석사학위논문
Master’s Thesis

입자 부착 요소 보간을 이용한 SPH 밀도 보정의 3차원 확장

3D extension of the particle-attached element interpolation in Smoothed Particle Hydrodynamics

2020

김지인 (金智仁 Kim, Ji In)

한국과학기술원

Korea Advanced Institute of Science and Technology

석사학위논문

입자 부착 요소 보간을 이용한 SPH 밀도보정의 3차원 확장

2020

김지인

한국과학기술원

기계항공공학부 기계공학과

입자 부착 요소 보간을 이용한 SPH 밀도보정의 3차원 확장

김지인

위 논문은 한국과학기술원 석사학위논문으로
학위논문 심사위원회의 심사를 통과하였음

2020년 6월 12일

심사위원장 이필승 (인)

심사위원 한순흥 (인)

심사위원 오일권 (인)

3D extension of the particle-attached element interpolation in Smoothed Particle Hydrodynamics

Kim, Ji In

Advisor: Lee, Phill-Seung

A thesis submitted to the faculty of
Korea Advanced Institute of Science and Technology in
partial fulfillment of the requirements for the degree of
Master of Science in Mechanical Engineering

Daejeon, Korea
June 12, 2020

Approved by

____
Lee, Phill-Seung
Professor of Mechanical Engineering

The study was conducted in accordance with Code of Research Ethics[1].

[1] Declaration of Ethical Conduct in Research: I, as a graduate student of Korea Advanced Institute of Science and Technology, hereby declare that I have not committed any act that may damage the credibility of my research. This includes, but is not limited to, falsification, thesis written by someone else, distortion of research findings, and plagiarism. I confirm that my dissertation contains honest conclusions based on my own careful research under the guidance of my advisor.

MME

Kim, Ji In. 3D extension of the particle-attached element interpolation in Smoothed Particle Hydrodynamics. Department of Mechanical Engineering. 2020. 38+iv pages. Advisor: Lee, Phill-Seung. (Text in English)

Abstract

Random pressure and density oscillations in Smoothed Particle Hydrodynamics (SPH) often lead to degradation of numerical stability and accuracy of the entire simulation. Shepard filter is the most commonly used technique to alleviate such an issue, but the major problem with this method is that it is highly sensitive to the smoothing frequency and tends to over-smooth the density field when applied too frequently. To overcome this problem, a new density correction method was proposed by Seo for 2D problems, where density of each particle is reinitialized using particle-attached element and the corresponding shape functions. As an extension to his work, this paper presents a 3D implementation of the new density correction method. Similar to the 2D method which uses a 4-node square element, a 8-node cubic element with linear shape functions are employed for interpolation of the density field within the smoothing domain. Variable smoothing domain size is selected for each particle in each smoothing step, which enables the simulation to avoid over-smoothing and have converged solutions regardless of the smoothing frequency. The major portion of the calculation is carried out using a graphics processing unit (GPU) for faster simulation and three benchmark problems are solved to verify the feasibility of the method in 3D.

Keywords Smoothed Particle Hydrodynamics, Density correction, Fluid dynamics, 3D sloshing, 3D dam-break, GPU computing

Contents

List of Tables

List of Figures

Chapter 1. Introduction

Smoothed Particle Hydrodynamics (SPH) is a particle-based computational method that is designed to simulate the dynamic behavior of system. It discretizes a continuous system into a finite number of nodes called particles, and tracks individual particles by observing how their physical quantities change over time. In SPH, it is assumed that the behavior of one particle is influenced by its surrounding particles, and the interaction data obtained from the surrounding particles is used to calculate particles' physical quantities, including their velocity, position, density and energy.

SPH was first introduced by Lucy, Gingold and Monaghan in an effort to simulate astrophysical phenomena [1,2]. Since then, it has been widely used in various fields including solid mechanics [3] and fluid mechanics [4]. Compared to traditional mesh-based methods such as Finite Element Method (FEM) or Finite Volume Method (FVM), there is no connectivity between nodes in SPH, which enables it to handle more complex problems effectively without suffering from mesh distortion. Because of such characteristics, SPH has become a useful tool for solving various types of fluid problems involving free-surface flows, turbulent flows, multi-phase flows, and fluid-structure interactions [4-7].

There exist two main approaches employed for the analysis of incompressible flows using SPH, namely incompressible SPH (ISPH) and weakly compressible SPH (WCSPH). In the ISPH approach, the particles are assumed to be truly incompressible and the particle's pressure value is obtained by solving the pressure Poisson equation (PPE). In the WCSPH approach, the density is modeled to be slightly compressible, usually with the maximum density variation of 1%, and the pressure is evaluated by solving Tait's equation, which is a type of artificial state equation that relates density to pressure. Although both methods have their own advantages and disadvantages, WCSPH is more widely used among researchers because it is easy to implement and tends to have more uniform particle distribution which is highly desirable for the stability and accuracy of the method [8]. Moreover, WCSPH does not require any special treatments for free surface flows, whereas detection and imposition of boundary conditions on free surface particles is necessary in ISPH [9].

The phenomenon of random pressure oscillations is one of the well-known disadvantages of WCSPH, which originates from both the approximation error in evaluation of density and the use of stiff state equation. Since SPH uses particles for the approximation of continuous field functions, it is prone to error, especially when the particle distribution is non-uniform. Various attempts have been made to tackle this problem. For example, an addition of diffusive term, such as artificial viscosity or density diffusion into the differential equations of SPH was suggested by Monaghan [10] and Molteni [11]. In later work, Monaghan developed the XSPH method using velocity variant term for the correction of velocity [12]. Similarly, Shadloo attempted to correct the position of particles by introducing artificial particle displacement term [8]. All of these approaches were intended to produce more evenly distributed particles, and thereby reduce approximation error in density. A more direct approach to reduce the numerical oscillations was introduced by Colagrossi [6] which involves a periodic correction of density using Shepard's interpolation. This direct modification of density was proven to be most effective in reducing numerical oscillations and hence, often used as a default numerical scheme for WCSPH [13].

One limitation of the existing density correction method is that application frequency plays an important role in the performance of the method. The optimized value of frequency varies from problem to problem, and setting it too low or too high can greatly affect the overall results of the simulation. To avoid this, Seo [14] developed a new element-based interpolation scheme, where a virtual element attached to each particle is used as a smoothing domain of that particle. The smoothing strength of each element is controlled by adjusting the size of the element, and by doing so, the frequency dependent behavior of the method can be reduced.

So far, the validation of the above method was conducted for 2D cases only. In this study, an extension of the 2D method into 3D space is realized by introducing a 3D smoothing element. Various numerical treatments are implemented for stable simulation, and acceleration of computation is achieved by utilizing the computational power of a GPU. Finally, several numerical examples involving static and dynamic flows are solved to verify the feasibility of the method.

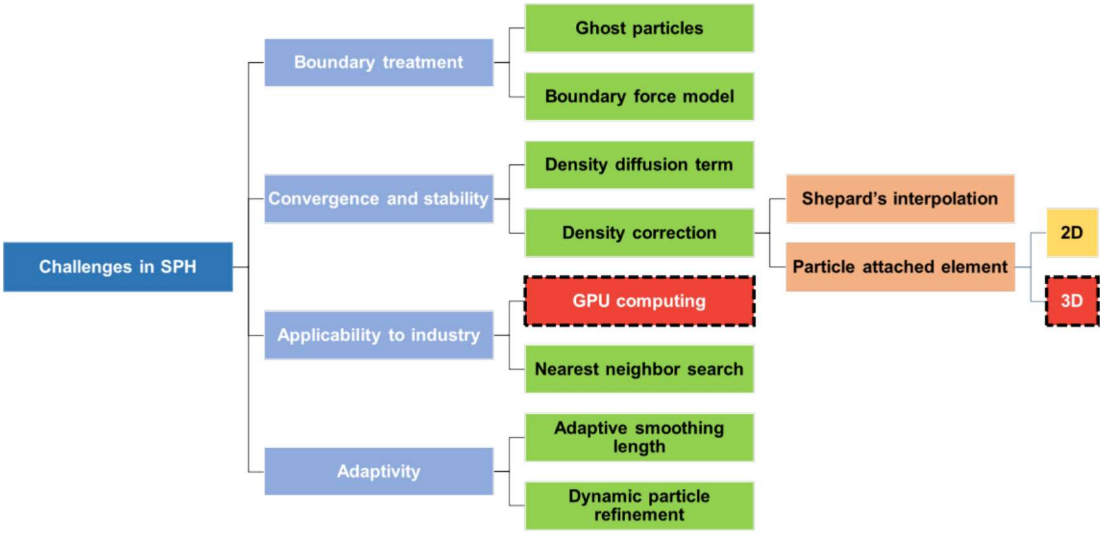

Figure 1. Technology tree of the SPH method

Chapter 2. Smoothed Particle Hydrodynamics

In this chapter, the formulation of SPH with some additional numerical treatments to improve the stability, accuracy and efficiency of the method is described in detail.

2.1. Formulation

2.1.1. Integral representation

The basic approximation scheme used in SPH consists of two key steps known as integral representation and particle approximation. In the integral representation, a field function of any point in a continuous body is represented as the weighted sum of the function value over its neighboring region Ω, as shown in equation (2.1). Substituting $\nabla \cdot f$ for f and rearranging the equation, a partial derivative of the field function can also be expressed as in equation (2.2).

$$f(\boldsymbol{x}) \doteq \int_{\Omega} f(\boldsymbol{x}')W(\boldsymbol{x}-\boldsymbol{x}',h)d\boldsymbol{x}' \qquad (2.1)$$

$$\nabla \cdot f(\boldsymbol{x}) \doteq -\int_{\Omega} f(\boldsymbol{x}') \cdot \nabla W(\boldsymbol{x}-\boldsymbol{x}',h)d\boldsymbol{x}' \qquad (2.2)$$

W in both equations (2.1) and (2.2) denotes a user-defined weight function, or so-called kernel function, which is used to represent how much influence each point has on its neighboring region. There exist a number of different kernel functions used in the SPH literature, and any function with the following properties can be a candidate for SPH kernel [15]:

$$\int W(\boldsymbol{x}-\boldsymbol{x}',h)d\boldsymbol{x}' = 1 \qquad \text{(normalization)}$$

$$W(\boldsymbol{x}-\boldsymbol{x}',h) = 0 \quad \textit{for} \quad |\boldsymbol{x}-\boldsymbol{x}'| > \kappa h \qquad \text{(compact support)}$$

$$W(\boldsymbol{x}-\boldsymbol{x}',h) \geq 0 \qquad \text{(positivity)}$$

$$\lim_{h \to 0} W(\boldsymbol{x}-\boldsymbol{x}',h) = \delta(\boldsymbol{x}-\boldsymbol{x}') \qquad \text{(Dirac delta)}$$

$$W(\boldsymbol{x}-\boldsymbol{x}',h)=W(-(\boldsymbol{x}-\boldsymbol{x}'),h) \qquad \text{(symmetric)}$$

The chosen kernel function should be monotonically decreasing and sufficiently smooth as shown in figure 2.

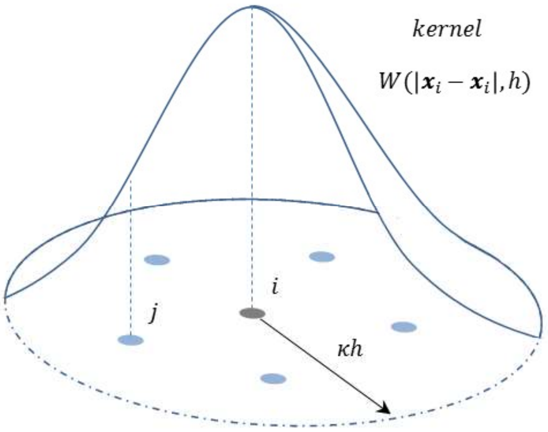

Figure 2. SPH kernel function

As can be seen, the value of kernel function approaches zero as a particle moves away from the center, and becomes zero when it is outside of the support domain. The radius of the support domain is expressed as κh, where h is the smoothing length and κ is some constant determining the size of the support domain. The choice of kernel function is critical in the SPH formulation as a poor choice of kernel may lead to instability of the entire simulation. Throughout this paper, a hyperbolic kernel function proposed by Yang and Liu [16] is used which has the form

$$W(r,h)=\frac{1}{3\pi h^2}\begin{cases} q^3-6q+6\,, & 0\le q<1 \\ (2-q)^3\,, & 1\le q<2 \\ 0\,, & 2\le q \end{cases} \qquad (2.3)$$

where $q=\dfrac{|\boldsymbol{r}|}{h}$.

2.1.2. Particle approximation

For the particle approximation, the continuous system is discretized into a finite number of particles and the physical quantity of an individual particle is determined by calculating the influence of its neighboring particles. The discretized, or the particle approximation form of equations (2.1) and (2.2) are shown below,

$$f(\boldsymbol{x}_i) \doteq \sum_{j=1}^{N} \frac{m_j}{\rho_j} f(\boldsymbol{x}_j) W(\boldsymbol{x}_i - \boldsymbol{x}_j, h) \tag{2.4}$$

$$\nabla \cdot f(\boldsymbol{x}_i) \doteq -\sum_{j=1}^{N} \frac{m_j}{\rho_j} f(\boldsymbol{x}_j) \cdot \nabla W(\boldsymbol{x}_i - \boldsymbol{x}_j, h) \tag{2.5}$$

where m and ρ are mass and density of each particle, and N represents the number of neighboring particles within the support domain of particle i.

2.1.3. Governing equation

The governing equations for the SPH method consist of three conservation laws, i.e. continuity, momentum and energy. Applying the particle approximation equation for general flow, these three laws can be expressed as equations (2.6), (2.7), and (2.8) respectively, where the effect of viscosity is neglected.

$$\frac{D\rho_i}{Dt} \cong \sum_{j=1}^{N} m_j v_{ij}^{\beta} \frac{\partial W_{ij}}{\partial x_i^{\alpha}} \tag{2.6}$$

$$\frac{Dv_i^{\alpha}}{Dt} \cong -\sum_{j=1}^{N} m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \frac{\partial W_{ij}}{\partial x_i^{\alpha}} \tag{2.7}$$

$$\frac{De_i}{Dt} \cong \frac{1}{2} \sum_{j=1}^{N} m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) v_{ij}^{\beta} \frac{\partial W_{ij}}{\partial x_i^{\alpha}} \tag{2.8}$$

In the above equations, p is pressure, W_{ij} is a kernel value of particle j with respect to i, and v_{ij} is a relative velocity of particle i with respect to j. Solving the above partial differential equations, the overall behavior of flow can be modeled over a period of time.

2.1.4. Pressure equation

In order to solve equations (2.7) and (2.8), the evaluation of pressure values for individual particles is essential. This can be done by finding the relationship between pressure and known state variables, which is often referred to as the pressure equation. There are several different pressure models one can choose from, such as pressure Poisson equation, Tait's equation, or ideal gas law, and depending on the choice of this pressure model, the SPH method can be classified as compressible, weakly compressible, or incompressible SPH [17]. In this work, we use a weakly compressible approach [4] with Tait's equation which has the form

$$P = \frac{c^2 \rho_o}{\gamma}\left(\left(\frac{\rho}{\rho_o}\right)^{\gamma} - 1\right) \tag{2.9}$$

where c is the speed of sound and ρ_o is the initial density. The speed of sound is set to be at least 10 times higher than the maximum velocity to ensure the weak compressibility of the fluid particles. Different values of γ is used for different materials, and as for water, $\gamma = 7$ is used.

2.2. Improved SPH

As SPH has become a powerful tool for analyzing and solving flow problems, a lot of research has been done on how to improve the stability and accuracy of SPH. Although the performance of most numerical techniques varies from problem to problem, there are a few techniques that are proven to be effective for a wide range of problems. In this section, some of the most commonly used numerical treatments to enhance the stability and accuracy of the SPH method are discussed in detail.

2.2.1. Boundary treatment

The main purpose of the boundary is to keep the fluid particles within the domain. To enforce this, various boundary models have been suggested by numerous researchers over the last few decades [4,18-19], where most of them incorporate the idea of introducing repulsive particles or ghost particles. However, the well-known particle deficiency problem arises when the particles approach the boundary. When the particles are near the boundary, part of their support domain is truncated by the boundary because no particles lie outside the boundary. The lack of these particles leads to particle deficiency problem, which often results in inaccurate and unstable solutions. To circumvent this problem, Shao proposed a coupled dynamic solid boundary model, which combines the idea of repulsive particle with ghost particle [20].

In this work, a modified coupled dynamic solid boundary with a single layer of wall particles and two layers of virtual particles are used [21]. Figure 3 shows the sketch of the boundary model. As fluid particles approach the boundary, the wall particles exert a repulsive force that pushes the fluid particles away from the boundary and hence, prevents them from penetrating the boundary. The repulsive force model is given by

$$\boldsymbol{F}_{ij} = 0.01\overline{c}_{ij}^{\,2} \cdot \chi \cdot f(\eta) \cdot \frac{\boldsymbol{x}_{ij}}{r_{ij}^{\,2}} \cdot m_i \tag{2.10}$$

$$\chi = \begin{cases} 1 - \dfrac{r_{ij}}{h}\,, & 0 < r_{ij} < h \\ 0\,, & \text{otherwise} \end{cases} \tag{2.11}$$

$$\eta = \frac{r_{ij}}{0.5h} \tag{2.12}$$

$$f(\eta) = \begin{cases} 2/3\,, & 0 < \eta < 2/3 \\ (2\eta - 1.5\eta^2)\,, & 2/3 < \eta < 1 \\ 0.5(2-\eta)^2\,, & 1 < \eta < 2 \\ 0\,, & \text{otherwise} \end{cases} \tag{2.13}$$

Moreover, both types of boundary particles are used in the interaction computations just like the regular fluid particles. Therefore, by introducing these boundary particles, the particle deficiency problem can be effectively eliminated which results in a more stable solution.

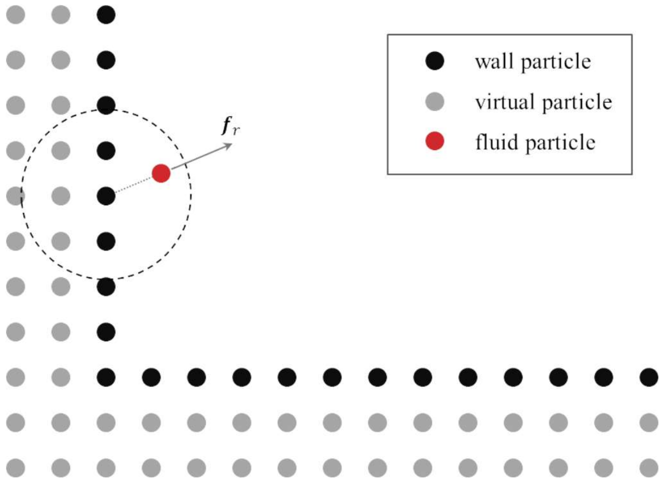

Figure 3. Sketch of coupled dynamic solid boundary

2.2.2. Artificial viscosity

Artificial viscosity is a form of numerical treatment that was first developed and applied by Monaghan and Gingold to handle shock wave problems [10]. It helps to remove the unphysical oscillations caused by shocks by dissipating energy in regions with sharp velocity variations [22]. There are a number of different artificial viscosity models present in the SPH literature, but the most widely used model is the Monaghan type artificial viscosity, which has the form

$$\Pi_{ij} = \begin{cases} \dfrac{-\alpha_\Pi \bar{c}_{ij}\phi_{ij} + \beta_\Pi \phi_{ij}{}^2}{\bar{\rho}_{ij}} , & \boldsymbol{v}_{ij} \cdot \boldsymbol{x}_{ij} < 0 \\ 0 , & \boldsymbol{v}_{ij} \cdot \boldsymbol{x}_{ij} \geq 0 \end{cases} \tag{2.14}$$

$$\phi_{ij} = \frac{h_{ij}\boldsymbol{v}_{ij} \cdot \boldsymbol{x}_{ij}}{\left|\boldsymbol{x}_{ij}\right|^2 + \varphi^2} \tag{2.15}$$

$$\bar{c}_{ij} = \frac{1}{2}(c_i + c_j) , \quad \bar{\rho}_{ij} = \frac{1}{2}(\rho_i + \rho_j) , \quad h_{ij} = \frac{1}{2}(h_i + h_j) \tag{2.16}$$

$$\boldsymbol{v}_{ij} = \boldsymbol{v}_i - \boldsymbol{v}_j , \quad \boldsymbol{x}_{ij} = \boldsymbol{x}_i - \boldsymbol{x}_j \tag{2.17}$$

where constants α_Π and β_Π are usually set around 1. Note that the value of Π_{ij} is non-zero only when two particles are approaching each other, i.e. when $\boldsymbol{v}_{ij} \cdot \boldsymbol{x}_{ij} < 0$, so the undesired particle clustering and interparticle penetration can be alleviated by using this method. Adding the artificial viscosity into the pressure term of equation (2.7) and (2.8), the final form of the momentum and energy equations can be rewritten as

$$\frac{Dv_i^{\alpha}}{Dt} \cong -\sum_{j=1}^{N} m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} + \Pi_{ij} \right) \frac{\partial W_{ij}}{\partial x_i^{\alpha}} \tag{2.18}$$

$$\frac{De_i}{Dt} \cong \frac{1}{2} \sum_{j=1}^{N} m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} + \Pi_{ij} \right) v_{ij}^{\beta} \frac{\partial W_{ij}}{\partial x_i^{\alpha}} \tag{2.19}$$

2.2.3. Shepard's filter

Shepard's interpolation is the most commonly used density correction method in the SPH literature. It is a well-known multivariate interpolation technique, which incorporates the idea of inverse distance weighting with SPH kernel as its weight function [6]. Using this method, the density of each particle is re-calculated as follows

$$\rho_i = \frac{\sum_{j=1}^{N} m_j W_{ij}}{\sum_{j=1}^{N} \frac{m_j}{\rho_j} W_{ij}} \tag{2.20}$$

While the numerical oscillations present in SPH can be effectively alleviated by applying this method periodically, a careful study of the application frequency is needed as it can affect the overall result of the method. Oftentimes, applying it too frequently results in over-smoothed distribution of pressure and density, especially in long time simulations.

2.3. Computational procedures

The five main computational steps in SPH are as follows: 1) Initialization of particle and grid cell data;

2) Assigning particles to the cells they belong to and store it in a cell linked-list; 3) Search for neighboring particles; 4) Calculations of time derivatives of particles' physical quantities; 5) Time integration of values found in step 4. Once the initialization is done, steps 2 through 5 are carried out for every time step in an iterative manner for the duration of simulation. The flowchart of the entire SPH simulation is displayed in figure 4.

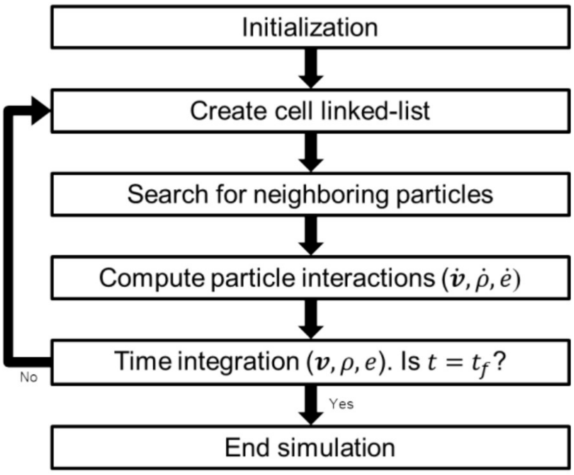
Initialization
Create cell linked-list
Search for neighboring particles
Compute particle interactions ($\dot{\boldsymbol{v}}, \dot{\rho}, \dot{e}$)
Time integration ($\boldsymbol{v}, \rho, e$). Is $t = t_f$?
No
Yes
End simulation

Figure 4. Flowchart of the SPH simulation

It should be noted that step 2 is not a fundamental step and can be eliminated or modified if a different search method is employed.

2.3.1. Neighbor search

Searching for neighboring particles is a time consuming process. The brute force neighbor search requires distance computation between every possible particle pair, which has a time complexity of $O(N^2)$. However, since the kernel function used in SPH enforces a compact support condition, only the particles in close vicinity are used in the interaction computations. Therefore, with the choice of appropriate neighbor search algorithm, this order can be reduced down to $N \log N$ or even N. There exist a number of different neighbor search algorithms, such as grid-based, tree-based, and hash-based, and in this work, the grid-based method is implemented, as it is fast, easy to implement and also well-suited for parallelization.

The grid-based neighbor search can be broken down into two parts, i.e. creating a cell linked-list and searching for neighboring particles. Cell linked-list is a memory efficient data structure which is composed of two separate arrays called head of the list, or simply head, and linked-list. In the head array, the highest particle number in each cell is stored, and this number is used as a pointer to the linked-list. Similarly, elements in linked-list represent both the particle number and the pointer to the next particle. Therefore, once the cell number is known, all particles belonging to that cell can be tracked sequentially using the cell linked-list as shown in figure 5.

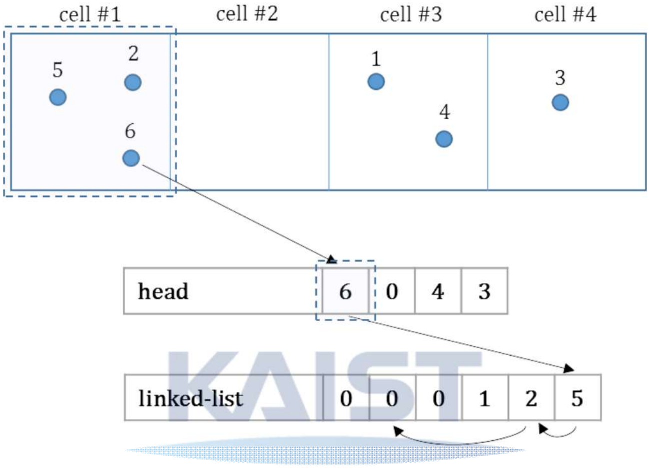

Figure 5. Structure of cell linked-list

In order to create a cell linked-list, a virtual cell grid with a cell dimension of κh is constructed. The cell dimension is selected to be κh to ensure that all possible neighboring particles in the support domain of particle i lie within the adjacent cells of i. Each cell is then assigned a cell number, and in every iteration, particles are sorted into cells according to their positions, and their information is stored in a cell linked-list. During the search step, the cell number of target particle i is calculated based on its position, and the distance is compared for all particles in the adjacent cells of i. The number of adjacent cells is 9 in 2D and 27 in 3D including the self cell. An illustration of the grid-based neighbor search is shown in figure 6.

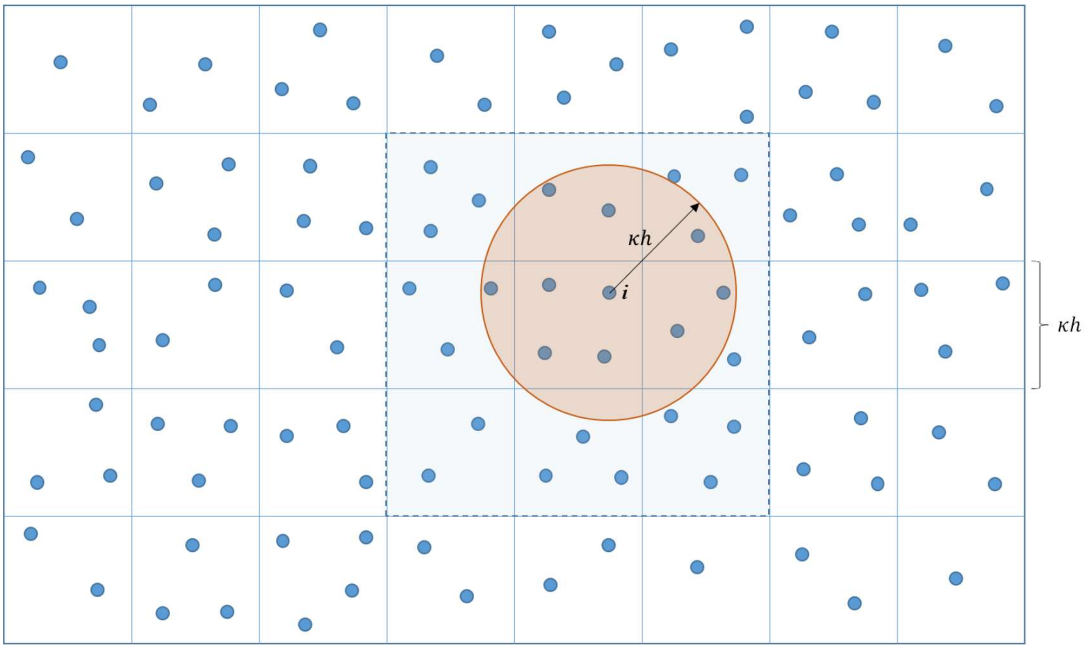

Figure 6. Grid-based neighbor search

2.3.2. Time integration

For the time integration of discrete ordinary differential equations derived in section 2.1, a two-stage predictor-corrector scheme is implemented [12,21]. In the prediction step, the intermediate values of particle's physical quantities at the mid-point of t and $t+\Delta t$ are evaluated as follows

$$\begin{aligned} \boldsymbol{x}_i^{n+0.5} &= \boldsymbol{x}_i^n + \frac{\Delta t}{2}\boldsymbol{v}_i^n \\ \boldsymbol{v}_i^{n+0.5} &= \boldsymbol{v}_i^n + \frac{\Delta t}{2}\frac{d\boldsymbol{v}_i^n}{dt} \\ \rho_i^{n+0.5} &= \rho_i^n + \frac{\Delta t}{2}\frac{d\rho_i^n}{dt} \\ e_i^{n+0.5} &= e_i^n + \frac{\Delta t}{2}\frac{de_i^n}{dt} \end{aligned} \tag{2.21}$$

In which the size of Δt is determined by applying the Courant-Friedrichs-Lewy (CFL) condition [23] as

$$\Delta t \le \min\left(\frac{h}{c},\ 0.25\sqrt{\frac{h}{a}},\ 0.125\frac{h^2}{\nu}\right) \tag{2.22}$$

where c is the speed of sound, a is minimum acceleration of particle, and ν is kinematic viscosity of fluid.

The values obtained in equation (2.21) are then use in the correction step to calculate the corrected values of particle's physical quantities as

$$
\begin{aligned}
\boldsymbol{x}_i^{n+0.5} &= \boldsymbol{x}_i^{n} + \frac{\Delta t}{2}\boldsymbol{v}_i^{n+0.5} \\
\boldsymbol{v}_i^{n+0.5} &= \boldsymbol{v}_i^{n} + \frac{\Delta t}{2}\frac{d\boldsymbol{v}_i^{n+0.5}}{dt} \\
\rho_i^{n+0.5} &= \rho_i^{n} + \frac{\Delta t}{2}\frac{d\rho_i^{n+0.5}}{dt} \\
e_i^{n+0.5} &= e_i^{n} + \frac{\Delta t}{2}\frac{de_i^{n+0.5}}{dt}
\end{aligned}
\qquad (2.23)
$$

Finally, the physical quantities of particles at the new time step is calculated as follows.

$$
\begin{aligned}
\boldsymbol{x}_i^{n+1} &= 2\boldsymbol{x}_i^{n+0.5} - \boldsymbol{x}_i^{n} \\
\boldsymbol{v}_i^{n+1} &= 2\boldsymbol{v}_i^{n+0.5} - \boldsymbol{v}_i^{n} \\
\rho_i^{n+1} &= 2\rho_i^{n+0.5} - \rho_i^{n} \\
e_i^{n+1} &= 2e_i^{n+0.5} - e_i^{n}
\end{aligned}
\qquad (2.24)
$$

2.3.3. GPU implementation

Just like any other numerical methods, SPH possesses some inherent problems which need to be resolved. One major challenge of the SPH method is its large computational cost. Various CPU-based approaches have been made to reduce computation time of SPH including a neighbor search algorithm, but further improvement is still demanded. This leads the researchers to work on a parallel implementation of SPH, especially by taking advantage of the highly parallel nature of GPU. Unlike other CPU-based parallel programming, overhead for creating threads is very little on GPU whereas that for calling a function on GPU can be relatively large. Hence, computational power of GPU can only be fully exploited when a sufficient number of threads are used. Typically, one thread handles calculations for one element in an output array, provided that the result of one thread is independent of the result of any other threads. Since the calculated results of one particle do not affect the results of any other particles during the same iteration step, it makes SPH an ideal candidate for GPU computing.

Since creating a cell linked-list is a serial process, only part of the computations steps is performed on GPU. For comparison, the computation time for each step is measured and recorded in table 1. As presented, the search and interaction calculation steps make up nearly 99 % of the total computation time, so by parallelizing these computationally intensive steps, a significant time reduction can be achieved.

Computation step	Time (s)	Percentage (%)
Cell linked-list	0.017	0.61
Neighbor search	1.972	70.48
Interaction calculations	0.793	28.34
Time integration	0.016	0.57
Total	2.798	100

Table 1. Computational time per time step on CPU

The flow chart of the GPU-accelerated SPH program is shown in figure 7, where the computation steps indicated by the red box are performed on GPU. The transfer of data occurs between CPU and GPU as shown in the figure.

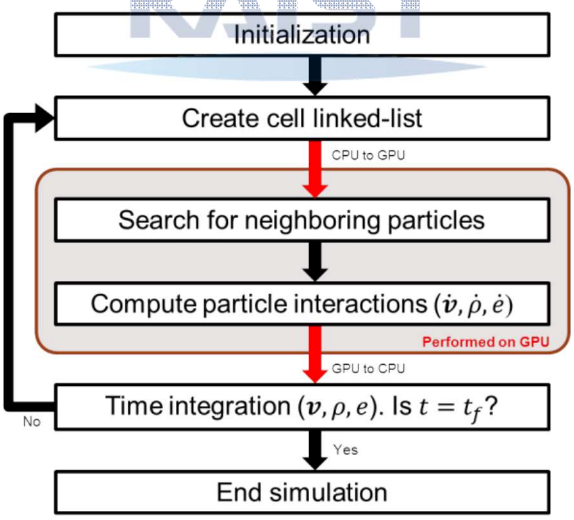
Initialization
Create cell linked-list
CPU to GPU
Search for neighboring particles
Compute particle interactions ($\dot{\boldsymbol{v}}, \dot{\rho}, \dot{e}$)
Performed on GPU
GPU to CPU
No
Time integration ($\boldsymbol{v}, \rho, e$). Is $t = t_f$?
Yes
End simulation

Figure 7. Flowchart of the SPH simulation with GPU implementation

Once the cell linked-list is created on the CPU, it is immediately transferred to the GPU with other particle data. From there, steps 3 and 4 are carried out on the GPU. Using the GPU, these steps can run simultaneously for each particle, which can lead to a significant time reduction in the overall simulation. Table 2 shows the time comparison chart between serial and parallel run of the same code.

Number of particles, N	CPU time (s)	GPU time (s)	Speedup
10,083	0.0597	0.0216	2.77
30,312	0.1635	0.0413	3.96
51,686	0.3217	0.0617	5.21
100,770	0.5951	0.1089	5.47
502,664	2.9611	0.5011	5.91
1,007,172	5.7927	0.9381	6.17

Table 2. Comparison of running time per time step with speedup ratio, CPU vs. GPU

In figures 8 and 9, the plot of the run time and the speedup ratio, which is obtained by dividing the computation time of the CPU code by that of the GPU code, are shown for the increasing number of particles. As shown in figure 9, the performance of the GPU code shows better results when the number of threads used increases, which in this case, is expressed as the number of particles. Note that the speedup factor is not ideally proportional to the number of particles, as a major GPU bottleneck occurs when transferring data between CPU and GPU. In the end, the speedup ratio of up to 6.17 is achieved when 1 million particles are used.

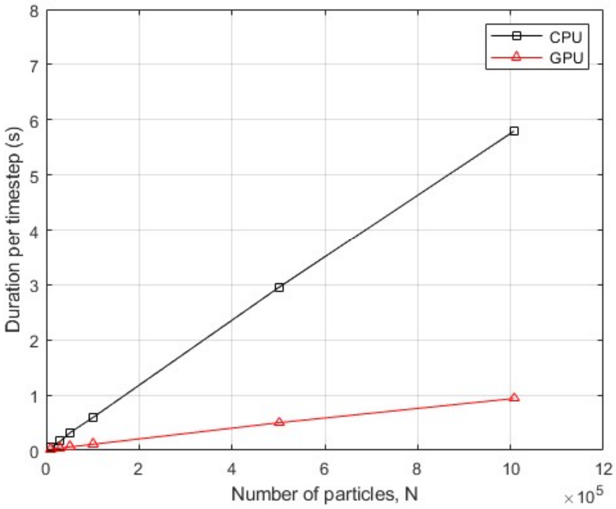

Figure 8. Comparison of running time, CPU vs. GPU

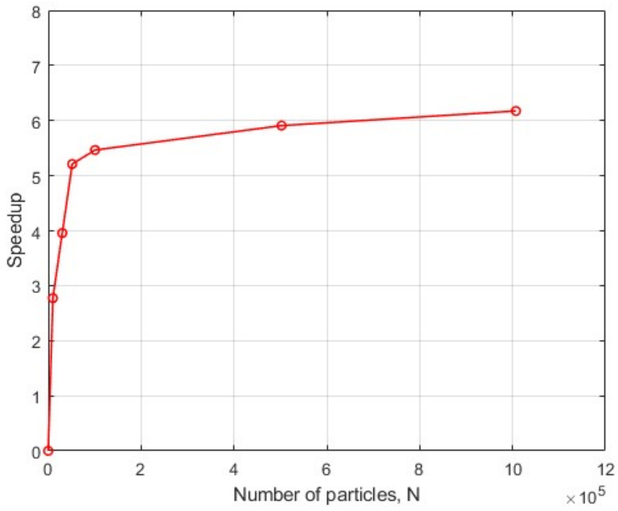

Figure 9. Speedup ratio with increasing number of particles

Chapter 3. 3D density correction using particle-attached element

The new density correction method to overcome the aforementioned drawbacks of Shepard's interpolation was proposed by Seo [14] for 2D cases, but its validity in 3D simulation is yet to be confirmed. In this section, a detailed description of the 3D implementation of the 2D method is provided with illustration for ease of understanding.

3.1. Particle-attached element

The idea of the new density correction method is based on the nodal interpolation method used in FEM. In FEM, a field variable at any point in an element is interpolated using the values at the element nodes. Similar to this concept, a virtual FEM-like element is created for SPH particles, where density is interpolated back and forth between element nodes and particles. This virtual background element is generated for each particle at each correction step, and hence termed particle-attached element.

In 2D approach, a 4-node square element with 2D linear shape functions are used to distribute the weighted density of particles to each node. Due to the extra dimension in 3D space, a 8-node cubic element with 3D linear shape functions are adopted, where the local coordinate system for each element is defined with particle i at the origin as shown in figure 10.

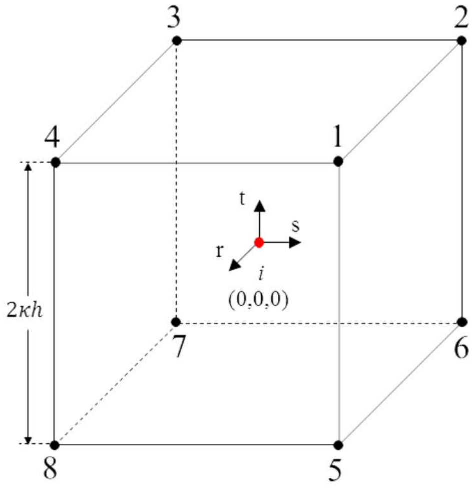

Figure 10. Local coordinate system of a background element

The densities of individual neighboring particles in the support domain of i are distributed among nodes using the following equation

$$D_n = \frac{\sum_{j=1}^{N} H_n(r_j, s_j, t_j)\, m_j \rho_j}{\sum_{j=1}^{N} H_n(r_j, s_j, t_j)\, m_j} \tag{3.1}$$

where N is the number of all particles in the support domain, and D_n is the nodal density of a background element. H_n in equation (3.1) denotes a set of nodal shape functions or interpolation functions, which, in this case, has a linear form as

$$\begin{aligned}
H_1(r,s,t) &= (\kappa h + r)(\kappa h + s)(\kappa h + t)/(2\kappa h)^3 \\
H_2(r,s,t) &= (\kappa h - r)(\kappa h + s)(\kappa h + t)/(2\kappa h)^3 \\
H_3(r,s,t) &= (\kappa h - r)(\kappa h - s)(\kappa h + t)/(2\kappa h)^3 \\
H_4(r,s,t) &= (\kappa h + r)(\kappa h - s)(\kappa h + t)/(2\kappa h)^3 \\
H_5(r,s,t) &= (\kappa h + r)(\kappa h + s)(\kappa h - t)/(2\kappa h)^3 \\
H_6(r,s,t) &= (\kappa h - r)(\kappa h + s)(\kappa h - t)/(2\kappa h)^3 \\
H_7(r,s,t) &= (\kappa h - r)(\kappa h - s)(\kappa h - t)/(2\kappa h)^3 \\
H_8(r,s,t) &= (\kappa h + r)(\kappa h - s)(\kappa h - t)/(2\kappa h)^3
\end{aligned} \tag{3.2}$$

The final product of the corrected density is calculated by interpolating the nodal densities at the origin as

$$\rho_i = \sum_{n=1}^{N_{node}} H_n(r_i, s_i, t_i)\, D_n = \sum_{n=1}^{N_{node}} H_n(0,0,0)\, D_n \tag{3.3}$$

Table 3 shows the comparison chart between 2D and 3D method, and an illustration of a background element in 2D and 3D spaces are shown in figure 11.

	2D method	3D method
Element type	4-node square element	8-node cubic element
Interpolation function	2D linear shape functions	3D linear shape functions
Platform	CPU	GPU

Table 3. Comparison chart between 2D and 3D method

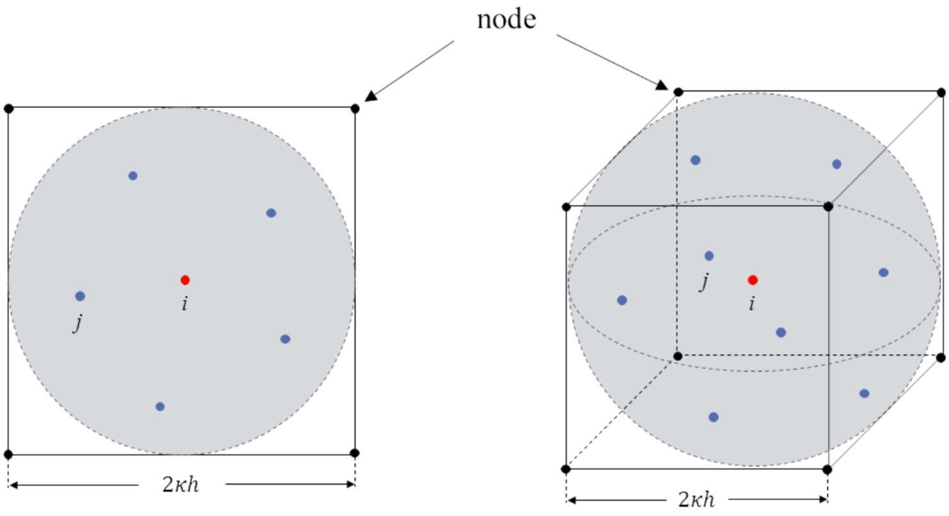

Figure 11. Particle-attached element in 2D (left) and 3D (right)

3.2. Correction steps

The density correction procedures can be broken down into three steps. In the first step, only the interactions between fluid particles are considered. The nodal density values of the smoothing element are obtained using only the information of the fluid particles, and any influence from the boundary particles is neglected during this step. The nodal density values for the fluid particle is calculated using

$$D_n^f = \frac{\sum_{j=1}^{N^f} H_n(r_j, s_j, t_j) m_j^f \rho_j^f}{\sum_{j=1}^{N^f} H_n(r_j, s_j, t_j) m_j^f} \tag{3.4}$$

where superscript f indicates the fluid particles and subscript n indicates the node number of a smoothing element. N^f here denotes a total number of fluid particles in the support domain of target fluid particle i. The final value of the corrected density of particle i is obtained by solving

$$\rho_i^f = \sum_{n=1}^{N_{node}} H_n(r_i, s_i, t_i) D_n^f = \sum_{n=1}^{N_{node}} H_n(0,0,0) D_n^f \tag{3.5}$$

Once the correction of all fluid particles is completed, the correction of the wall particles is carried out in two separate steps. First, the densities of wall particles are updated using the corrected fluid density acquired in the previous step as follows

$$\overline{D}_n^{\,w} = \frac{\sum_{j=1}^{\overline{N}^f} H_n(r_j,s_j,t_j)\, m_j^f \rho_j^f}{\sum_{j=1}^{\overline{N}^f} H_n(r_j,s_j,t_j)\, m_j^f} \tag{3.6}$$

$$\overline{\rho}_i^{\,w} = \sum_{n=1}^{N_{node}} H_n(r_i,s_i,t_i)\,\overline{D}_n^{\,w} = \sum_{n=1}^{N_{node}} H_n(0,0,0)\,\overline{D}_n^{\,w} \tag{3.7}$$

where $\overline{\rho}_i^{\,w}$ and $\overline{D}_i^{\,w}$ represent intermediate density and nodal density values of target wall particle i, and $\overline{N}^f$ is the number of fluid particles in the support domain of i. In the final step, the densities of wall particles are corrected once more, and this time, only the interactions between wall particles are considered. The final value of corrected densities of wall particles are calculated using

$$D_n^{\,w} = \frac{\sum_{j=1}^{N^w} H_n(r_j,s_j,t_j)\, m_j^f \overline{\rho}_j^{\,w}}{\sum_{j=1}^{N^w} H_n(r_j,s_j,t_j)\, m_j^f} \tag{3.8}$$

$$\rho_i^{\,w} = \sum_{n=1}^{N_{node}} H_n(r_i,s_i,t_i)\, D_n^{\,w} = \sum_{n=1}^{N_{node}} H_n(0,0,0)\, D_n^{\,w} \tag{3.9}$$

The density correction procedures are illustrated in figure 12, where the shaded region represents the area of interest for each correction step.

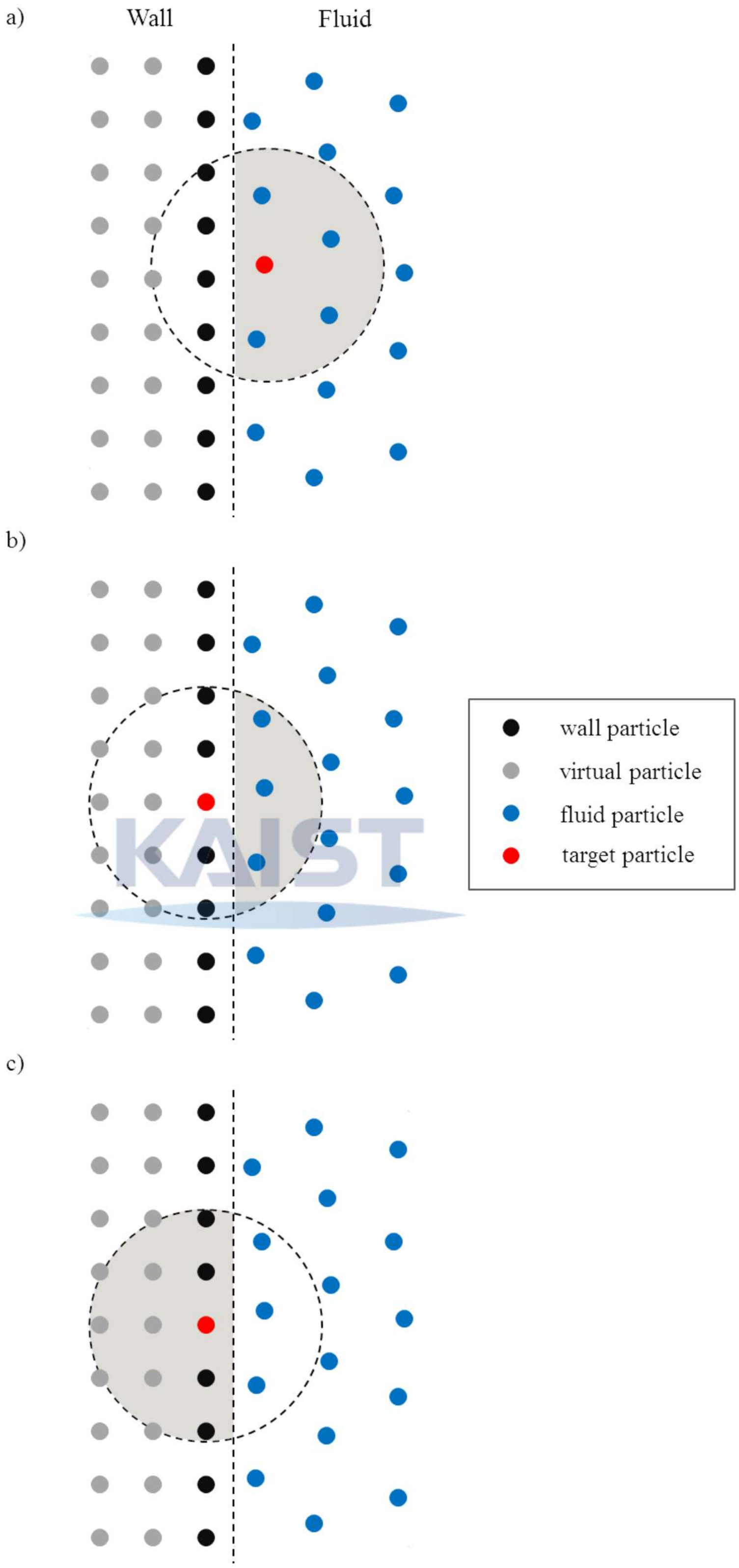

Figure 12. Illustration of density correction steps: (a) fluid-fluid, (b) wall-fluid, (c) wall-wall

3.3. Element resizing

As mentioned earlier, one of the major problems of the existing method is that it is sensitive to smoothing frequency, and frequent application of the method can lead to over-smoothed pressure fields. To avoid this, Seo suggested an additional treatment which enables an adaptive adjustment of smoothing strength by resizing the smoothing element. When the smoothing domain is large, the density is filtered over a large region, which results in stronger smoothing effect. Therefore, one can control the smoothing strength by changing the size of the smoothing element. See ref. [14].

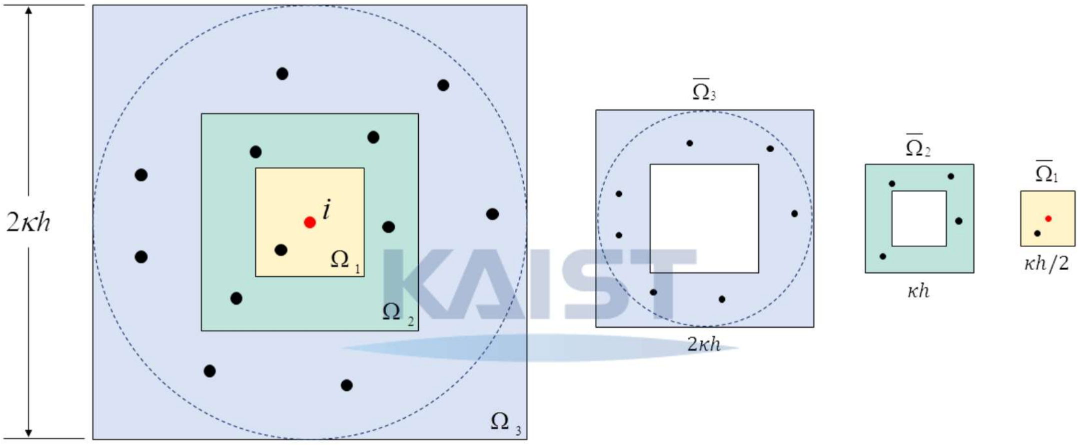

Figure 13. Different sizes of background element for smoothing domain resizing

First, the average density of all particles in the support domain is determined. Once this value is obtained, the original smoothing element is subdivided into three different regions $\bar{\Omega}_1$, $\bar{\Omega}_2$, and $\bar{\Omega}_3$ as shown in figure 13, where $\bar{\Omega}_1 = \Omega_1$, $\bar{\Omega}_2 = \Omega_2 - \Omega_1$, and $\bar{\Omega}_3 = \Omega_3 - \Omega_2$. To see how far away the densities of particles in each region are from the average value, the standard deviation of particles' densities is calculated for each region using

$$\sigma_i = \sqrt{\frac{1}{N_i}\sum_{j=1}^{N_i}(\rho_{avg} - \rho_j)^2}\ , \qquad i = 1, 2, 3 \tag{3.10}$$

where N_i is the total number of particles in region $\bar{\Omega}_i$. The values of σ_1, σ_2, and σ_3 are then compared with each other to determine the size of the smoothing element as

$$\begin{cases} \text{if } \sigma_1 \le \sigma_2 \text{ then, } \Omega = \Omega_1 \\ \text{else if } \sigma_2 \le \sigma_3 \text{ then, } \Omega = \Omega_2 \\ \text{else, } \Omega = \Omega_3 \end{cases}$$

Ω is the new smoothing element whose side length is 2ε. The influence of particles that lie outside of Ω is now neglected during the correction step. Applying this to equation (2.2) we get a modified shape functions of the new smoothing domain as

$$\begin{aligned} H_1(r,s,t) &= (\varepsilon + r)(\varepsilon + s)(\varepsilon + t) / (2\varepsilon)^3 \\ H_2(r,s,t) &= (\varepsilon - r)(\varepsilon + s)(\varepsilon + t) / (2\varepsilon)^3 \\ H_3(r,s,t) &= (\varepsilon - r)(\varepsilon - s)(\varepsilon + t) / (2\varepsilon)^3 \\ H_4(r,s,t) &= (\varepsilon + r)(\varepsilon - s)(\varepsilon + t) / (2\varepsilon)^3 \\ H_5(r,s,t) &= (\varepsilon + r)(\varepsilon + s)(\varepsilon - t) / (2\varepsilon)^3 \\ H_6(r,s,t) &= (\varepsilon - r)(\varepsilon + s)(\varepsilon - t) / (2\varepsilon)^3 \\ H_7(r,s,t) &= (\varepsilon - r)(\varepsilon - s)(\varepsilon - t) / (2\varepsilon)^3 \\ H_8(r,s,t) &= (\varepsilon + r)(\varepsilon - s)(\varepsilon - t) / (2\varepsilon)^3 \end{aligned} \qquad (3.11)$$

Using this method, the size of the smoothing element is adaptively adjusted for each particle, which leads to more reliable solutions.

Chapter 4. Numerical examples

In this section, a validation of 3D approach is carried out through several numerical examples, which includes hydrostatic flow, sloshing tank under sinusoidal excitations, and dam break flow with a cuboid obstacle. The results are compared with analytical solutions for the hydrostatic case. As for the sloshing and dam break problems, the analytical solutions do not exist, and therefore, available experimental data is used for the analysis of numerical results. To verify the performance of the new method, a comparison between the existing method and the proposed method is presented for various smoothing frequencies.

4.1. Hydrostatic

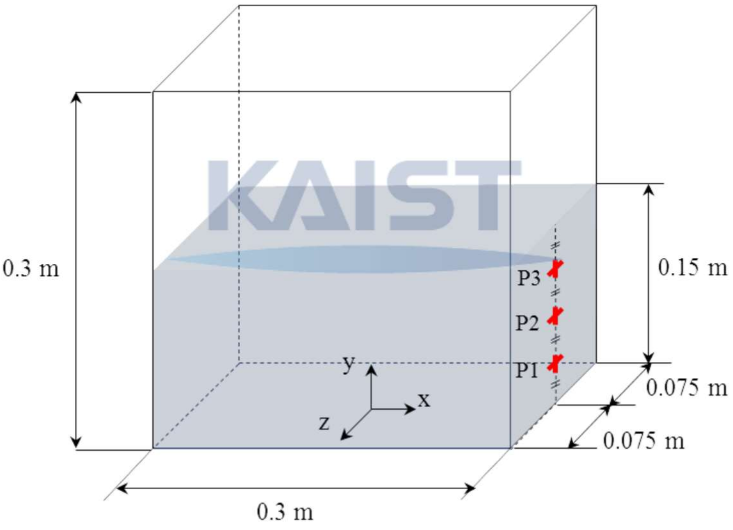

Figure 14. Initial configuration of hydrostatic example

A tank with dimension of 0.3m x 0.3m x 0.15m is filled with water to a height of 0.15 m. Three pressure sensors P1, P2 and P3 are attached to the right side of the wall at $y_1 = 0.0375\,\mathrm{m}$, $y_2 = 0.075\,\mathrm{m}$, and $y_3 = 0.1125\,\mathrm{m}$ as shown in figure 14. A total of 109,363 particles are used to model the tank and the water, in which 58,621 are used for fluid particles and 50,742 for boundary particles. The initial particle spacing is set at $dx = 0.005\,\mathrm{m}$ and the density of water is assumed to be $1000\,\mathrm{kg/m^3}$. Gravitational acceleration of $9.81\,\mathrm{m/s^2}$

is acting downward as the only source of external force. The analytical solution for hydrostatic pressure is given by

$$P = \rho g h \tag{4.1}$$

where $g = 9.81 \mathrm{m/s^2}$ is gravitational acceleration, and h is the depth of the water.

Figure 15 shows the pressure histories at sensor P3 for five different smoothing frequencies, i.e. every 1, 5, 10, 20, and 30 steps. As shown, the results of the existing method vary greatly by the smoothing frequency. The more often the correction is applied, the faster the pressure value deviates from the analytical value as shown in figure 15(a). On the other hand, the results of the proposed method in figure 15(b) show that the solution always converges to the analytical value no matter how often the correction is applied.

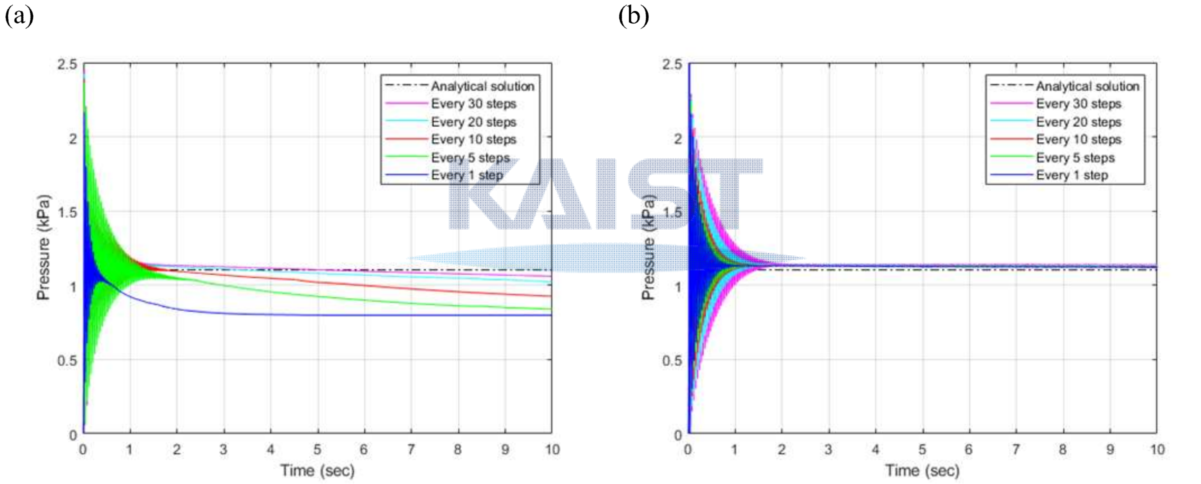

Figure 15. Pressure histories of hydrostatic flow at sensor P3: (a) Shepard's interpolation, (b) proposed method

The pressure histories at all three locations P1, P2, and P3 are plotted in figure 16, where the density is corrected every 10 steps. In figure 16(a), it can be seen that the pressure values of the existing method gradually deviate from the analytical solutions as time passes, whereas those of the proposed method converge to the analytical solutions as shown in figure 16(b). Furthermore, it can be noted that the pressure values of the existing method at different locations appear to converge to one single value, and this is because a frequent use of the existing method leads to the over-smoothing of the density field, which eventually results in a density distribution of a uniform value over the entire fluid domain.

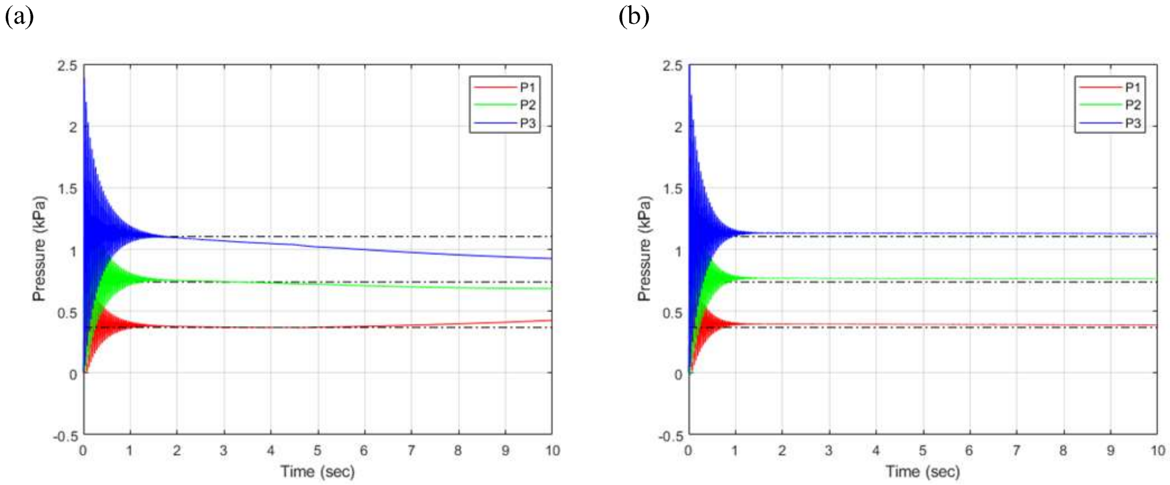

Figure 16. Pressure histories of hydrostatic flow at P1, P2, and P3 when corrected every 10 steps: (a) Shepard's interpolation, (b) proposed method

This trend is more clearly depicted in figure 17, where pressure of water at t = 10s is plotted against the depth of the water. Again, from the results of the existing method, one can see that the pressure values average out more rapidly when the density is corrected more frequently. When the correction is applied every step, the pressure distribution of the fluid becomes almost uniform, depicted as a flat line in figure 17(a). The proposed method, on the other hand, does not show this tendency and produces more reliable results as shown in figure 17(b).

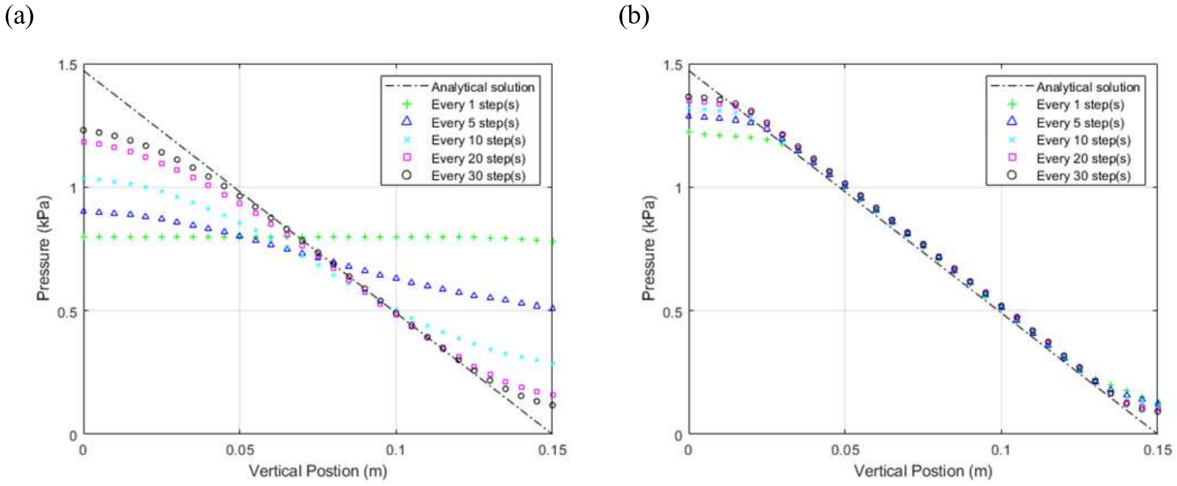

Figure 17. Pressure vs. water height at t = 10s: (a) Shepard's interpolation, (b) proposed method

Finally, snapshots of the pressure distribution of water are given in figure 18 for the case of Shepard's method, proposed method, and the conventional method with no density correction. The pressure scale is represented as a color bar on the right side of each plot, where its range is fixed in all three cases. Solving the hydrostatic equation, the pressure value at the bottom of the tank should be equal to 1471.5 kPa and zero at the free surface. It can be seen that the pressure gradient of the proposed method is close to the analytical solution as shown in figure 18(b), whereas the result of the existing method in figure 18(a) shows the over-smoothed pressure field.

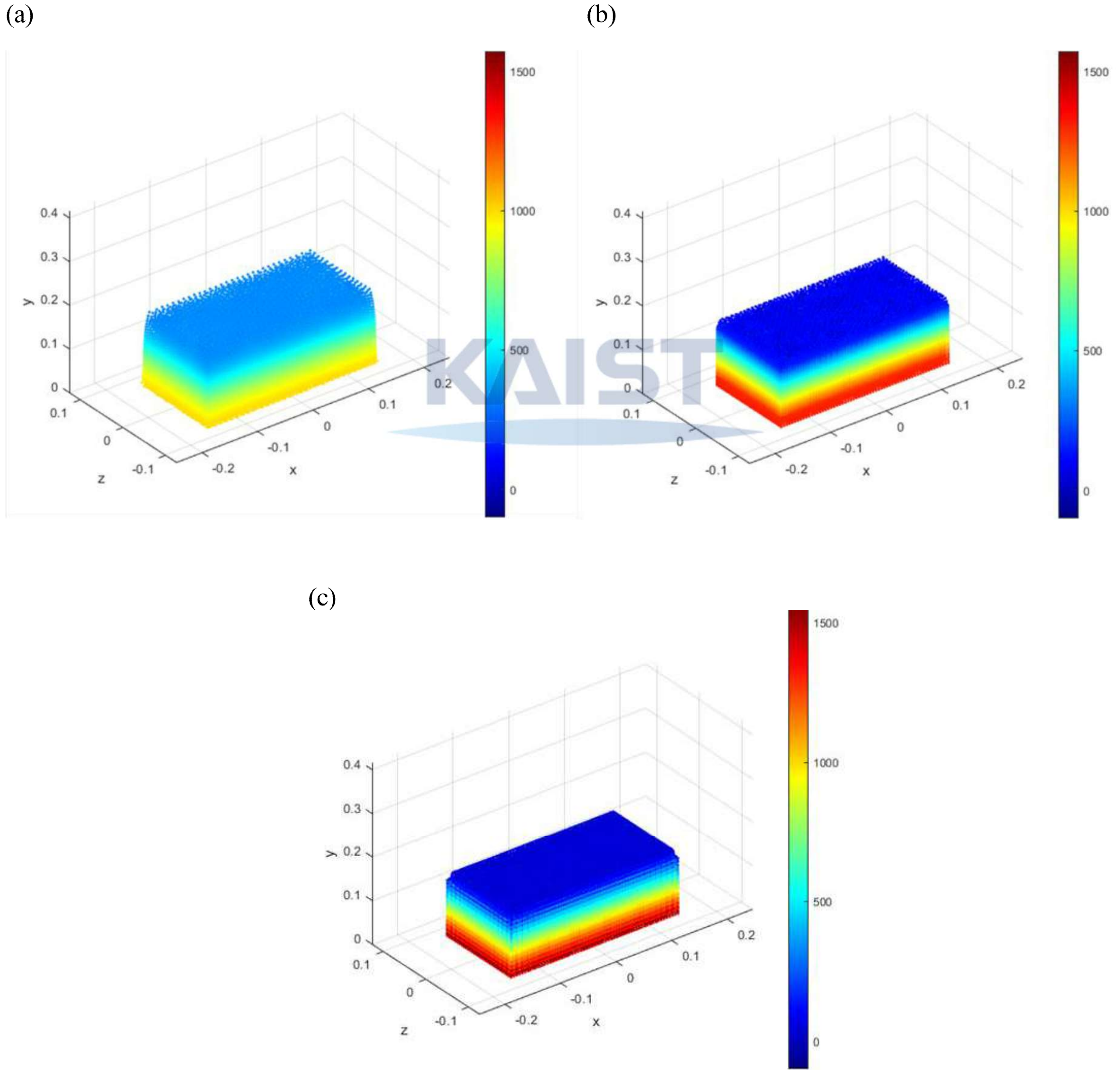

Figure 18. Snapshots of hydrostatic case at t=10s: (a) Shepard's interpolation, (b) proposed method, (c) no correction

4.2. Tank sloshing

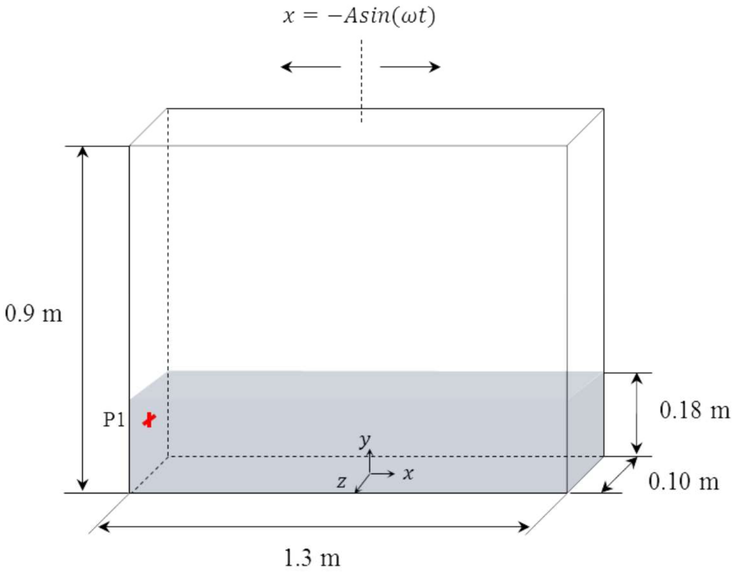

Figure 19. Initial configuration of tank sloshing example

A tank with dimension of 1.3m x 0.9m x 0.1m is filled with water to a height of 0.18m. The tank is subjected to the gravitational acceleration of $9.81 \mathrm{m/s}^2$ and sinusoidal sway motion of $x = -A\sin(\omega t)$ with an amplitude of $A = 0.1\mathrm{m}$ and an angular frequency of $\omega = 2\pi f$, where $f = 0.496\,\mathrm{Hz}$. A pressure sensor P1 is attached to the left side of the wall at $y_1 = 0.12\,\mathrm{m}$. The total number of particles used is 122,161 with 27,379 fluid particles and 94,782 boundary particles, and the initial particle spacing is $dx = 0.01\,\mathrm{m}$. The initial set up of the tank sloshing case is illustrated in figure 19.

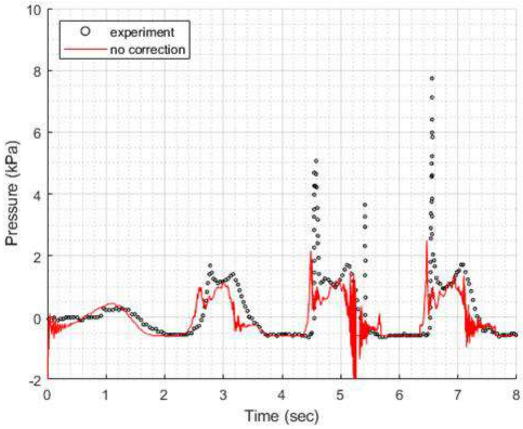

Figure 20. Pressure history of sloshing flow at P1 when no density correction is applied

In figure 20, the pressure history of the conventional SPH without any density treatments is plotted against experimental data [24]. As time goes on, the flow becomes more violent and the numerical instability induced by this violent motion of flow becomes more severe. This leads to large oscillations in density and pressure fields, which can be observed in figure 20.

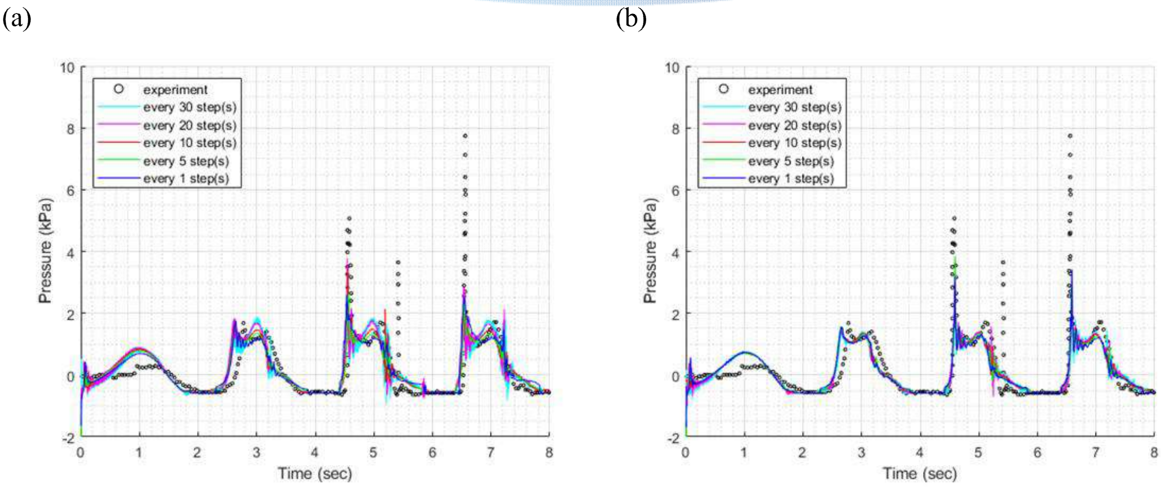

Figure 21. Pressure histories of sloshing flow at P1: (a) Shepard’s interpolation, (b) proposed method

In figure 21, the pressure histories of both density correction methods for five different smoothing frequencies are plotted against experimental data. Compared to the results of the conventional SPH, both methods effectively reduce the oscillations in pressure field and in turn produce better solutions that well match the experimental data. Yet, the performance of the existing method varies greatly when different smoothing frequency is selected, whereas the results of the proposed method are consistent and relatively independent of the smoothing frequency.

4.3. Dam break

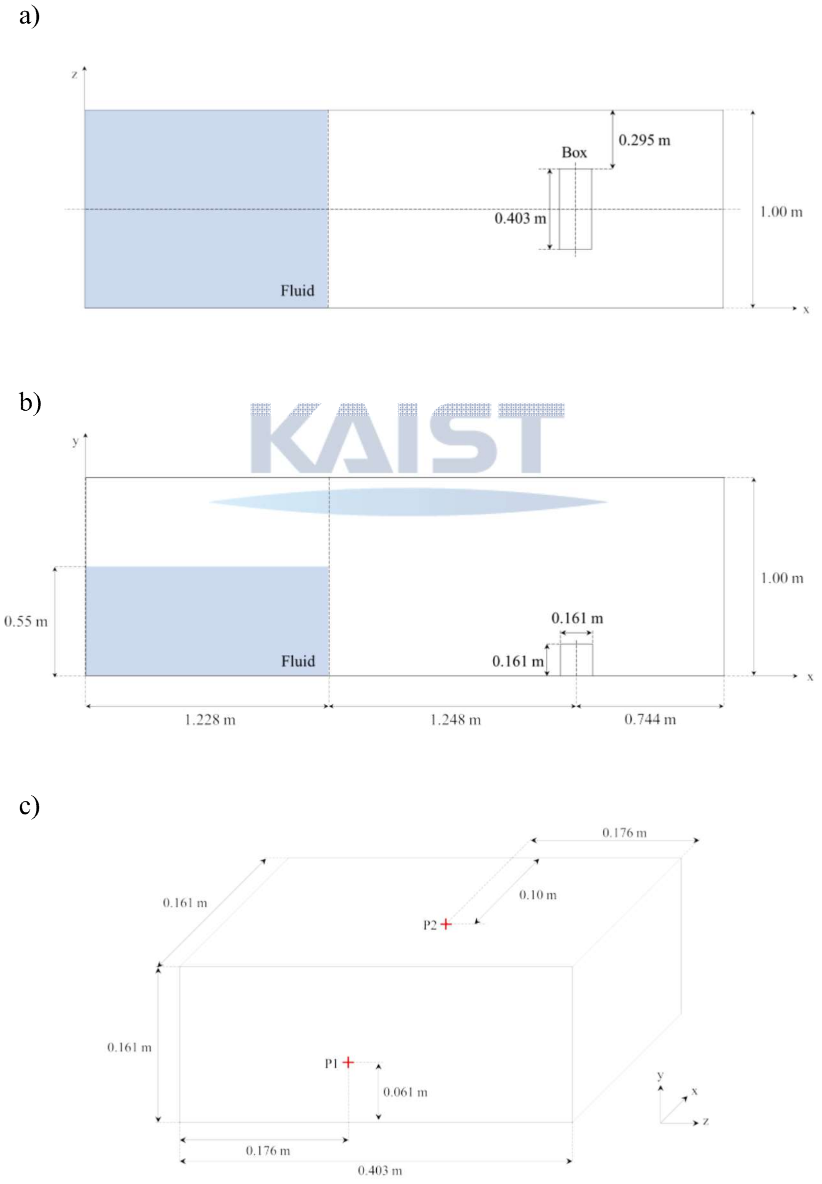

Figure 22. Initial configuration of dam break example: (a) top view, (b) side view, and (c) sensor box dimension

A reservoir of water with initial dimension of 1.228 m x 0.55 m x 1.0 m is released in a water tank of size 3.22 m x 1.0 m x 1.0 m. The density of water is $1000\,\mathrm{kg}/\mathrm{m}^3$ and the total number of particles used is 268,828 with 118,048 fluid particles and 150,780 boundary particles. A small box of the size 0.161 m x 0.161 m x 0.403 m with two pressure sensors is placed 0.744 m away from the wall opposite the reservoir. The gravitational force is acting downward as the only source of external load. The initial configuration of the problem with the sensor locations is illustrated in figure 22.

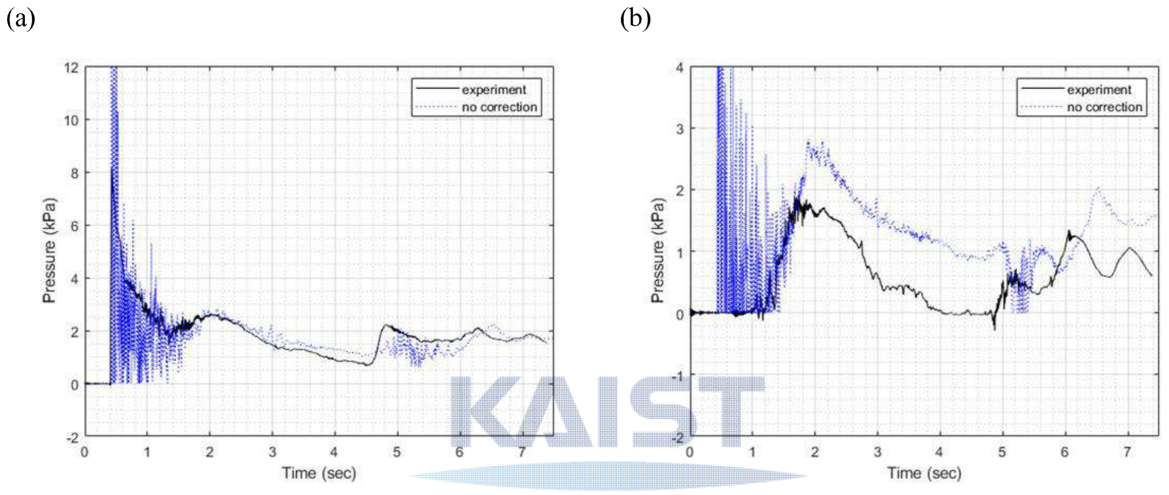

Figure 23. Pressure histories of dam break flow at (a) P1 and (b) P2, when no desnsity correction is applied

Figure 23 shows the pressure histories of conventional SPH at points P1 and P2. A large pressure oscillations are observed near the point of wall impact, which occurs around t = 0.5s. At impact, the flow experiences shock pressure caused by breaking waves, which is a major source of unphysical oscillations as discussed in section 2.2.2. Such error in pressure field often leads to inaccurate solutions as depicted in figure 23(b).

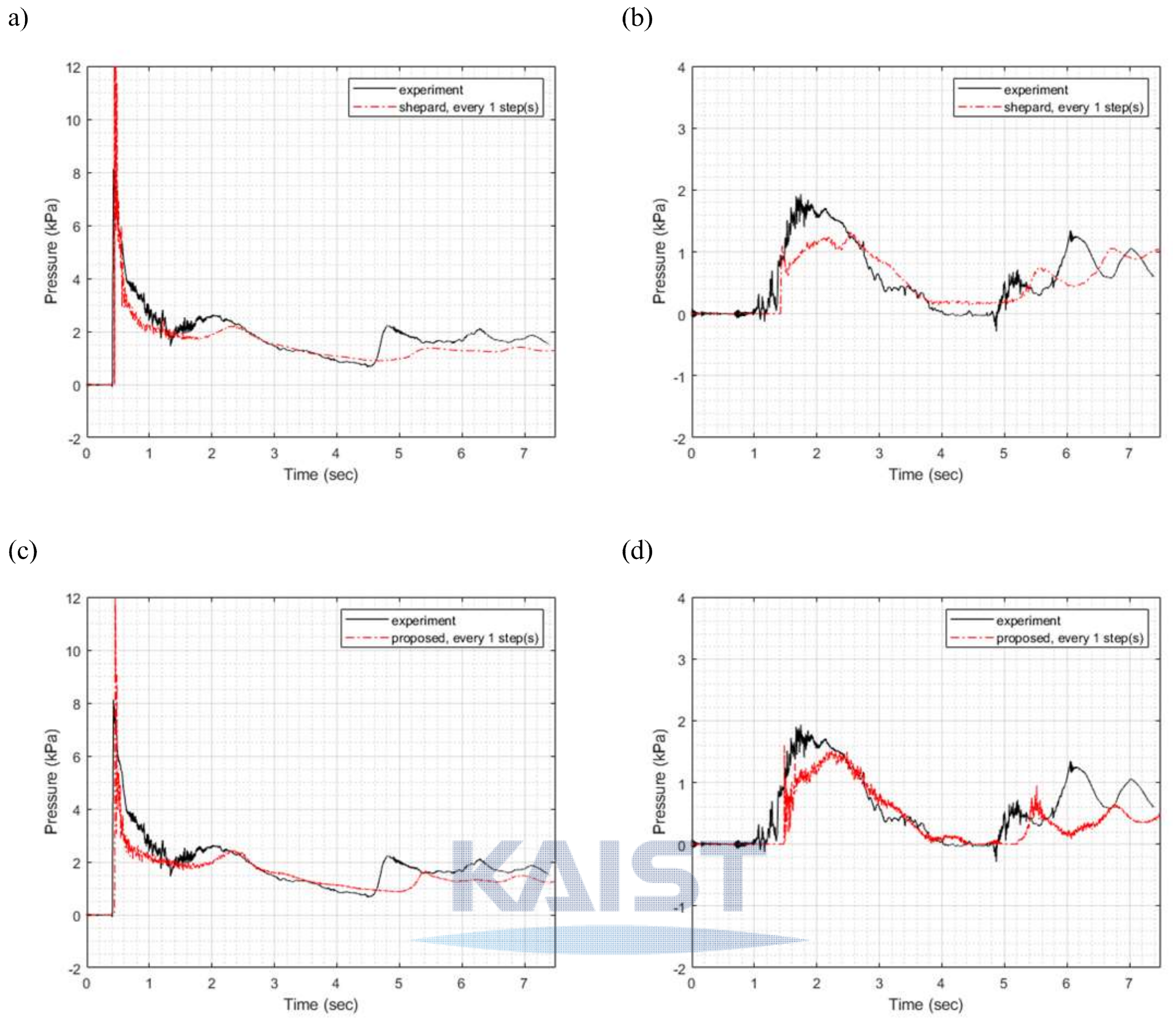

Figure 24. Pressure histories of dam break flow when density is corrected every step: Shepard's interpolation at (a) P1, (b) P2, proposed method at (c) P1, (d) P2

In figure 24, the pressure histories at points P1 and P2 are plotted against experimental data [25] for smoothing frequency of every 1 step. Again, the unphysical oscillations in figure 23 are effectively reduced by both methods. At P1, the point of wall impact and the peak pressure at impact are well predicted by both methods as shown in figures 24(a) and 24(c). However, the result of the existing method at P2 show slight deviation from the experimental results, especially after t = 2s as shown in figure 24(b).

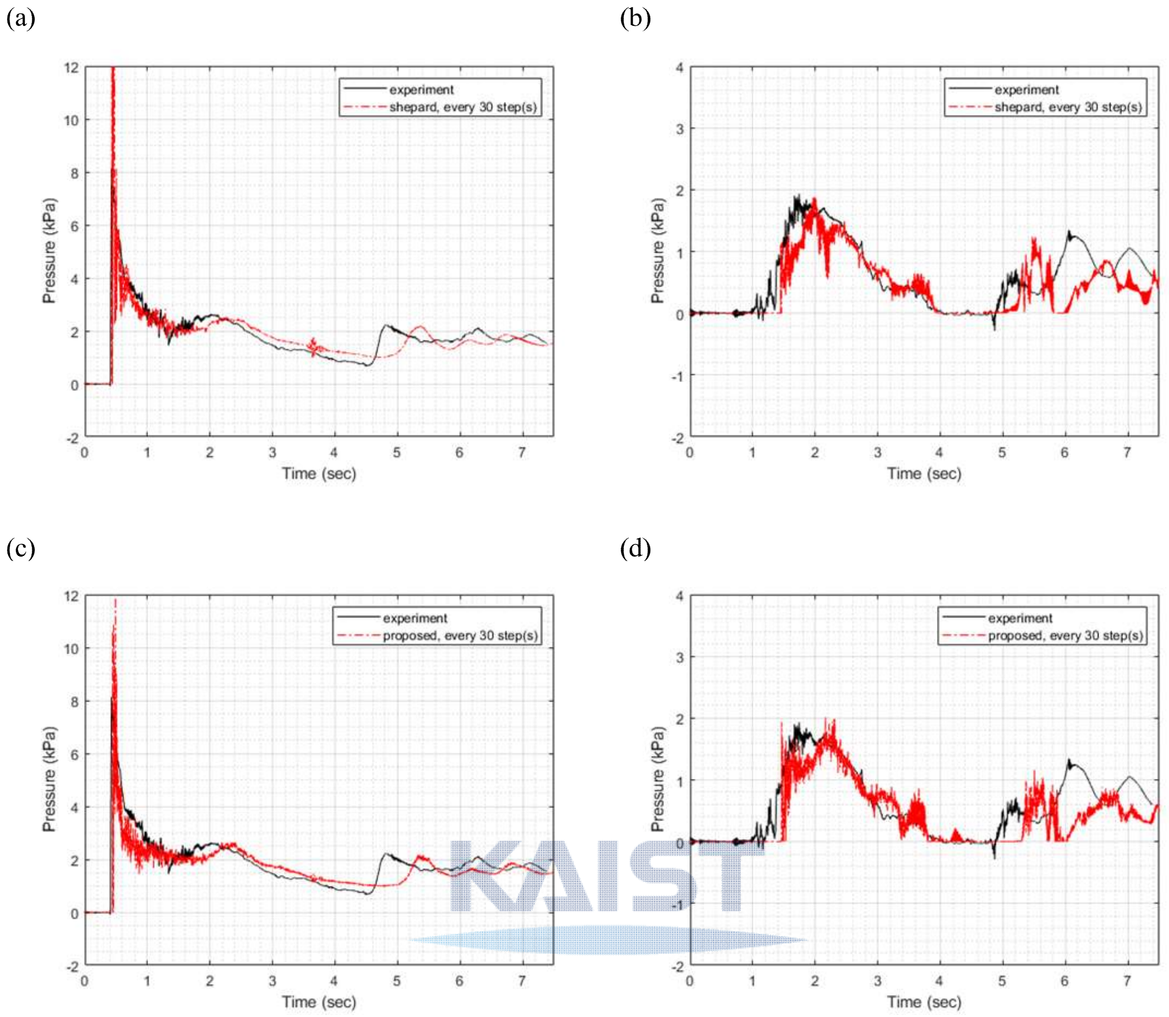

Figure 25. Pressure histories of dam break flow when density is corrected every 30 steps: Shepard's interpolation at (a) P1, (b) P2, proposed method at (c) P1, (d) P2

In figure 25, the same plots are obtained for smoothing frequency of every 30 steps. In this case, the results of the existing method and the proposed method are in good agreement with each other for both P1 and P2. Although it can be seen that the results of the proposed method is not entirely independent of the smoothing frequency, the effect of the smoothing frequency on the performance of the proposed method is far less severe than that of the existing method.

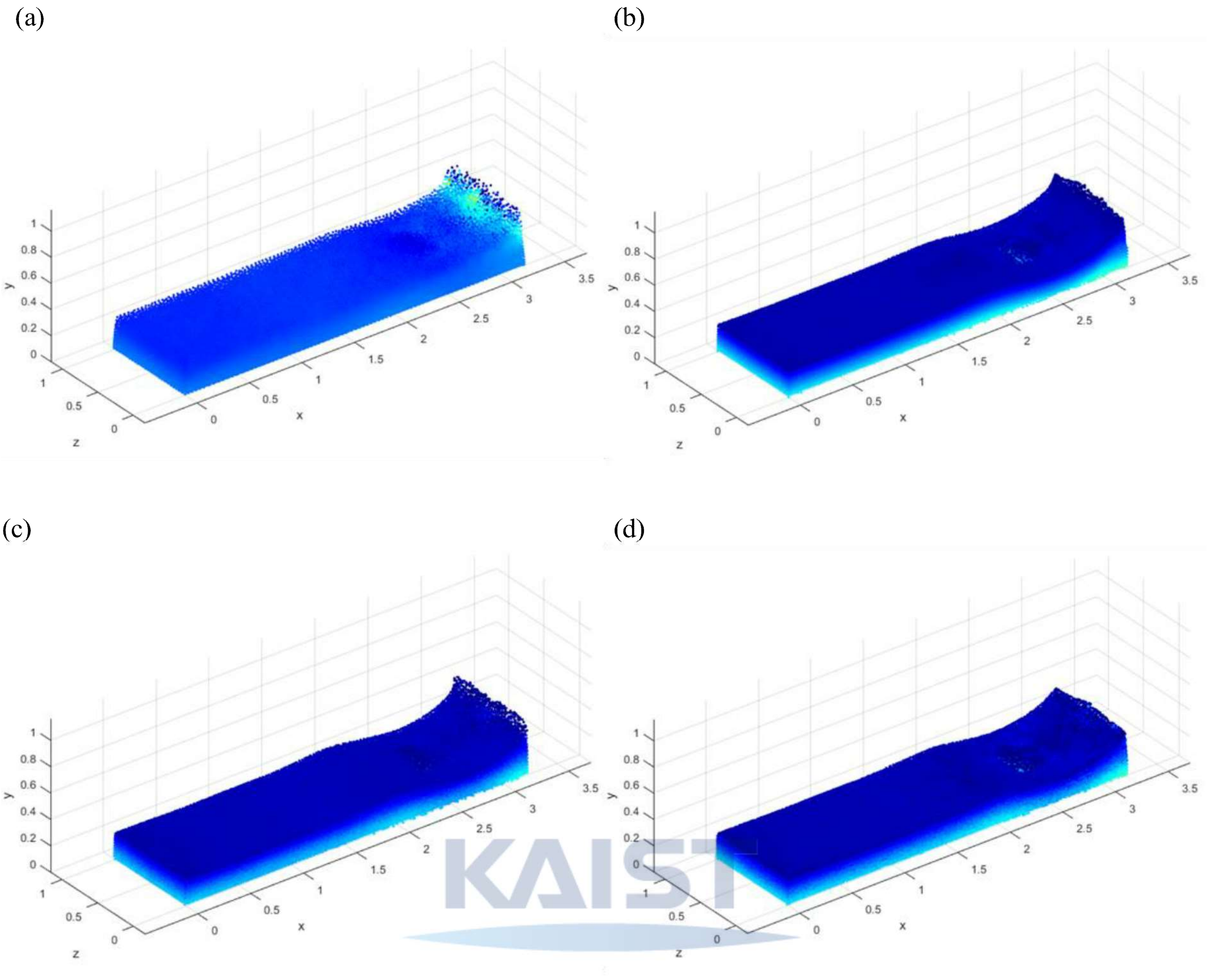

Figure 26. Snapshots of dam break flow at t = 6s: Shepard's interpolation is applied (a) every step, (b) every 30 steps, proposed method is applied (c) every step, (d) every 30 steps

To see the global behavior of the flow, the pressure distributions at t = 6s is plotted for each method with different smoothing frequencies. While the results of all other cases show similar pressure distributions, the over-smoothed pressure field is observed for the case when the existing method is applied every step, as shown in figure 26(a).

Chapter 5. Conclusion

In this paper, the 3D extension of the 2D density correction method using particle-attached element is presented. A 8-node cubic element of varying sizes is generated for each particle in each smoothing step, and the density of an individual particle is re-calculated by interpolating the nodal density values of the particle-attached element using linear 3D shape functions. Several numerical schemes are adopted for treatments of solid boundary and numerical oscillations, and the GPU implementation of the SPH method with an efficient neighbor search method is also adopted for faster simulation. As a result, a speedup of up to 6.17 is achieved for 1 million particles, and this number is expected to go up with the increasing number of particles.

To verify the feasibility of the new density correction method, three examples of hydrostatic, tank sloshing and dam break problems are solved numerically, where the results are compared with experimental data, analytical solutions, and numerical solutions obtained using the existing method. In the case of hydrostatic example, it is shown that the proposed method effectively eliminates the over-smoothing problem. For the tank sloshing example, the results of the proposed method are relatively consistent and do not vary as much by smoothing frequency compared to the existing method. In the dam break example, both the over-smoothing problem and the frequency dependent problem of the existing method are effectively reduced by using the proposed method, which results in more reliable solutions. Overall, the results of the proposed method not only show good agreement with reference data, but also show that the proposed method has better convergence characteristics compared to the existing method.

Bibliography

[1] Lucy, L.B., “A numerical approach to the testing of the fission hypothesis,” *Astronomical Journal*, 82, 1013-1024, (1977)

[2] Gingold, R.A., Monaghan, J.J., “Smoothed particle hydrodynamics: theory and application to non-spherical star,” *Monthly Notices of the Royal Astronomical Society*, 181, 375-389, (1977)

[3] Libersky, L.D., Petschek, A.G., “Smoothed particle hydrodynamics with strength of materials,” *in: H. Trease, J. Fritts and W. Crowley eds., Proceedings, The Next Free Lagrange Conference, Springer-Verlag, NY*, 395, 248-257, (1991)

[4] Monaghan, J.J., “Simulating free surface flow with SPH,” *Journal of Computational Physics*, 110(2), 399-406, (1994)

[5] Monaghan, J.J., “SPH compressible turbulence,” *Monthly Notices of the Royal Astronomical Society*, 335(3), 375-389, (2002)

[6]Colagrossi, A., Landrini, M., “Numerical simulation of interfacial flows by smoothed particle hydrodynamics,” *Journal of Computational Physics*, 191(2), 448-475, (2003)

[7] Liu, M.B., Shao, J.R., Chang, J.Z., “On the treatment of solid boundary treatment in smoothed particle hydrodynamics.” *Science China Technological Sciences*, 55, 244-254, (2012)

[8] Shadloo, M.S., Zainali, A., Yildiz, M., Suleman, A., “A robust weakly compressible SPH method and its comparison with an incompressible SPH,” *International Journal for Numerical Methods in Engineering*, 89, 939-956, (2012)

[9] Asai, M., Mahmoud Aly, A., Sonoda, Y., Sakai, Y., “A Stabilized Incompressible SPH method by relaxing the density invariance condition,” *Journal of Applied Mathematics*, Special issue, (2012)

[10] Monaghan, J.J., Gingold, R.A., “Shock simulation by the particle method SPH,” *Journal of Computational Physics*, 52(2), 374-389, (1983)

[11] Molteni, D., Colagrossi, A., “A simple procedure to improve the pressure evaluation in hydrodynamic context using the SPH,” *Computer Physics Communications*, 180(6), 861-872, (2009)

[12] Monaghan, J.J., “On the problem of penetration in particle methods,” *Journal of Computational Physics*, 82, 1-15, (1989)

[13] Ozbulut, M., Yildiz, M., Goren, O., “A numerical investigation into the correction algorithms for SPH method in modeling violent free surface flows,” *International Journal of mechanical sciences*, 79, 56-65, (2014)

[14] Seo, H.D., Park, H.J., Kim, J.I., Lee, P.S., “The particle-attached element interpolation for density correction in smoothed particle hydrodynamics,” *Advances in Engineering Software*, submitted, (2020).

[15] Liu, M.B., Liu, G.R., Lam, K.Y., “Constructing smoothing functions in smoothed particle hydrodynamics with applications,” *Journal of Computational and Applied Mathematics*, 155(2), 263-284, (2003)

[16] Yang, X.F., Liu, M.B. “Improvement on stress instability in smoothed particle hydrodynamics”, *Acta Physica Sinica*, 61(22), 2012

[17] Becker, M., Matthias, T., “Weakly compressible SPH for free surface flows” *Proceedings of the 2007 ACM SIGGRAPH: Eurographics symposium on Computer animation*, 209-217, (2007)

[18] Libersky, L.D., Petschek, A.G., Carney, T.C., Hipp, J.R., Allahdadi, F.A., "High strain Lagrangian hydrodynamics: a three-demensional SPH code for dynamic material response," *Journal of Computational Physics,* 109(1), 67-75, (1993)

[19] Liu, M.B., Liu, G.R., Lam, K.Y., "Investigations into water mitigations using a meshless particle method," *Shock Waves*, 12(3), 181-195, (2002)

[20] Shao, J.R., Li, H.Q., Liu, G.R., Liu, M.B., "An improved SPH method for modeling liquid sloshing dynamics," *Computers and Structures*, 100-101, 18-26, (2012)

[21] Chen, Z., Zong, Z., Li, H.T., Li, J., "An investigation into the pressure on solid walls in 2D sloshing using SPH method", *Ocean Engineering*, 59, 129-141, (2013)

[22] Liu, G.R., Liu, M.B., "Smoothed particle hydrodynamics: a meshfree particle method," *World Scientific*, (2003)

[23] Morris, J.P., Fox, P.J., Zhu, Y., "Modeling Low Reynolds Number Incompressible Flows Using SPH," *Journal of Computational Physics*, 136(1), 214-226, (1997)

[24] Rafiee, A., Pistani, F., Thiagarajan, K., "Study of liquid sloshing: numerical and experimental approach," *Computational Mechanics*, 47, 65-75, (2011)

[25] Kleefsman, K.M.T., Fekken, G., Veldman, A.E.P., Iwanowski, B., Buchner, B., "A Volume-of-Fluid based simulation method for wave impact problems," *Journal of Computational Physics*, 206(1), 363-393, (2005)