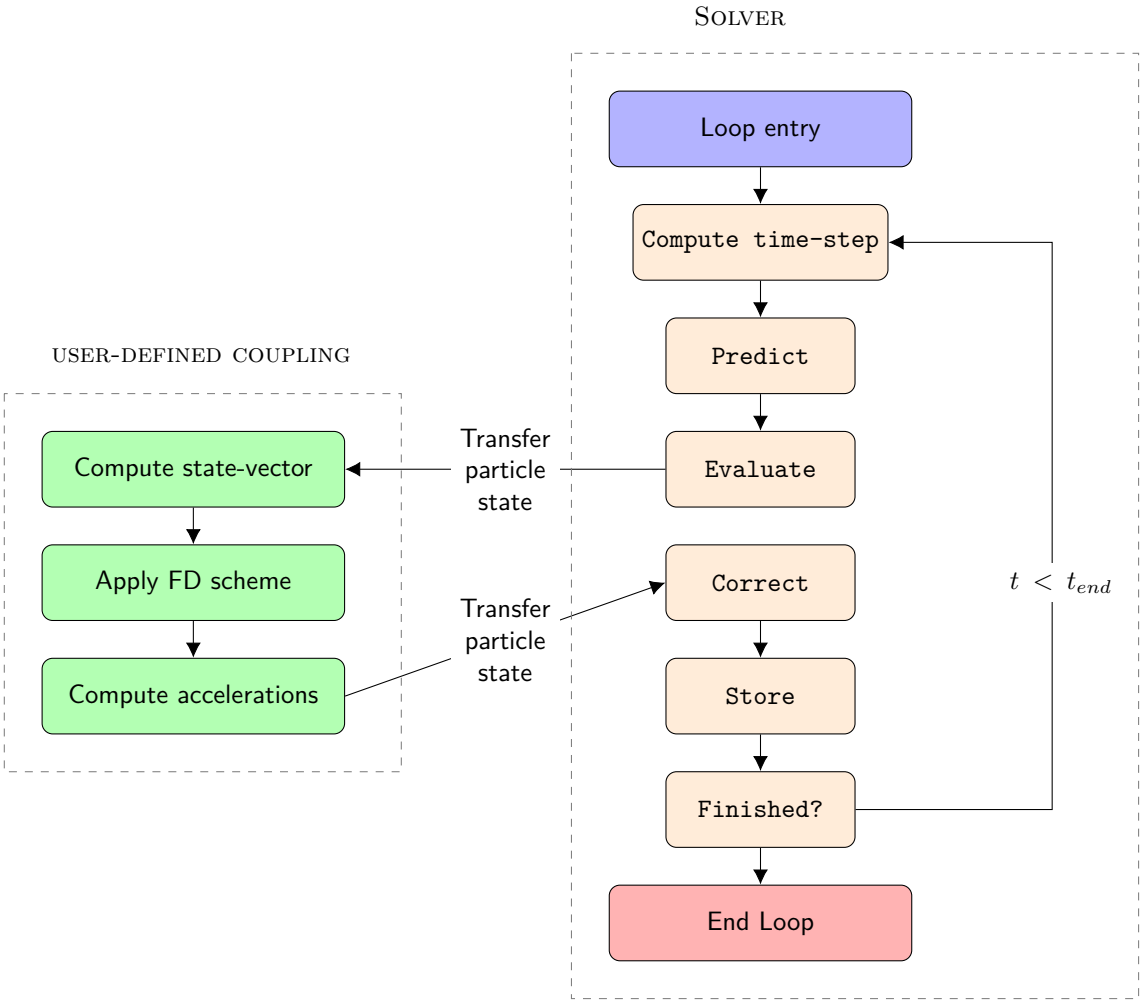


Figure 4.5: Schematic user-defined (FD) coupling integration with main loop.

The user-defined coupling can be a simple FD scheme, as seen in Figure 4.5, or any user-defined function that returns accelerations for the coupled particles.

4.7 Post-processing

A crucial part in the extraction of results from a simulation is post-processing. To help the user, a default animation exporter has been created. This allows the user to view the results of the simulation in an easily digestible way, adjusting parameters and positions to get the desired results. For more detailed post-processing certain properties of the particles, such as pressure, density, position, can be saved to disk by default. Custom properties can be saved to disk when desired by the user, see Figure 4.6.

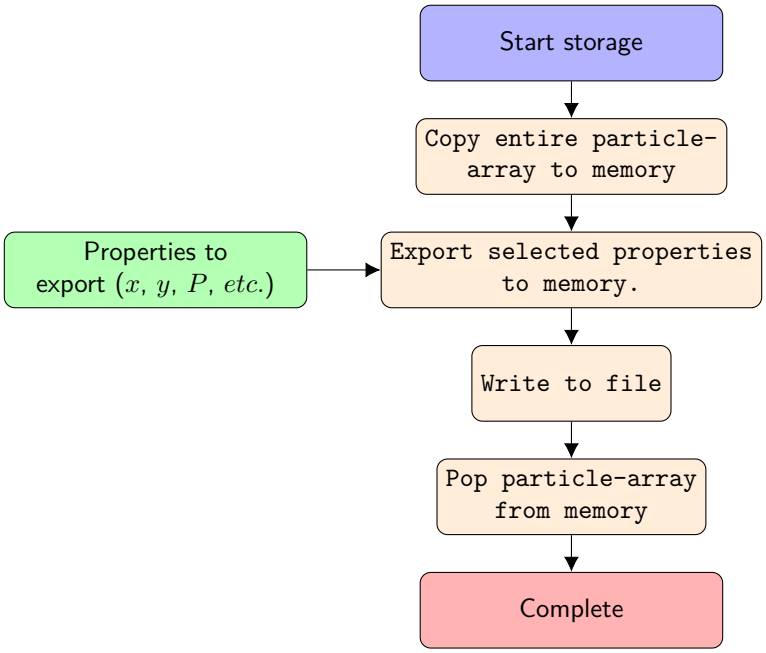


Figure 4.6: Schematic storage procedure.

The schematic, shows that the user-input (properties to export) is taken and exported. Any property contained in the particle-array can be exported to a file. The user can then use these properties in post-processing to calculate kinetic energy, for example. The pop particle-array from memory is carried out periodically when a certain amount of time-steps has been reached, by default 500. This means that the results of up to 500 time-steps is kept in memory continuously. This helps conserve memory during the simulation. Furthermore, the write to file is also only carried out periodically to speed-up processing, since writing to disk is slow. It can also be disabled completely, only exporting when the simulation is complete.

Chapter 5

Results

5.1 General

In general, the results acquired from the SPH simulations using the program are satisfactory. The particles behave as expected for a fluid, see also sections 5.2 and 5.2.1. Due to the modular setup of the implementation, a simulation is easy to set up, requiring few lines of code to produce adequate results.

The performance of the simulations are adequate, performing the dam-break, see Section 5.2, in around one hour on a (modern) laptop and the containment study, see Section 5.2.1, in around 30 minutes. More powerful hardware will reduce computational time, see also Section 5.3.

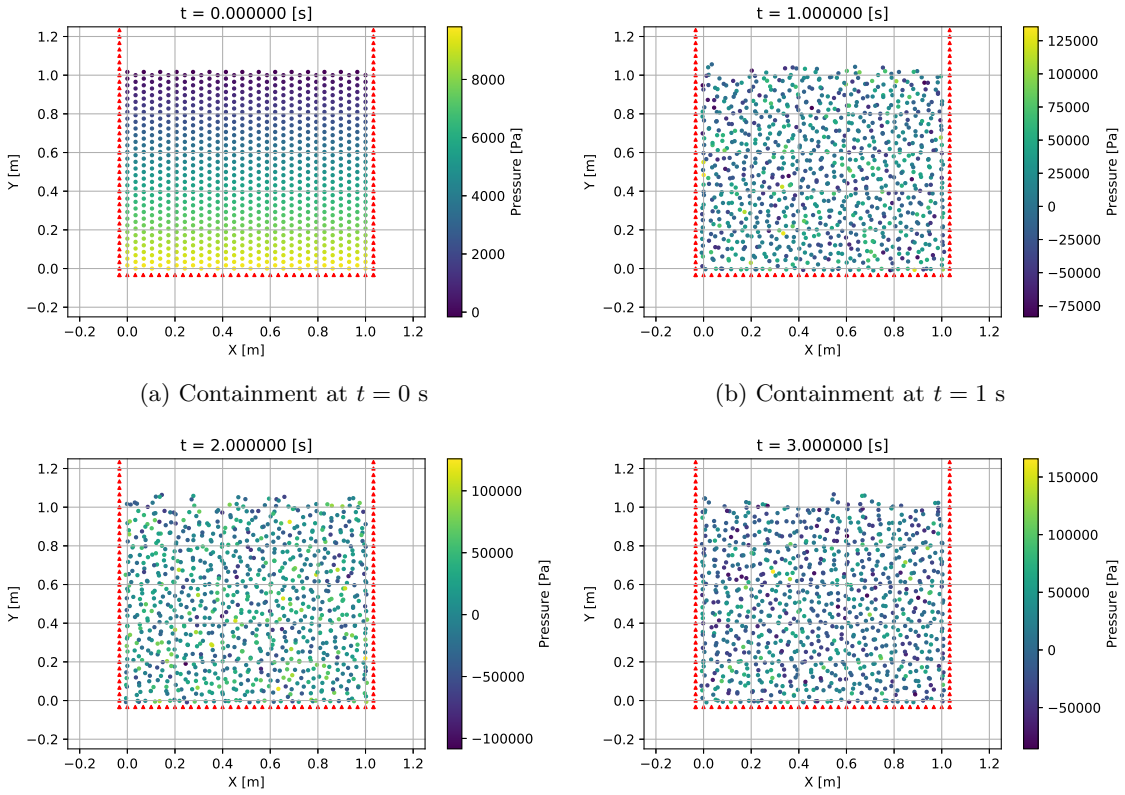
Significant pressure fluctuations occur in any simulation due to the factors mentioned in Section 3.2. Implementing more sophisticated methods such as ICSPH, and/or tensile correction could prevent this. However, these methods are outside of the scope of this implementation.

5.2 Verification

In-order to assess the validity of the implemented model, two verification studies have been carried out. In these verification studies, the results of the model are verified against known data. The first verification study is a confinement study. In this study, a box of water is at rest and is confined by borders on all sides. The second verification study is a dam-break which is a standard method for fluid verification.

5.2.1 Containment

In the containment study, the behaviour of the fluid itself and the borders is verified. A vessel full of fluid should neither expand nor contract and should not penetrate the boundaries. As described in Section 2.5, the solid walls are simulated by particles and a boundary force. This boundary force should prevent penetration. In-order to assess the model, a 1 m $\times$ 1 m box of fluid is created using 900 particles (30×30), and simulated for 3 seconds. This yields the following time-series, see Figure 5.1.



(a) Containment at $t = 0$ s

(b) Containment at $t = 1$ s

(c) Containment at $t = 2$ s

(d) Containment at $t = 3$ s

Figure 5.1: Development of fluid particles in containment over time.

As can be seen in Figure 5.1, the fluid behaves as expected. No boundary penetration occurs, and the top of the fluid stays more or less level. However, as described in Section 3.2, there are a lot of spurious pressure fluctuations. This also leads to extremely small time-steps following the time-step criteria laid out in Section 3.4, see Figure 5.2.

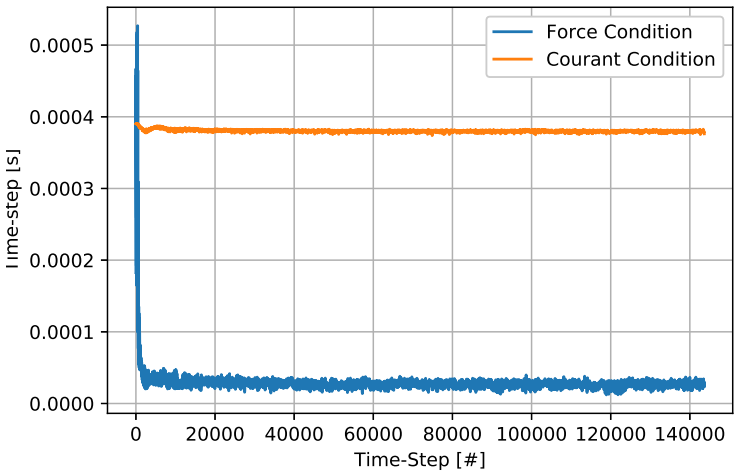


Figure 5.2: Containment time-step

This small time-step greatly increases the total time required for the simulation, requiring over 140 thousand time-steps. As can be seen from Figure 5.2, the force condition is the governing time-step for the entire simulation, this is likely due to the spurious pressure fluctuation described in Section 3.2, and the negative pressures that can be seen in Figure 5.1.

A secondary verification of the correctness of the simulation can be done by looking at the total kinetic energy of the system, the kinetic energy of the SPH particle can be estimated using the following equation (5.1).

$$KE = \sum_{n} \left(\frac{1}{2} m_n (v_{n,x}^2 + v_{n,y}^2) \right) \tag{5.1}$$

The total kinetic energy of the system of particles should be more or less constant over time, since the boundaries are static and the fluid is (almost) at rest. Calculating the kinetic-energy over time for the fluid, yields the following kinetic-energy over time, see Figure 5.3.

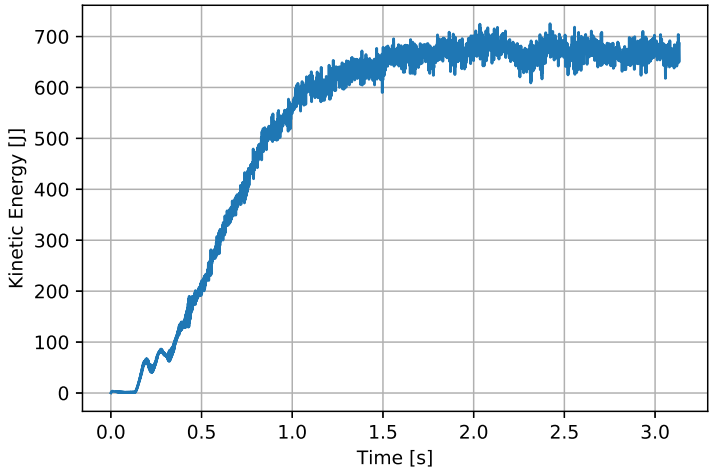


Figure 5.3: Kinetic-Energy of the system over time, in containment scenario.

From the figure it can be concluded that the requirement is met, from 1 s onward the kinetic-energy of the system reaches a stable plateau. This is expected since initially the particles are at complete rest, $\vec{v} = 0$, and the spurious pressure changes cause (small) movements, thus increasing KE of the system.

5.2.2 Dam Break

A dam break is a classic hydraulic verification case and is commonly used as a verification case in SPH literature (Monaghan 1994; Crespo 2008). A number of real world experiments exist for the dam-break, which makes it an attractive case to use for verification (Martin and Moyce 1952), allowing not only verification against previous models but also against experimental results.

In the comparison by Monaghan 1994, 2311 particles are used, resulting in an approximate fluid particle grid of 50 $\times$ 50. For the model proposed in this thesis a total of 2864 particles are used, a slight increase compared to Monaghan's implementation. The density of the fluid is assumed to be 1000.0 kg/m^3, and the fluid volume to be 625 L (25 m $\times$ 25 m). This results in the following configuration:

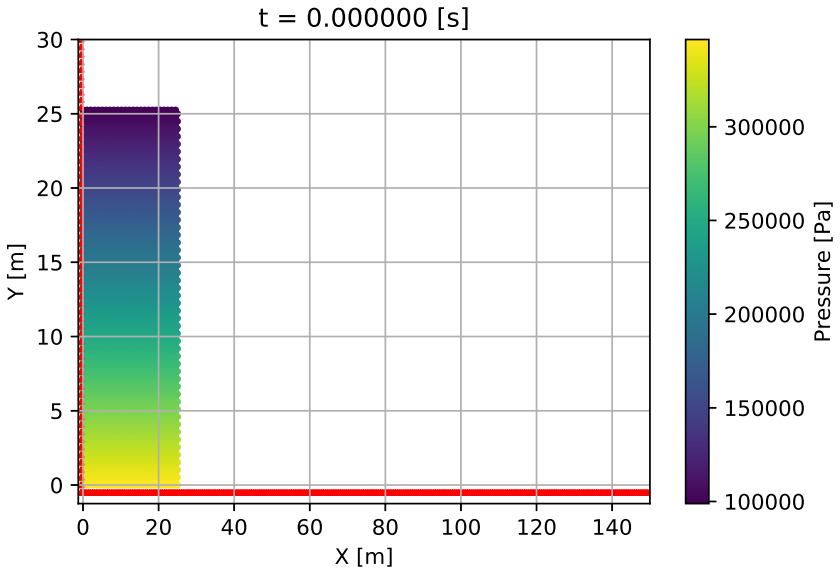


Figure 5.4: Dam-break configuration ($t = 0$ s)

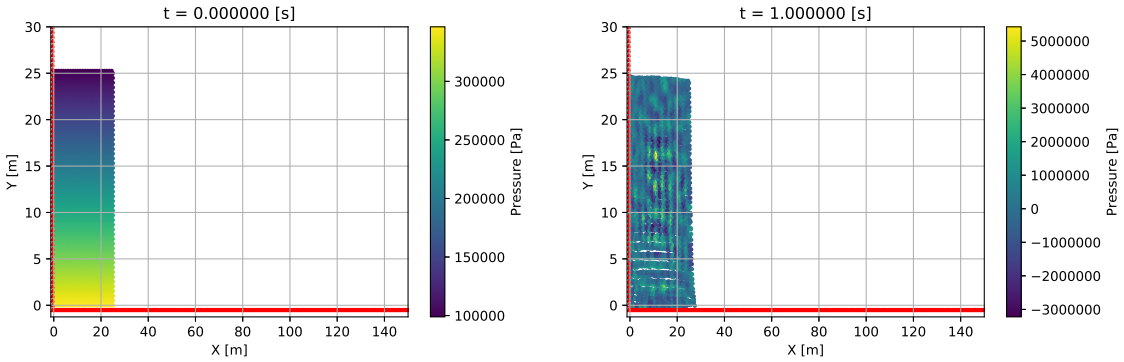
For this comparison two parameters are compared at certain intervals in time: the height of the fluid column (h) and the width of the fluid column (z). Both of these quantities are normalized for their original value, thus at rest both h and Z would be 1.0.

Table 5.1: Dam-break result comparison with experimental and other SPH code.

	Experimental		**Monaghan 1994**		**Proposed Model**	
t	h	z	h	z	h	z
0.71	0.90	1.33	0.90	1.56	0.96	1.51
1.39	0.76	2.25	0.75	2.55	0.81	2.44
2.10	0.57	3.22	0.56	3.75	0.69	3.74
3.10	0.32	4.80	0.37	5.00	0.44	5.66

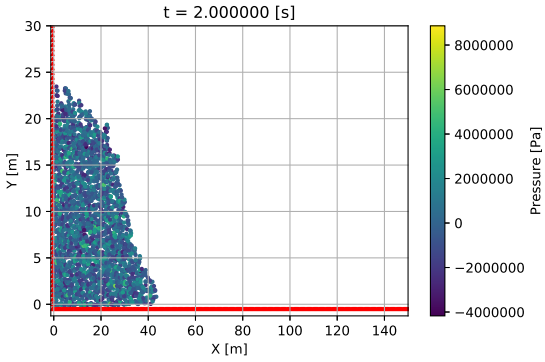
As can be seen from Table 5.1, the results for the vertical direction (h) match very closely between the experiment and the simulation. Furthermore, the proposed model is extremely close to the simulation as implemented in Monaghan 1994. The results in the horizontal surge direction (z) show slight differences between the simulation and experiment. Monaghan 1994 proposed that this is due to the lack of friction in the SPH model.

Analogous to the containment scenario, a time-series has also been developed for the dam-break scenario, see Figure 5.5. This time-series shows that there are significant outliers at $t = 4$ s and $t = 5$ s. Furthermore, it displays the spurious pressure changes described in Section 3.2.

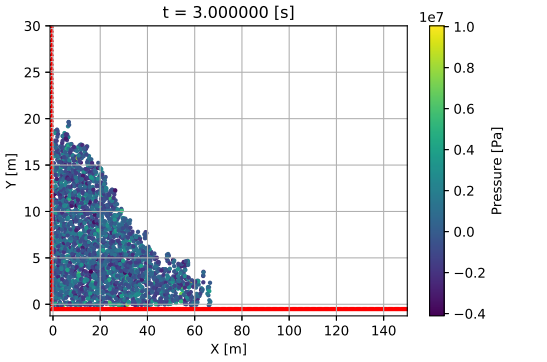


(a) Dam-break at $t = 0$ s

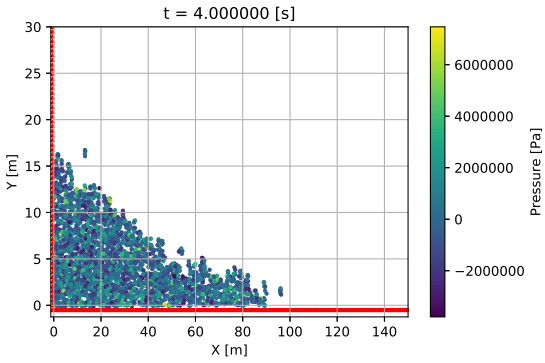
(b) Dam-break at $t = 1$ s



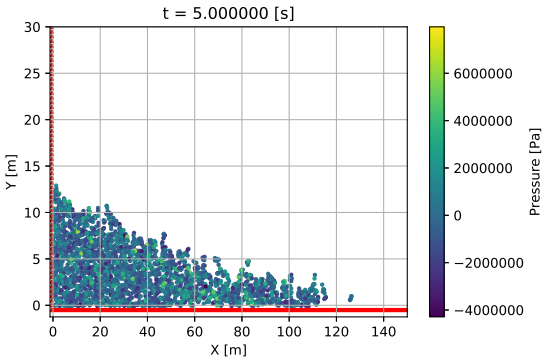
(c) Dam-break at $t = 2$ s



(d) Dam-break at $t = 3$ s



(e) Dam-break at $t = 4$ s



(f) Dam-break at $t = 5$ s

Figure 5.5: Development of fluid particles in dam-break scenario over time.

At $t = 5$ s significant outliers are visible, these six outliers skew the results and cause a significant deviation in z-value. These particles are highlighted in Figure 5.6, with the outliers highlighted in red and the border presented in black for enhanced contrast.

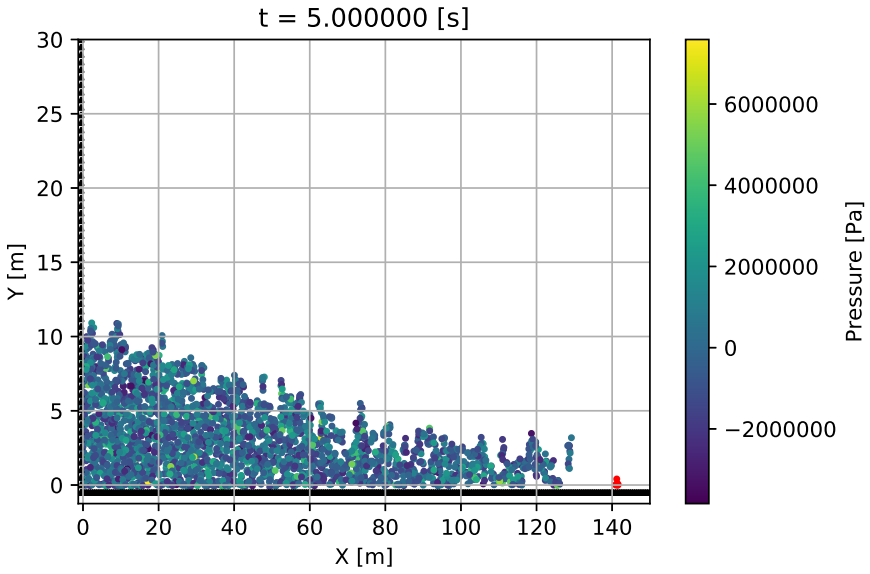


Figure 5.6: Dam-break with highlighted outliers.

Removal of these outliers improves the results, see Table 5.2, for the entire simulation with a total z difference of -1.4%.

Table 5.2: Outlier corrected dam-break result comparison.

	Experimental		**Monaghan 1994**		**Corrected Impl.**	
t	h	z	h	z	h	z
0.71	0.90	1.33	0.90	1.56	0.96	1.51
1.39	0.76	2.25	0.75	2.55	0.81	2.44
2.10	0.57	3.22	0.56	3.75	0.69	3.51
3.10	0.32	4.80	0.37	5.00	0.44	5.17

This correction causes a decrease only in the Z for the implemented dam-break results. This is expected since particles removed are only relevant for the z parameter.

5.3 Performance

While the performance of the code is far off current cutting edge graphics processing unit (GPU) SPH code, such as fluids v.3 (Hoetzlein 2014), the focus of the application is the adaptability to new equations and having a sufficient speed and detail required for this. Furthermore, the focus on 2D simulations which require far less particles then 3D simulations, therefore, the current performance is sufficient for the goals set out in the objectives, see Section 1.2.

The majority of the computational time is spent on the near-neighbourhood search, see Section 4.4, this is in-line with the literature. Implementing a more sophisticated near-neighbourhood search algorithm such as a Verlet-Linked-List (Winkler, Rezavand, and Rauch 2018) could lead to

performance gains.

Counter-intuitively, the computational time can decrease when the number of particles increase, this is due to the spurious pressure changes in WCSPH. The spurious pressure changes occur due to the stiff equation of state in which a small difference in velocity can lead to a large difference in density and pressure, see Section 3.2. Increasing the number of particles increases the particles in the neighbourhood which reduces the chance of negative pressure since other neighbourhood particles contribute to the (positive) pressure at the particle.

When pressure is forced to be positive, by implementing Equation 5.2 at the end of the integration, the resulting simulation is several factors faster. Improving the dam-break performance from around 30 minutes to around 20 minutes. This indicates that the tensile instability combined with the dynamic time-stepping create significant noise and extremely small time-steps. However, this is not a suitable solution since it causes energy increases and inconsistencies. Implementation of the tensile corrections and pressure corrections as described in Sections 3.2 and 3.3.

$$\tilde{\rho_a} = \begin{cases} 0 & \rho_a < 0 \\ \rho_a & \rho_a \geq 0 \end{cases} \tag{5.2}$$

Chapter 6

Ice Structure Interaction

6.1 Model

Keijdener, Hendrikse, and A. Metrikine 2018 developed a semi-analytical model (Figure 6.1), for the analysis of the effects of hydrodynamics on the ice-structure interactions. The model consists of an semi-infinite Kirchhoff-Love plate, representing the ice-sheet, floating on an infinite wide layer of fluid of depth H. The ice-sheet moves towards the structure with a constant velocity V_{ice}. The structure is assumed to be rigid, immovable, and have no interaction with the fluid. This structure is represented by a line passing through the origin, having angle θ with respect to the x-axis, see Figure 6.1. The forward movement of the ice, with speed V_{ice} causes it to slide down the structure until it fails in bending.

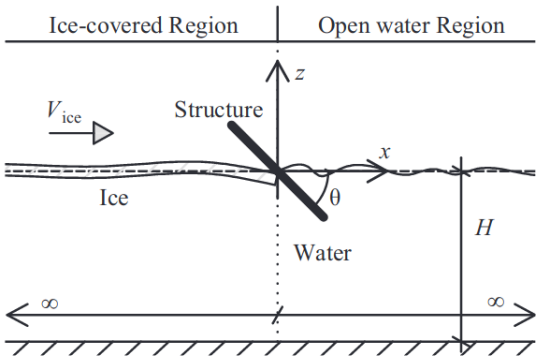


Figure 6.1: Original ice-model (Keijdener, Hendrikse, and A. Metrikine 2018).

6.1.1 Equations of Motion

The ice-sheet is modeled as an Euler-Bernoilli (EB) beam, in this thesis this beam is modeled different than described by Keijdener, Hendrikse, and A. Metrikine 2018, completely free at one end and clamped at the other, and allows deformation across the length of the beam, see Figure 6.2.

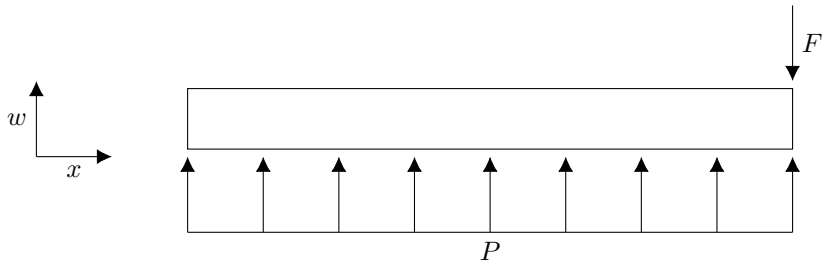


Figure 6.2: Beam

Based on the figure, the following equation of motion for the EB beam is derived:

$$M\frac{\partial^2 w}{\partial t^2} + c\frac{\partial w}{\partial t} = F(x,t) + P(x,t) + EI\frac{\partial^4 w}{\partial x^4} \tag{6.1}$$

With w being the direction of movement, M being the mass of the beam, EI the stiffness, and $F(x)$ the externally applied force. In the considered scenario the externally applied force both is the tip-load, and the pressure. In order to solve this problem numerically, the Finite Difference Method (FD) is applied. The central difference (6.2) is applied to Equation 6.1.

$$\frac{\partial^4 w_n}{\partial x^4} = \frac{w_{n-2} - 2w_{n-1} + 6w_n - 2w_{n+1} + w_{n+2}}{\Delta x^4} \tag{6.2}$$

Applying the central difference FD method (6.2) to Equation 6.1, yields the following ordinary differential equation for the system:

$$\rho A l \mathbf{I_n}\ddot{w} = P_n + F_n + \frac{EI}{l^3}\mathbf{O}w_n \tag{6.3}$$

With $\mathbf{I_n}$ the identity matrix of size n, and $\mathbf{O}$ the finite difference coefficient matrix. Applying the clamped-free beam boundary conditions, see Equation 6.5 and 6.4, results in the following matrix for 5 elements. The finite difference coefficient matrix is confirmed using by the values found in Davoudi and Öchsner 2013.

$$\begin{aligned} w''' &= 0 \\ w'' &= 0 \end{aligned} \tag{6.4}$$

$$\begin{aligned} w' &= 0 \\ w'' &= 0 \end{aligned} \tag{6.5}$$

$$\mathbf{K} = \frac{EI}{l^3}\begin{bmatrix} 2 & -4 & 2 & 0 & 0 \\ -2 & 5 & -4 & 1 & 0 \\ 1 & -4 & 6 & -4 & 1 \\ 0 & 1 & -4 & 5 & -2 \\ 0 & 0 & 2 & -4 & 2 \end{bmatrix} \tag{6.6}$$

Evaluating the mass (**M**), and stiffness (**K**) matrix for the system leads to the following modal shapes for the beam-model, see Figure 6.3. In this figure a beam consisting of 40 elements is discretized. These modal shapes show agreement with expected results for a clamped-beam, signifying a correct mass and stiffness matrix.

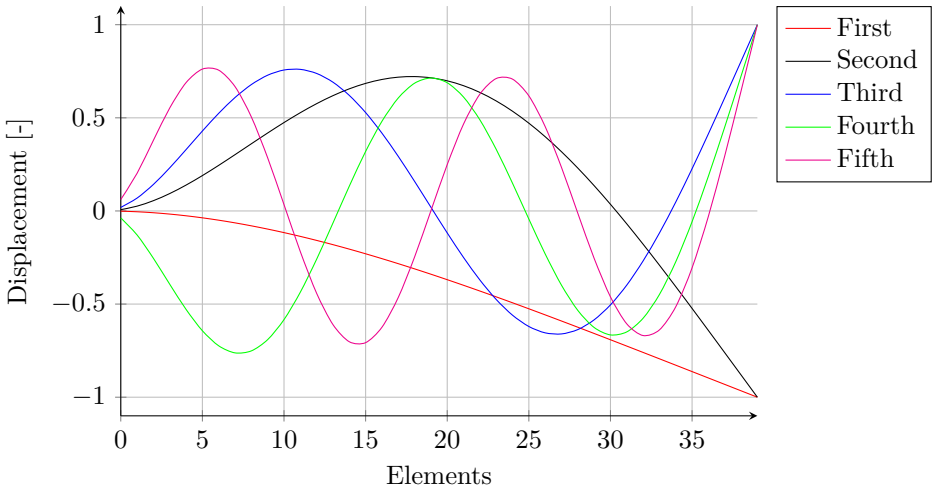


Figure 6.3: Normalized first five modal shapes of the discretized beam.

6.1.2 Pressure

The pressure at the ice-sheet is determined using the default SPH interpolation technique. The summation density (Equation 2.16) is used to determine the density at the nodal location of the beam. Using the Tait EOS, see Equation 2.23, the pressure at the nodal location is determined.

6.1.3 Force

The force exerted on the ice by the hull can be described by piece-wise linear model (Keijdener and A. V. Metrikine 2014), with two distinct phases: a crushing phase and an elastic phase . In the crushing phase the stiffness of the ice remains constant (σ_c), and the contact area ($A_{ct}(t)$) varies with time (t). In the sliding phase the contact area remains constant, however, the stiffness of the ice varies with time ($\sigma_{cr}(t)$). The crushing phase lasts until the contact force starts to decrease or, similarly, when the ice starts to retract from the structure, i.e. when the penetration speed ($V_\perp$) becomes negative.

$$F_{ct}(t) = \begin{cases} \sigma_c \cdot A_{ct}(t), & \text{when crushing} \\ \sigma_{cr}(t) \cdot A^*_{ct}, & \text{when sliding} \end{cases} \tag{6.7}$$

The sliding strength (σ_{cr}) should be determined by Lagrange Multipliers, however, following Keijdener, Hendrikse, and A. Metrikine 2018, this resulted in numerical errors. Therefore, a stiff spring contact is implemented instead. The change between modes is triggered by $V_\perp \leq 0$. This difference in penetration speed is calculated using a first-order central difference scheme.

The contact area of the ice (A_{ct}) can be determined by decomposing the crushed area, see Figure 6.1, this yields the following equation for the contact area:

$$\begin{aligned} L_{\perp} &= v_{ice}t \cdot \sin(\alpha) + w \cdot \cos(\alpha) \\ A_{ct}(t) &= bL_{\perp}\left(\tan(\alpha) + \tfrac{1}{\tan(\alpha)}\right) \end{aligned} \tag{6.8}$$

where α is the hull angle, and w the deflection of the beam.

6.2 Integration

The proposed model for the beam has a significant stiffness, which would result in large accelerations for small displacements, requiring prohibitively small time-steps to solve. Therefore, an alternative solver is required. An implicit solver requires time-consuming matrix inversion to find the solution, although, is generally more stable than explicit solvers. However, implicit solvers can not be used for the SPH fluid, since no matrices are known or computed for the particles, only explicit accelerations. Research is ongoing to try to create suitable implicit methods for SPH, such as half-implicit methods (Rakhsha et al. 2018), explicit-explicit methods (Lanzafame 2013), and derivation of stiffness matrices for SPH (Dinçer et al. 2019).

The Newmark-Beta solver is a commonly used solver in FEM (Newmark 1959), it is an adjustable solver using two parameters (β, and γ). This solver is a second-order ($\mathcal{O}(n^2)$) correct, when $\gamma = \frac{1}{2}$. This solver is preferred for the beam model since it is unconditionally stable, for $\beta \geq 1/4$ and $\gamma \geq 1/2$, and will thus converge for any time increment (Lindfield and Penny 2019). It consists of a predictor stage, see Equation 6.9.

$$\begin{aligned} \vec{\tilde{r}} &= \vec{r} + \vec{v}\delta t + \vec{a}\big(\tfrac{1}{2} - \beta\big)\delta t^2 \\ \vec{\tilde{v}} &= \vec{v} + \vec{a}\big(1 - \gamma\big)\delta t \end{aligned} \tag{6.9}$$

Using these predicted positions and velocities, the acceleration is computed implicitly using Equation 6.10. With -1 signifying the inverse matrix operation.

$$\vec{a} = \Big(\mathbf{M} + \gamma\mathbf{C}\delta t + \beta\mathbf{K}\delta t^2\Big)^{-1}\Big(\vec{F} - \mathbf{K}\cdot\vec{\tilde{r}} - \mathbf{C}\cdot\vec{\tilde{v}}\Big) \tag{6.10}$$

Finally, the resulting positions of the node can be computed using a correction scheme, see Equation 6.11.

$$\begin{aligned} \vec{r} &= \vec{\tilde{r}} + \vec{a}\beta\delta t^2 \\ \vec{v} &= \vec{\tilde{v}} + \vec{a}\gamma\delta t \end{aligned} \tag{6.11}$$

The time-steps used for this integration method is the same used for the integration of the SPH particles, with the general SPH time-step controls, see Section 3.4.

6.3 Verification

The model can be verified in two separate steps. Firstly, the beam/ice model is verified. Secondly, the pressure modeling is verified.

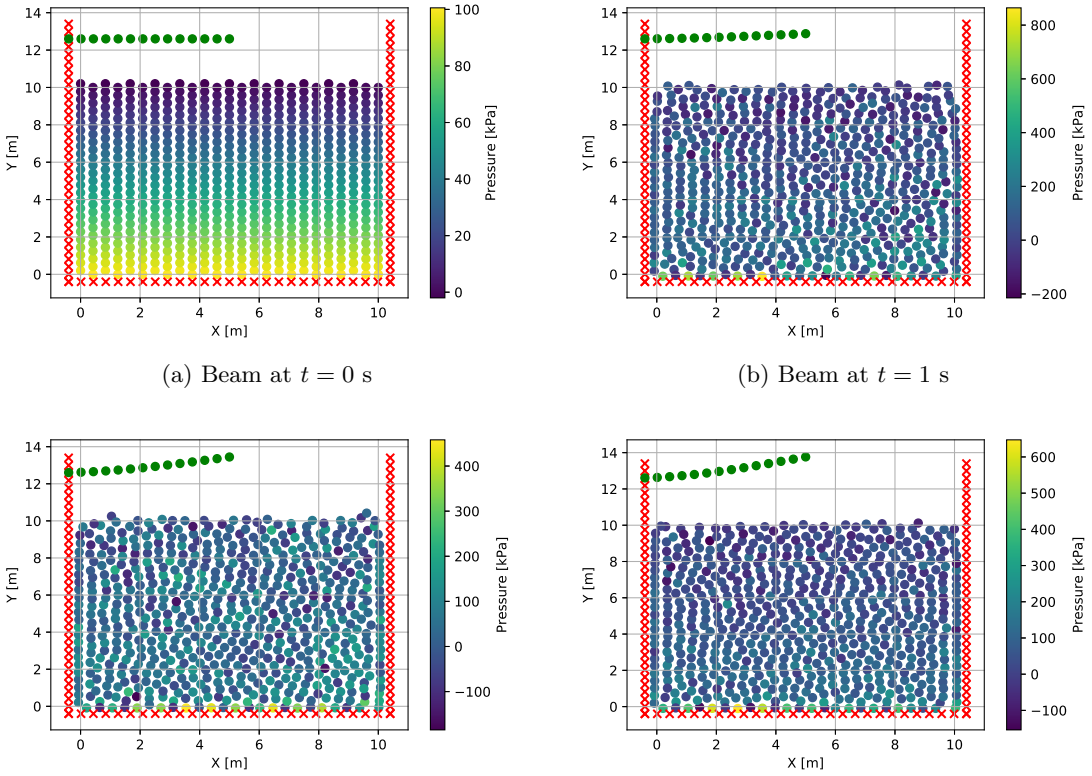
6.3.1 Beam Model

The beam-model, and Newmark-Beta integration method are verified by simulating a single metal beam. A Wendland kernel is used as the kernel function, see Section 2.1, because of its increased stability, see Section 3.2. The following parameters are used for the modeling of the beam, see Table 6.1. A β of 1/2, and a γ of 1/4 are used for the Newmark-Beta integration method for the beam. The force (F) is only applied at the last node (14).

Table 6.1: Beam parameters.

Parameter	**Value**
L	5 m
E	200 GPa
ρ	7800 kg/m^3
v	0.3
h	1/33.33 m
F	10 kN

An 10m×10m box of fluid is added to complete the SPH integration loop, and test the integration. This leads to the following time-series, see Figure 6.4. In this figure the borders are highlighted in red, and the beam is shown in green on the top of the figure.



(a) Beam at $t = 0$ s

(b) Beam at $t = 1$ s

(c) Beam at $t = 2$ s

(d) Beam at $t = 3$ s

Figure 6.4: Development of particles in beam scenario over time.

The time-series shows that the beam follows expected behaviour, oscillating between neutral and an upward displacement over-time. From the time-series the position of a specific node over-time can also be extracted. This position should follow the first modal shape since the force is applied constantly, without any frequency.

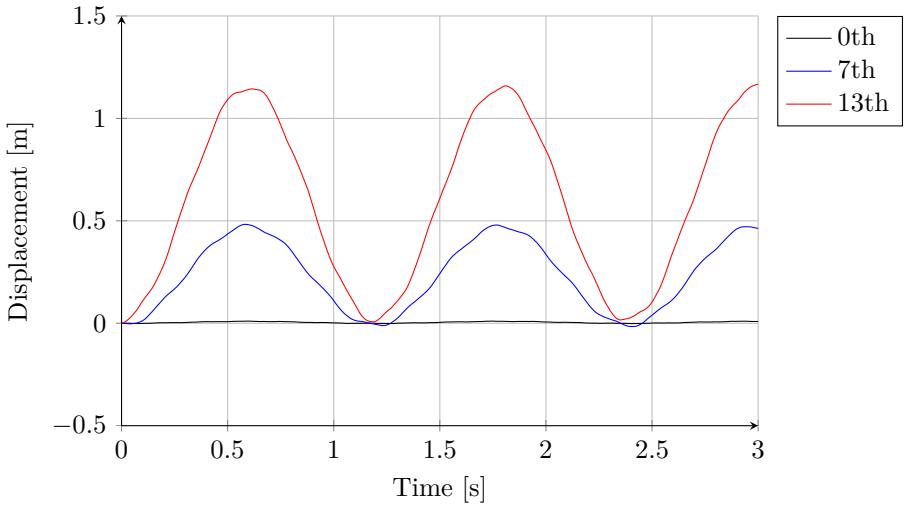


Figure 6.5: Displacement of specific nodes.

6.3.2 Pressure

The pressure at the surface of the model, and the interpolation procedure chosen is verified by way of a floating ice-sheet without stiffness. If the pressure is interpolated correctly, the ice-sheet, having a lower density than the water, should float on-top of the water with a mean pressure of approximately 9 kPa, given the values in Table 6.2. Furthermore, due to kernel inconsistencies near the boundaries, the pressure at the boundary location is expected to be lower than at the center of the beam. Again, the Newmark-Beta integration method, see Section 6.2, is used for the ice-sheet with parameters: $\gamma = 1/4$ and $\beta = 1/2$.

Table 6.2: Parameters for floating ice-sheet and water.

Parameter	**Value**
ρ_{ice}	916 kg/m^3
L_{ice}	20 m
B_{ice}	1 m
h_{ice}	1 m
ρ_{water}	1025 kg/m^3
H_{water}	10 m
L_{water}	20 m

This leads to the following pressures over-time, see Figure 6.6. The minimum pressure occurs at the edges and the maximum in the center of the beam. This is expected since the particles at the borders have fewer neighbours, thus, having a lower density and pressure, following Equation 2.16 and 2.23.

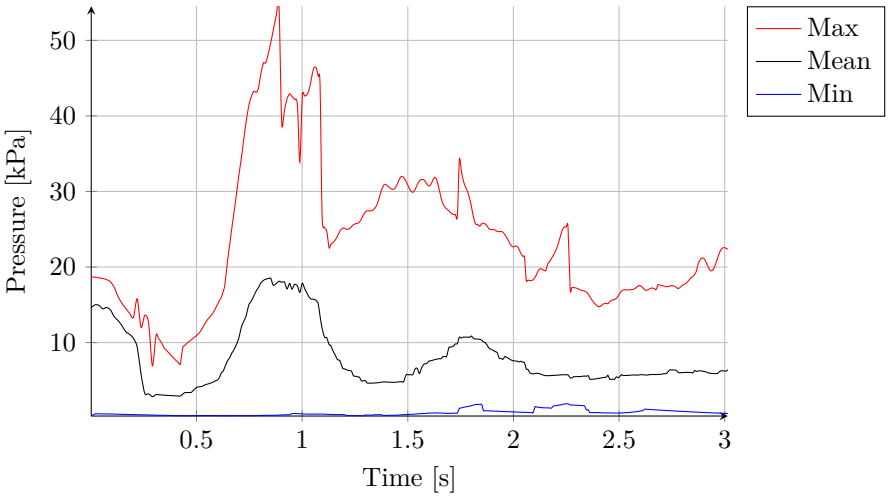


Figure 6.6: Maximum, mean, and minimum pressure over-time.

This pressure over-time follows the expected pattern, however, significant increases in pressure occur at the middle nodes.

6.4 Breaking Length

A critical value for the determination of the failure of the ice-sheet is the position at which failure occurs. This length, from the tip, is called the breaking-length of the ice. Failure of the ice-sheet occurs when the maximum stress (σ_{max}) exceeds the flexural strength of the ice (σ_{fl}).

$$\sigma_{max} = \max \left| \frac{Eh}{2(1-v^2)} \frac{\partial^2 w}{\partial^2 x} \right| \leq \sigma_{fl} \tag{6.12}$$

For the determination of the breaking-length and failure time the full-scale values provided for the mathematical model by Keijdener and A. V. Metrikine 2014 are utilised. The experimental values by Valanto 1992 could not be used in the SPH simulation since very low weight of the ice sheet, due to its small thickness (1/33.33 m), caused extreme instability. The full-scale parameters are shown in Table 6.3.

Table 6.3: Parameters for breaking-length determination.

Parameter	**Value**
ρ_{ice}	925 kg/m^3
L_{ice}	50 m
B_{ice}	1 m
h_{ice}	1 m
E_{ice}	5 GPa
v	0.3
σ_c	11 kPa
σ_{fl}	50 kPa

Settling is required due to Equation 2.18, in which the change in density is computed using the velocity difference ($\vec{v}_{ab}$). This velocity difference is initially zero at rest, therefore, no change in density and pressure occur until the fluid compresses and changes in velocity between particles develop. The settling time is determined by running the model without applying the contact-force and determining the time at which the fluid particles cross the initial height.

WCSPH suffers from spurious pressure fluctuations, in-order to minimize these the pressure at the surface is not used directly. Instead, a running average pressure is used, this should eliminate spurious pressure fluctuations.

Other numerical parameters such as the number of particle (n), see Section 6.6.1, damping-coefficient (ξ), width (L_{water}), and height (H_{water}) are determined using a grid search while optimizing for the correct failure time (t_{fail}) at an ice velocity (V_{ice}) of 0.2 m/s. These results are then compared to the results gathered by Keijdener, Hendrikse, and A. Metrikine 2018. This yields the following configuration for the beam, see Figure 6.7. This model consists of 9621 particles, of which 9048 fluid particles, 221 coupling particles, and 352 boundary particles.

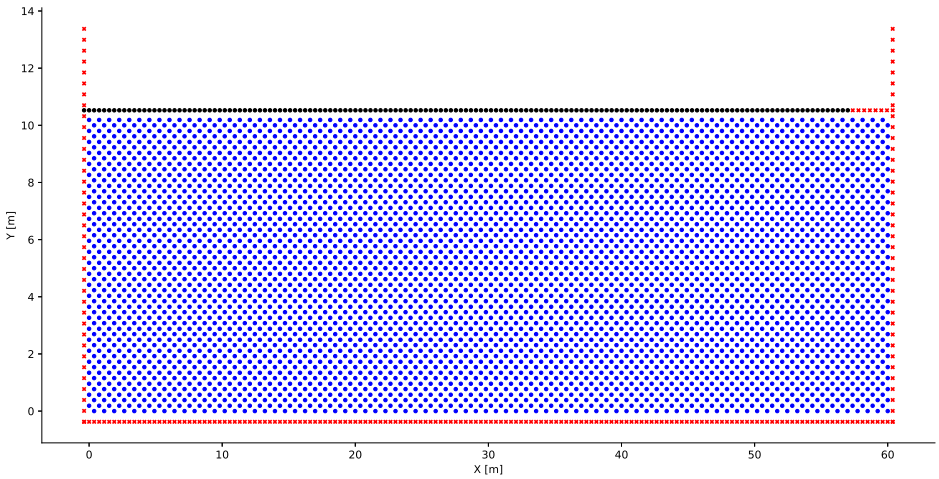


Figure 6.7: Configuration of proposed SPH model, boundaries are highlighted as $\times$, the fluid in blue, and the beam in black.

Using these values, the simulation is ran for different ice velocities ranging from 0.05 m/s to 0.5 m/s, the detailed velocities used and the numeric results from the simulations can be found in Appendix A. This results in the following failure time for different ice-velocities:

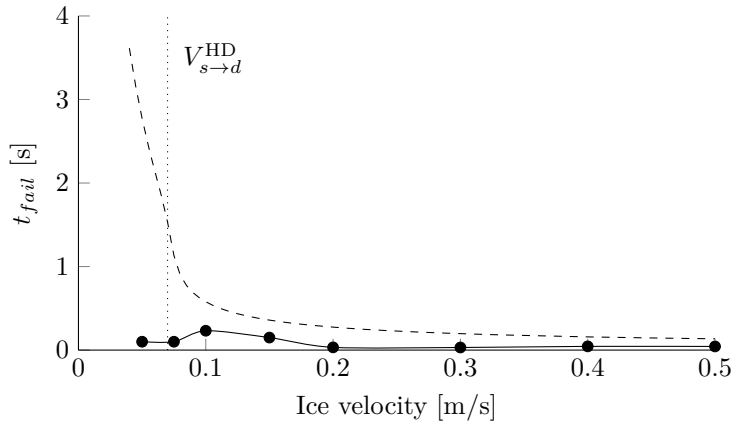


Figure 6.8: Failure time (t_{fail}) of the hydrodynamic model of Keijdener, Hendrikse, and A. Metrikine 2018 (dashed) compared to the proposed SPH model (solid). The dotted line is the transition velocity of static to dynamic failure.

As can be seen from the figure, the failure time of the proposed SPH model is significantly lower than the failure time as determined by Keijdener, Hendrikse, and A. Metrikine 2018. A possible explanation for this is that the coupled/ice particles behave like boundary particles, exerting a repulsive force on the fluid when the distance becomes smaller than the initial particle separation (r_0). Even though the pressure calculation is corrected for this, by measuring below the sheet of ice, the actual fluid level is slightly below this corrected level, because of the repulsive forces, this results in a lower pressure at the ice sheet than expected this lower pressure allows the ice to bend further and marginally decreases the failure times (t_{fail}). Furthermore, the settling time could also significantly impact the failure time, therefore, this is expected. Implementing an ICSPH model would resolve this issue since it requires less settling time since the density, and pressure are not based on the velocity difference between two particles.

The breaking-length, the position at which failure occurs, following criterium 6.12, measured from the tip of the ice-sheet, is also determined, see Figure 6.9. This information is collected in the same run as the failure times, see Figure 6.8.

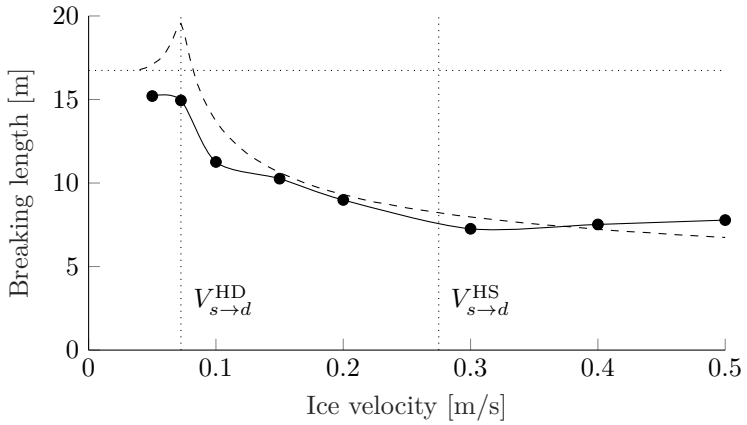


Figure 6.9: Breaking length ($L_{breaking}$) of the proposed SPH model (solid) compared to the model by Keijdener, Hendrikse, and A. Metrikine 2018 (dashed). The dotted line indicate the static breaking length and transition velocities.

The figure shows good agreement between the breaking length ($L_{breaking}$) and the model by Keijdener, Hendrikse, and A. Metrikine 2018. However, as the velocity increases, the breaking length for the SPH model slightly increases, this could be due to convergence issues or coarse particle spacing. The stress of the sheet is shown in Figure 6.10, this figure shows the stress along the beam due to the bending of the sheet.

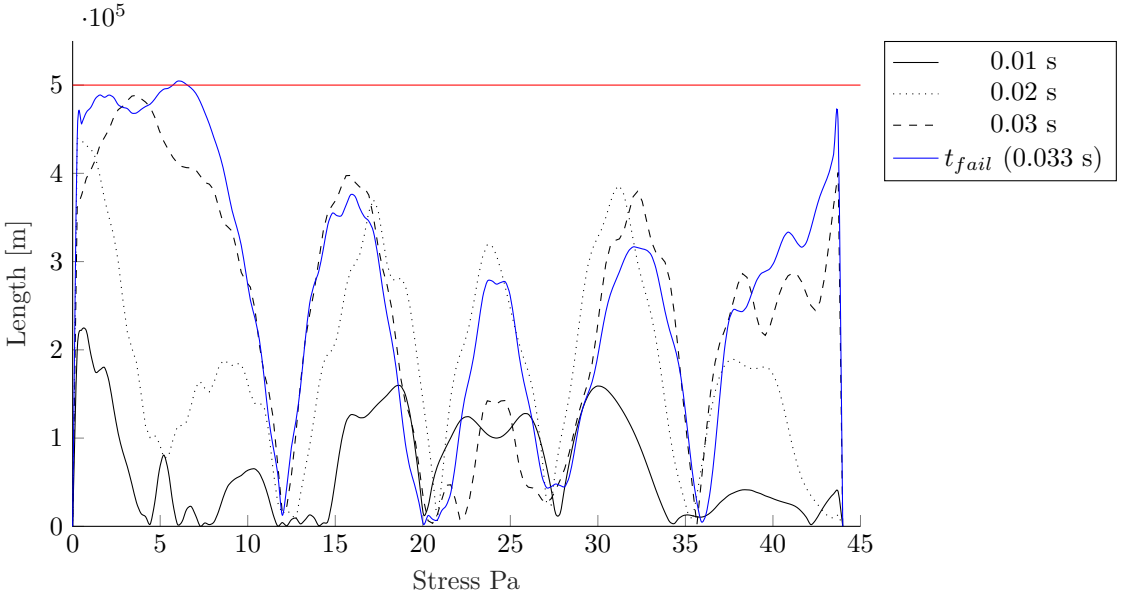


Figure 6.10: Stress at different locations in the ice-sheet during the simulation until t_{fail}. The failure strength of the ice is shown in red.

6.5 Conclusion

In conclusion, based on the results of the ice-breaking length simulation and failure time simulation, SPH shows promise in simulating ice-structure interactions. However, there is still a significant disagreement between the results, especially the failure time (t_{fail}). This could be explained by the settling time, the spurious pressure fluctuations present is WCSPH, the minimum boundary separation, or convergence issues. Despite these issues, it has been shown that it is possible to combine two integration methods within the SPH framework, that SPH can be used to model these ISI, and that dynamic effects are captured in the SPH model.

It is recommended to implement an ICSPH model, that model should result in much smoother behaviour of the ISI, since it does not suffer from spurious pressure fluctuations.

6.6 Discussion

6.6.1 Convergence

A convergence study was carried out for the failure time simulation, in this convergence study the number of particles was changed and compared to the results obtained by Keijdener, Hendrikse, and A. Metrikine 2018. All particles have an equidistant spacing, and thus only the number of particles in the ice sheet are shown in Figure 6.11. Furthermore, the number of particles is limited by the computational power available, increasing the number of particles requires more computational time and is unfeasible for the number of runs required.

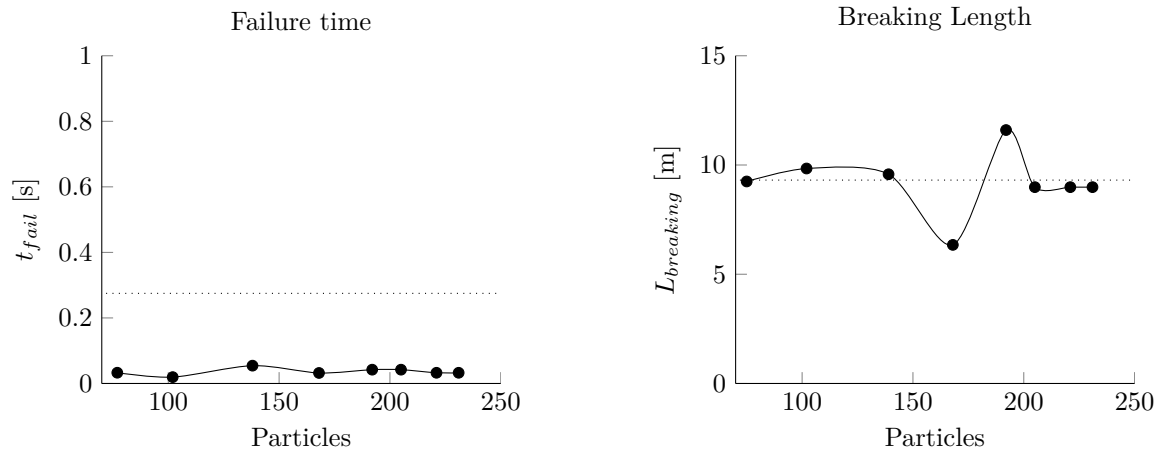


Figure 6.11: Results from the convergence study, all results are obtained at an ice velocity (V_{ice}) of 0.2 m/s. The dotted line indicates the results obtained by Keijdener, Hendrikse, and A. Metrikine 2018, the solid line the results obtained by the proposed SPH model.

The breaking length of the proposed SPH model initially drops as the number of particles increases, indicating a non-converged solution for the lower particle numbers. Above approximately 200 particles the breaking length stays relatively constant, indicating a converged solution. However, this is not extensively verified since particle numbers above 300 could not be run due to time constraints. The value for the failure time is significantly lower than the failure time found in Keijdener, Hendrikse, and A. Metrikine 2018, however, as described in Section 6.4, this can be explained by the settling time.

The number of particles is chosen to be 221, since this provides the optimal balance between computational time, and accuracy. Increasing the number of particles further would increase the accuracy but the computational time required would be prohibitive, since the computational time scales almost exponentially with the number of particles.

6.6.2 Force

The contact force, is the force exerted at the contact point between the hull and the ice-sheet. It is important to note that these figures do not match exactly since, the velocity of the ice is not equal in both scenarios. This is due to the velocity used by Keijdener, Hendrikse, and A. Metrikine 2018 being below the transition velocity, as noted in Section 6.4, the SPH model yields erroneous values

for velocities below the transition velocity. Therefore, the proposed SPH model is modeled at a velocity of 0.075 m/s, a velocity extremely close to the transition velocity. However, despite being close to the transition velocity, it represents an increase of 50% in velocity. Therefore, the forces are expected to be also approximately 50% higher.

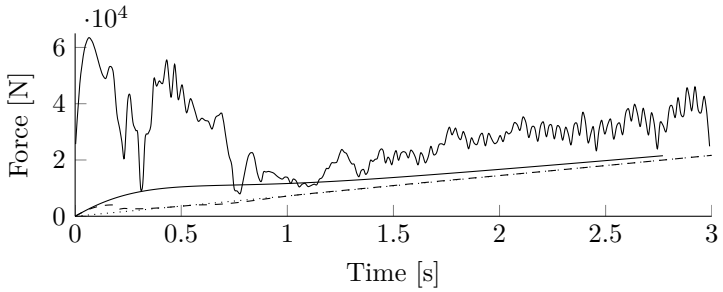


Figure 6.12: Contact force of the hydrodynamic (solid) and hydrostatic (dashed) by Keijdener, Hendrikse, and A. Metrikine 2018 at V_{ice} of 0.05 m/s. The contact force of the proposed SPH model is shown in blue, at V_{ice} of 0.075 m/s.

The figure shows a noisy contact force for the proposed model. However, the trend follows the same pattern as the mathematical model of Keijdener, Hendrikse, and A. Metrikine 2018. With the steady state closely matching, when accounting for the increase in velocity.

6.6.3 Penetration Velocity

Similarly to the contact force, see Section 6.6.2, the penetration velocity is evaluated for the proposed SPH model and compared to the

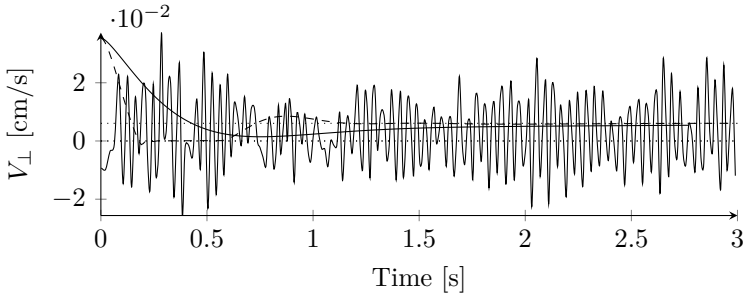


Figure 6.13: Contact force and $V_{\perp}$ for the hydrodynamic (solid) and hydrostatic (dashed) by Keijdener, Hendrikse, and A. Metrikine 2018. The dotted line show the static contact force, a $V_{\perp}(\infty)$ and a zero line.

Analogous to the contact force, the penetration velocity also shows a noisy pattern. Most likely due to the spurious pressure fluctuations present in the WCSPH implementation.

Chapter 7

Discussion

7.1 Limitations

The implementation as executed in this thesis, see Chapter 4, has several limitation that are outlined below. Firstly, the current implementation only covers the weakly compressible (WCSPH) method with few corrections. As stated in Chapter 3, this implementation suffers from some severe drawbacks namely spurious pressure changes, and time-stepping issues. Implementing a different method would improve this. Secondly, only limited validation, two cases, has been carried out for the current implementation. Furthermore, the validation scenarios are simple scenarios that do not model complex phenomenon. This limited validation constrains the practical use of the implementation for real world scenarios. While a number of problems, and solutions are listed in Chapter 3, this list is far from exhaustive. Many more novel solutions exist for the SPH field.

The ISI simulation carried out in Chapter 6, has several limitations. While the simulation works in general, a large amount of damping had to be added to prevent the simulation from breaking down. The simulation broke down because the water first sinks before expanding again, this behaviour occurs due to to continuity equation. The continuity equation is based on the difference in velocity of particles, when the simulation starts, this velocity difference is zero and thus no increase in pressure occurs until some compression occurs.

7.2 Relevance

Excellent implementation of SPH exist such as: DualSPHysics [1], Fluids v.3[2], and PySPH[3]. However, none of these allow for the easy to use switching as the implementation described in this thesis does. Furthermore, DualSPHysics and Fluidsv.3 are entirely written in C++, therefore, prohibitively complex for engineering students. PySPH allows for integration of custom equations and is written in python, however, it uses the Cython[4] extension to speed up the python program. This requires the user to write code in a rigid and non-pythonic manner, unsuitable for non computer science majors. Therefore, this implementation is necessary from a user perspective since it allows easy use for simple scenarios. Providing the user with quick iteration tools to increase productivity.

The literature study is relevant since it provides an aggregate of the current knowledge. Review literature exists, however, they tend to focus on the solutions; not asking why it is necessary in the first place.

The ISI simulation shows that SPH can be used to simulate ISI. The expands the toolbox of methods that can be used to simulate ISI. Furthermore, it demonstrates the possibility of combining SPH with solid mechanics.

7.3 Recommendations

WCSPH has been described in literature as unstable without significant corrections such as tensile, and kernel corrections (Violeau and Rogers 2016). Furthermore, offshore problems typically do not require the compressibility of the fluid to be modeled, therefore, it is recommended to implement ICSPH.

While the current performance of the program is sufficient, see Section 5.3, it is fairly slow. Several performance improvements can be made in order to improve this. Implementation of an improved near-neighbourhood search algorithm, such as fast-radix sort (Hoetzlein 2014) or a Verlet-Linked List (VLL) (Winkler, Rezavand, and Rauch 2018), would lead to significant performance gains.

Tackling the spurious pressure changes by implementing tensile corrections, kernel corrections, and/or other corrections described in Sections 3.3, 3.1, and 3.2. Implementing ICSPH would also improve simulation performance, as noted by Lee et al. 2008.

The improvement by ICSPH should also be used for the simulation of the ice-structure interactions. Currently, the pressure field is extremely noisy in the simulation which results in many mode switches between crushing and sliding ice. Furthermore, the number of particles should also be increased when possible. The current particle separation is in the order of 10^{-1} m, which results in difficulty assessing the breaking length. Increasing the number of particles would result in a finer grain.

[1] https://dual.sphysics.org/
[2] http://www.fluids3.com/
[3] https://github.com/pypr/pysph
[4] https://cython.org/

7.4 Conclusion

In conclusion, SPH is an extremely powerful particle method. It is completely mesh-free, therefore, uniquely suited for problems with large deformations, or discontinuities. Furthermore, it is an Lagrangian method, thus, solving these problems without numerical diffusion. However, the traditional weakly-compressible (WCSPH) implementation suffers from significant drawbacks such as spurious pressure fluctuations and clustering issues. With the right corrections, these problems can be overcome and the true power of SPH unlocked.

The implementation presented in this thesis, and available publicly on GitHub[5], shows a flexible implementation of the base WCSPH method. It provides the user with endless flexibility for their own couplings, methods, and equations; supercharging development of SPH related codes. Using this implementation researches can focus on what really maters, developing new and innovative applications for SPH.

The proposed SPH model solution for the determination of the ice-breaking length closely follows the trend determined by Keijdener, Hendrikse, and A. Metrikine 2018 in regards to the breaking length ($L_{breaking}$). However, it does not predict the failure time of the sheet correctly, the failure time follows the same downwards trend as the velocity increases, however, stays significantly below the failure time as determined by Keijdener, Hendrikse, and A. Metrikine 2018.

In the future I expect the use of SPH to grow in all industries, the off-shore industry included. It is a modeling technique with a lot of potential, further, developments in computing power, and algorithm improvements will dramatically reduce the computational time required.

[5] https://github.com/KoningJasper/Offshore-SPH

Bibliography

Antuono, M. et al. (2015). "Energy balance in the δ-SPH scheme". In: *Computer Methods in Applied Mechanics and Engineering*.

Balsara, D. S. (1995). "von Neumann Stability Analysis of Smoothed Particle Hydrodynamics Suggestions for Optimal Algorithms". In: *Journal of Computational Physics*.

Batchelor, G. K. (1967). "An Introduction to Fluid Dynamics". In:

Bathe, K. J. (2007). "Finite element method". In: *Wiley encyclopedia of computer science and engineering*, pp. 1–12.

— (2014). *Finite Element Procedures*. Prentice Hall.

Becker, M. and M. Teschner (2007). "Weakly compressible SPH for free surface flows". In: *Eurographics/ ACM SIGGRAPH Symposium on Computer Animation*.

Bird, K. J. et al. (2008). "Circum-Arctic Resource Appraisal: Estimates of Undiscovered Oil and Gas North of the Arctic Circle". In: *USGS Fact Sheet 2008*.

Bonet, J. and T.-S.L. Lok (1999). "Variational and momentum preserving aspects of Smooth Particle Hydrodynamics (SPH) formulations". In: *Computer Methods in Applied Mechanics and Engineering*.

Cherfils, J. M., G. Pinon, and E. Rivoalen (2012). "JOSEPHINE: A parallel SPH code for free-surface flows". In: *Computer Physics Communications*.

Colagrossi, A. and M. Landrini (2003). "Numerical simulation of interfacial flows by smoothed particle hydrodynamics". In:

Colagrossi, A., C. Lugni, and M. Brocchini (2010). "A study of violent sloshing wave impacts using an improved SPH method". In: *Journal of Hydraulic Research*.

Crespo, A. J. C. (2008). "Application of the Smoothed Particle Hydrodynamics model SPHysics to free-surface hydrodynamics". In:

Cummins, S. J. and M. J. Rudman (1999). "An SPH projection method". In: *Journal of Computational Physics*.

Davoudi, M. M. and A. Öchsner (2013). "On the accuracy offinite difference schemes for beam problems in the elastic range". In: *Materialwissenschaft und Werkstofftechnik*.

Dehnen, W. and H. Aly (2012). "Improving convergence in smoothed particle hydrodynamics simulations without pairing instability". In: *Monthly Notices of the Royal Astronomical Society*.

Dilts, G. A. (1999). "MOVING-LEAST-SQUARES-PARTICLE HYDRODYNAMICS—I. CONSISTENCY AND STABILITY". In: *International Journal for Numerical Methods in Engineering*.

Dinçer, A. et al. (2019). "Fully Coupled Smoothed Particle Hydrodynamics-Finite Element Method Approach for Fluid–Structure Interaction Problems With Large Deflections". In: *Journal of Fluids Engineering*.

Domínguez, J. M. et al. (2010). "Neighbour lists in smoothed particle hydrodynamics". In: *International Journal for Numerical Methods in Fluids*.

Frontiere, N., C. D. Raskin, and J. M. Owen (2017). "CRKSPH – A Conservative Reproducing Kernel Smoothed Particle Hydrodynamics Scheme". In: *Journal of Computational Physics*.
Gautier, Donald L. et al. (2009). "Assessment of Undiscovered Oil and Gas in the Arctic". In: *Science* 324.5931, pp. 1175–1179. ISSN: 0036-8075. DOI: 10.1126/science.1169467.
Gomez-Gesteira, M. et al. (2010). "Foreword: SPH for free-surface flows". In: *Journal of Hydraulic Research*.
Goswami, P. and C. Batty (2014). "Regional Time Stepping for SPH". In: *Eurographics*.
Gotoh, H. and A. Khayyer (2016). "Current achievements and future perspectives for projection-based particle methods with applications in ocean engineering". In: *Journal of Ocean Engineering and Marine Energy*.
Hoetzlein, R. K. (2014). "Fast Fixed-Radius Nearest Neighbors: Interactive Million-Particle Fluids". In: *GPU Technology Conference*.
Keijdener, C., H. Hendrikse, and A. Metrikine (2018). "The effect of hydrodynamics on the bending failure of level ice". In: *Cold Regions Science and Technology*.
The effect of ice velocity on the breaking length of level ice failing in downward bending (2014).
Khayyer, A., H. Gotoh, and S. Shao (2009). "Enhanced predictions of wave impact pressure by improved incompressible SPH methods". In: *Applied Ocean Research*.
Lanzafame, G. (2013). "Implicit integrations for SPH in semi-Lagrangian approach: Application to the accretion disc modeling in a microquasar". In: *Computer Physics Communications*.
Lattanzio, J. C. et al. (1985). "Interstellar cloud collisions". In: *Monthly Notices of the Royal Astronomical Society*.
Lee, E.-S. et al. (2008). "Comparisons of weakly compressible and truly incompressible algorithms for the SPH mesh free particle method". In: *Journal of Computational Physics*.
Lindfield, G. and J. Penny (2019). *Numerical Methods*. 4th ed. Academic Press.
Liu, M.B. and G. R. Liu (2010). "Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments". In: *Archives of Computational Methods in Engineering*.
Ma, Z. H. et al. (2018). "An overset mesh based multiphase flow solver for water entry problems". In: *Computers & Fluids*.
Martin, J. C. and W. J. Moyce (1952). "An Experimental Study of the Collapse of Liquid Columns on a Rigid Horizontal PLane". In: *Philosophical Transactions of the Royal Society of London*.
Moltini, D. and A. Colagrossi (2009). "A simple procedure to improve the pressure evaluation in hydrodynamic context using the SPH". In: *Computer Physics Communications*.
Monaghan, J. J. (1989). "On the Problem of Penetration in Particle Methods". In: *Journal of Computational Physics*.
— (1992). "Smoothed Particle Hydrodynamics". In: *Annual Review of Astronomy and Astrophysics*.
— (1994). "Simulating Free Surface Flows with SPH". In: *Journal of Computational Physics*.
— (2000). "SPH without a Tensile Instability". In: *Journal of Computational Physics*.
— (2005). "Smoothed particle hydrodynamics". In: *Reports on Progress in Physics*.
Monaghan, J. J. and R. A. Gingold (1977). "Smoothed particle hydrodynamics: theory and application to non-spherical stars". In: *Monthly Notices of the Royal Astronomical Society*.
Monaghan, J. J. and J. B. Kajtar (2009). "SPH particle boundary forces for arbitrary boundaries". In: *Computer Physics Communications*.
Morris, J. P., P. J. Fox, and Y. Zhu (1997). "Modeling Low Reynolds Number Incompressible Flows Using SPH". In: *Journal of Computational Physics*.
Newmark, N. M. (1959). "A Method of Computation for Structural Dynamics". In: *Journal of the Engineering Mechanics Division*.
Panizzo, A., T. Capone, and R. A. Dalrymple (2007). "Accuracy of kernel derivatives and numerical schemes in SPH". In:

Price, D. J. (2012). “Smoothed particle hydrodynamics and magnetohydrodynamics”. In: *Journal of Computational Physics*.

Rakhsha, M. et al. (2018). “Using a half-implicit integration scheme for the SPH-based solutionof fluid–solid interaction problems”. In: *Computer Methods in Applied Mechanics and Engineering*.

Shadloo, M. S. et al. (2012). “A robust weakly compressible SPH method and its comparison with an incompressible SPH”. In: *International Journal for Numerical Methods in Engineering*.

Skillen, A. et al. (2013). “Incompressible smoothed particle hydrodynamics (SPH) with reduced temporal noise and generalised Fickian smoothing applied to body-water slam and efficient wave-body interactions”. In: *Computer Methods in Applied Mechanics and Engineering*.

Stansby, P. K. (2018). “Smoothed Particle Hydrodynamics (SPH) for renewable energies offshore”. In: *Advances in Renewable Energies Offshore*.

Valanto, P. (1992). “THE ICEBREAKING PROBLEM IN TWO DIMENSIONS: EXPERIMENTS AND THEORY”. In: *Journal of Ship Research*.

Vaughan, G. L. et al. (2008). “Completeness, conservation and error in SPH for fluids”. In: *International Journal for Numerical Methods in Fluids*.

Violeau, D. and A. Leroy (2014). “On the maximum time step in weakly compressible SPH”. In: *Journal of Computational Physics*.

Violeau, D and B. D. Rogers (2016). “Smoothed Particle Hydrodynamics (SPH) for free-surface flows: past, present and future”. In: *Journal of Hydraulic Research*.

Wendland, H. (1995). “Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree”. In: *Advances in Computational Mathematics*.

Winkler, D., M. Rezavand, and W. Rauch (2018). “Neighbour lists for smoothed particle hydrodynamics on GPUs”. In: *Computer Physics Communications*.

Appendix A

Detailed Results

The detailed results for the proposed SPH model implementation are tabulated below. V_{ice} indicates the velocity of the ice, t_{fail} the time at which bending failure occurs, following criterium 6.12, and $L_{breaking}$ the position at which failure occurs. The parameters used for the simulation are shown in Table 6.3.

V_{ice} [m/s]	t_{fail} [s]	$L_{breaking}$ [m]
0.050	0.100112	15.205431
0.075	0.100112	14.945159
0.100	0.231688	11.260272
0.150	0.149806	10.260272
0.200	0.032648	8.986246
0.300	0.031092	7.260272
0.400	0.044531	7.520543
0.500	0.043623	7.780815