Pós-Graduação em Ciência da Computação

André Luiz Buarque Vieira e Silva

A fluid simulation system based on the MPS method

Federal University of Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife
2018

André Luiz Buarque Vieira e Silva

A fluid simulation system based on the MPS method

A M.Sc. Dissertation presented to the Center of Informatics of Federal University of Pernambuco in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Field of study: Virtual reality

Supervisor: Veronica Teichrieb

Recife
2018

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

S586f Silva, André Luiz Buarque Vieira e
A fluid simulation system based on the MPS method / André Luiz Buarque Vieira e Silva. – 2018.
77 f.: il., fig., tab.

Orientadora: Veronica Teichrieb.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2018.
Inclui referências.

1. Computação gráfica. 2. Realidade virtual. I. Teichrieb, Veronica (orientadora). II. Título.

006.6 CDD (23. ed.) UFPE- MEI 2018-117

André Luiz Buarque Vieira e Silva

A Fluid Simulation System Based on the MPS Method

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação

Aprovado em: 24/08/2018.

BANCA EXAMINADORA

_____

Prof. Dr. Silvio de Barros Melo
Centro de Informática / UFPE

_____

Prof. Dr. José Maria Andrade Barbosa
Departamento de Engenharia Mecânica / UFPE

_____

Profa. Dra. Veronica Teichrieb
Centro de Informática/UFPE
(**Orientadora**)

Dissertação de Mestrado apresentada por **André Luiz Buarque Vieira e Silva** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título "**A fluid simulation system based on the MPS method**" **Orientador: Veronica Teichrieb** e aprovada pela Banca Examinadora formada pelos professores:

Prof. Silvio de Barros Melo
Centro de Informática / UFPE

Prof. José Maria Andrade Barbosa
Departamento de Engenharia Mecânica, CTG / UFPE

Profa. Orientadora: Veronica Teichrieb
Centro de Informática / UFPE

Visto e permitida a impressão.
Recife, 17 de Agosto de 2018.

Prof. Aluizio Fausto Ribeiro Araújo
Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

ACKNOWLEDGEMENTS

É com muita satisfação que chego ao final desse mestrado, que é uma grande compilação de conhecimentos obtidos nos últimos anos através do projeto navegante. Obviamente essa conquista não é só minha, portanto, quero agradecer primeiramente à minha família, meu pai Luiz, minha mãe Maria das Neves e meu irmão Felipe por sempre me apoiarem incondicionalmente e a todos os outros parentes (avós, tios e primos) que sempre estiveram presente. Quero agradecer à minha namorada Mirella pelos essenciais incentivos e inúmeras conversas que me auxiliaram e me guiam. Gostaria também de agraceder a orientação fundamental da professora Veronica Teichrieb e também agradecer a cada um dos extraordinários colaboradores do Voxar Labs. Por fim, não posso esquecer meus amigos de Apoio, bandas e de graduação que também tiveram sua importante parcela de contribuição!

Obrigado!

ABSTRACT

Fluid flow simulation is a high active area in Computer Graphics and Virtual Reality, with applications in a wide range of engineering problems. In this scenario, meshless methods like the Moving Particle Semi-implicit (MPS) are a great alternative to deal with large deformations and free-surface flow, problems that usually impose the traditional mesh-based methods to perform inefficiently. This dissertation presents a stable, accurate and parallelized MPS-based technique which benefits from different advances in the MPS literature, and also from parallel computing, to obtain a method that can be adapted for a wide variety of scenarios. The proposed technique can simulate fully incompressible/weakly compressible fluid under different fluid behaviors such as two levels of compressibility, different fluid' kinematic viscosity, turbulent flows and multiphase interaction. The method was evaluated under classical scenarios like Water Drop, Dam Break flow, R-T instability and Oil Spill, presenting comparable results to the State-Of-The-Art methods. The method and its variations are also integrated on a single solution which can switch on improvements such as better momentum conservation, more precise discretization of differential operators and less erroneous pressure oscillations through a user-friendly graphical interface. This enables a practical selection of models, approaches and parameter tuning, from, for instance, a stable physically coherent free-surface incompressible fluid flow simulation, to a GPU-accelerated multiphase free-surface weakly compressible flow simulation. Based on three different implementations (single-core CPU as the reference, multi-core CPU with OpenMP and multi-core GPU with CUDA for performance improvements), it is shown that the OpenMP-enabled weakly compressible approach achieves a speedup of 2.02 times and the fully incompressible approach of 1.82 times. The CUDA-enabled weakly compressible approach achieves a speedup of 3.15 times while the fully incompressible approach of 2.23 times.

Key-words: MPS. Numerical improvement. Turbulence. Multiphase. CUDA. OpenMP.

RESUMO

A simulação de fluidos é uma área altamente ativa em computação gráfica e realidade virtual, com aplicações em uma ampla gama de problemas de engenharia. Nesse cenário, métodos sem malha, como o Moving Particle Semi-implicit (MPS), são uma ótima alternativa para lidar com grandes deformações e movimento de superfície livre, problemas que normalmente fazem com que os métodos tradicionais baseados em malha executem de forma ineficiente. Esta dissertação apresenta uma técnica baseada em MPS estável, precisa e paralelizada, que se beneficia de diferentes avanços na literatura do MPS, e também da computação paralela, para obter um método que pode ser adaptado para uma ampla variedade de cenários. A técnica proposta pode simular fluidos totalmente incompressíveis ou fracamente compressíveis sob diferentes comportamentos, como os dois níveis de compressibilidade mencionados, viscosidade cinemática de diferentes fluidos, escoamentos turbulentos e interação multifásica. O método foi avaliado em cenários clássicos como o da Gota de Água, a Quebra de Barragem, Instabilidade R-T e Derramamento de Óleo, apresentando resultados comparáveis aos métodos do Estado da Arte. O método e suas variações também são integrados em uma única solução em que podem ser ativadas melhorias, como melhor conservação de momento, discretização mais precisa de operadores diferenciais e menos oscilações errôneas da pressão do fluido, isso tudo por meio de uma interface gráfica. Isso permite uma seleção prática de simuladores, abordagens e ajuste de parâmetros, de, por exemplo, uma simulação de escoamento de fluido incompressível de superfície livre fisicamente coerente, a uma simulação de fluxo multifásico acelerada por GPU. Com base em três implementações diferentes (CPU de núcleo único como referência, CPU multi-core com OpenMP e GPU multi-core com CUDA para melhorias de desempenho), é mostrado que a abordagem fracamente compressível acelerada com OpenMP atinge uma aceleração de 2.02 vezes e a abordagem totalmente incompressível de 1.82 vezes. A abordagem fracamente compressível habilitada para CUDA alcança uma aceleração de 3.15 vezes, enquanto a abordagem totalmente incompressível de 2.23 vezes.

Palavras-chaves: MPS. Melhorias numéricas. Turbulência. Multifase. CUDA. OpenMP.

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

D_S	Number of space dimensions
$\mathbf{g}$	Gravitational acceleration
H	Initial height
L	Initial width
n	Particle number density
n_0	Initial particle density (constant)
n^*	The particle number density calculated in an intermediate step
n'	Correction value of particle number density
p	Pressure
r	Distance between two particles
r_e	Radius of interaction
r_0	Initial spacing distance
$\mathbf{r}_i$	Position in space of particle i
r_{ij}	Distance between particle i and j
t	Time
u_{ij}	Difference between the velocity of particle i and j in the x direction
$\mathbf{u}$	Fluid velocity vector
$\mathbf{u}^*$	The velocity field calculated in an intermediate step
v_{ij}	Difference between the velocity of particle i and j in the y direction
$w(r)$	Kernel function
w_{ij}	Difference between the velocity of particle i and j in the z direction
W	Kernel function
x_{ij}	Distance between particle i and j in the x direction
y_{ij}	Distance between particle i and j in the y direction

z_{ij}	Distance between particle i and j in the z direction
β	Free surface coefficient
δ	Dirac delta function
ν	Kinematic viscosity
ρ	Fluid density
ρ^*	The density calculated in an intermediate step
φ	Represents a physical quantity
Ω	Influence area

CONTENTS

1 INTRODUCTION

Some of the most common problems in naval hydrodynamics involve the study of fluid flow. For this, it is necessary to deal with large deformations near the surface such as those presented in a good portion of computational mechanics problems (CLEARY; PRAKASH; HA, 2006).

Conventional methods, as the Finite Element Methods (FEM), Finite Difference Methods (FDM) and other mesh-based methods, are considered well consolidated and accurate. However, they are relatively inefficient when dealing with certain problems where it is required the simulation of large deformations. The best approach considered to deal with large deformations and the moving discontinuities caused by them is to constantly regenerate the mesh in order to keep its discontinuities coincident through the simulation (BELYTSCHKO et al., 1996a).

Clearly, this constant remeshing makes the process quite expensive in terms of computation, sometimes even causing accuracy degradation (JOHNSON; TEZDUYAR, 1999). As an attempt to reduce those issues, hybrid methods that also use discrete elements, called particles, were proposed. One example is the Particle Finite Elements Method (PFEM) (OÑATE et al., 2004); another promising alternative, which has presented great potential over the years, are the entirely meshfree methods. They enable, mainly, that free-surface flow can be discretized and solved the Navier-Stokes equations without the need of a grid of any kind, such as in the work of Frey and Alauzet (FREY; ALAUZET, 2005), achieving flexibility in situations where the classic methods are too complex. Each particle carries a set of physics quantities and constitutive properties, such as mass, velocity and position, and they are responsible for characterizing the system state and its evolution through time. An interesting advantage of the meshless methods with Lagrangian characteristics, is that it allows an easy tracking of each particle's quantities in any step of the simulation.

Some of the techniques fully free of meshes are the Moving Particle Semi-implicit method (MPS) and the well-known Smoothed Particle Hydrodynamics (SPH). The SPH was designed in the 1970s by Lucy (LUCY, 1977) and Gingold and Monaghan (GINGOLD; MONAGHAN, 1977) and intended to astrophysics applications. The first method mentioned, the MPS, was introduced in 1996 with the work of Koshizuka and Oka (KOSHIZUKA; OKA, 1996) and it was idealized to simulate the flows of incompressible fluids, which refers to a fluid that its material density is constant within a fluid parcel. In many scenarios the changes in temperature and pressure are so small that the density fluctuation is negligible; in such cases the flow may be modeled as incompressible. Its main difference from the original SPH method, which is considered a notable advantage for the MPS method, is that the calculations adopt a semi-implicit predictor-corrector model (which later has been similarly used in some incompressible SPH methods (XU; STANSBY;

LAURENCE, 2009)). However, the SPH has been preferred in Computer Graphics (CG) and Virtual Reality (VR) applications due the high computational load occasioned by the precise MPS calculations, including solving the Poisson Pressure Equation (PPE).

The MPS method was chosen in this work to be in-depth studied, implemented and accelerated due to its appealing intrinsic incompressibility since this type of fluid flow presents environmental importance (SHAO; LO, 2003) and appears in many industrial applications (KOSHIZUKA; TAMAKO; OKA, 1995). Recent papers by Gotoh and Khayyer (GOTOH; KHAYYER, 2018) (GOTOH; KHAYYER, 2016) perform surveys on the particle methods for fluid simulation. In (GOTOH; KHAYYER, 2016) the authors highlight the current achievements and future perspectives for projection-based particle methods, that is, methods that require solving a PPE such as the MPS method and the incompressible SPH method. In this work the authors present a set of papers regarding the applicability of the MPS method in ocean engineering, including wave breaking (GOTOH; SAKAI, 1999) (KHAYYER, 2008), wave overtopping (GOTOH et al., 2005), wave impact (KHAYYER; GOTOH, 2009) (LEE et al., 2011), green water on ships (SHIBATA; KOSHIZUKA, 2007), sediment transport (GOTOH; SAKAI, 2006), landslide-generated waves (FU; JIN, 2015) and fluid–structure interactions (SHIBATA et al., 2012) (HWANG et al., 2014). In (GOTOH; KHAYYER, 2018), the authors expose the latest advancements related to particle methods applied to coastal and ocean engineering. Weakly compressible (or fully explicit) methods are also considered for the review. One example is shown in (TAYEBI; JIN, 2015) in which the Moving Particle Explicit (MPE) method is proposed by solving an appropriate equation of state in a fully explicit form to obtain the particles pressure. Other recent applications of the MPS method brought up by the referred review are related to oil spilling (DUAN et al., 2017) and the swash beach process (HARADA et al., 2017).

It is possible to say that the present study has a correlation with the regional and national context, where many people suffer from water-related disasters and accidents, due to for instance, the breaking of dams or the heavy rains, mainly during the winter period, when they are more intense and appear with a higher frequency. Also, another concern in this study is to help understand major fluid-related disasters such as oil spilling in oceans by precisely simulating it. Considering this, the tests used in this work are the classic Dam Break, the Oil Spill, the Water Drop and the Rayleigh-Taylor instability, where the last two are more focused in assessing the method's precision. Further studies that analyse the consequences of such disasters can greatly benefit from the system developed in this research.

The main issues of meshfree methods, in general, are in the modeling of solid boundary interaction, fluid flow and, in the specific case of the MPS method, spurious pressure oscillation of the particles (KHAYYER; GOTOH, 2009). Therefore, several solutions have been proposed in the literature, such as local particle refinement and corrected formulations (KHAYYER; GOTOH, 2008; KHAYYER; GOTOH, 2010; KHAYYER; GOTOH, 2011; KHAYYER;

GOTOH, 2012).

Many improvements and adaptations of the original method of both SPH and MPS techniques have been proposed in order to adequate them to the simulation of various kinds of physical phenomena or, more commonly, to get better stability and accurate calculations. In the methodology chapter of this dissertation, the improvements used in this work are addressed. A set of modifications to the MPS and the SPH method can be seen in the works of (SILVA et al., 2015) and (ALMEIDA et al., 2016).

1.1 OBJECTIVES

A significant disadvantage of fluid simulation models that value numerical precision is time spent in the application execution, more specifically in the simulation generation (ZHU et al., 2011). The challenge of dealing with this problem has been diminished through the use of computational platforms that make possible to benefit from the various processing cores of a CPU and a Graphics Processing Unit (GPU). Some works provide various kinds of performance speedups but always focusing on the standard MPS method (HORI et al., 2011; ZHU et al., 2011; TANIGUCHI; SATO; CHENG, 2014).

In this work, a significant literature review is made, MPS applications are visited, and the method proposed by Gotoh (GOTOH, 2013) (CMPS-HS-HL-ECS) is implemented. The multi-viscosity and multi-density method proposed by (SHAKIBAEINIA; JIN, 2012), *MPARS* (SHAKIBAEINIA, 2012), is incorporated so that both methods benefit from each other models and enhancements almost entirely. The system is extended to 3D, including its accelerations structures and models, such as the neighbors search strategy structure, and the turbulence, multi-density and viscoplastic models. Initially the whole implementation is done sequentially to run in CPU and subsequently the multiple cores in the CPU are explored developing a parallelized version through OpenMP (DAGUM; MENON, 1998). Later, using CUDA (NVIDIA, 2007), the same code is implemented to run in GPU, exploiting its parallelism and aiming at near iterative rates for at least relatively low particle number cases (order of 10^3 particles). Lastly, the simulation is rendered using a screen space approach for illustration purposes. This points in a direction where not only truly incompressible about also quite accurate fluid simulation methods can be used in VR and CG applications, allowing even more realism, now focusing not only on the rendering quality but also on the physics.

Despite the high level of applicability of the MPS method as shown in the beginning of this chapter, only a few solutions or softwares were found that implemented the method, such as (Fuji Technical Research Inc., 2013; Prometech Software, 2014; SHAKIBAEINIA, 2012), in which (Fuji Technical Research Inc., 2013; Prometech Software, 2014) are paid. Differently from them, the system developed in this research allows combining significant numerical improvements with diverse fluid behaviors, such as multiphase and viscoplastic simulations and distinct code optimization/acceleration in different levels, such as multithreaded

CPU and multithreaded GPU execution, all of this in two or three dimensional scenarios. This wide variety of combinations is in one system with a user-friendly interface that allows potential users, such as environmental engineers, animators, designers, etc, to easily explore it. Dam break scenarios, flood simulations, oil spilling disasters and others can be simulated with decent precision due to the numerical improvements and in smaller amount of time, thanks to the source code parallelization. These features help to achieve procedure or system requirements in which precision and or time are critical factors.

Objectively, this research aims at enhancing the numerical precision of the MPS, as well as expanding the universe of simulation possibilities of different types of fluids and enable its practical utilization though CPU and GPU optimization and parallelization. More importantly, the system should present an adequate level of abstraction to make its usage possible by people that do not necessarily have programming knowledge. This is achieved by the development of a graphical user interface (GUI) that gives the user almost full control to the tool, since the desired test must be previously generated. Finally, the possibilities provided by the resulting system are many, and, considering that the code is written in a way that modification and addition to new models are easily coupled, the system potential is promising. The resulting system can already facilitate the solution of engineering problems regarding natural and environmental disasters in coastal and flooded (or floodable) areas, also, it can be used in CG, and, for some cases, even real-time applications.

1.2 CONTRIBUTIONS

A substantial survey in the literature is fulfilled related to the MPS method with the purpose of understand and expose the strong and weak points of the MPS technique, what the community has been proposing in relation to this technique and for which purposes this method can be applied. Mathematical formulations and theoretical explanations of MPS variations are also presented.

Based on the work of Gotoh (GOTOH, 2013), a stable free-surface fully incompressible fluid simulation method was implemented to run in CPU and, subsequently, an unprecedented parallelized version of this method was developed to run in CPU, through the use of OpenMP, and in GPU, through the use of the CUDA. Then, the method was extended to three-dimensions in both versions, so the user can choose whether the whole scene can be fully simulated in 2D or 3D.

In order to achieve a system which satisfies the needs of different potential users, the advances proposed in the work of Shakibaeinia and Jin (SHAKIBAEINIA; JIN, 2012) were incorporated. In this study a weakly compressible approach was adopted in the MPS method to gain computational efficiency in exchange for some numerical precision in the pressure calculation. A lean multiphase scheme with multi-viscosity and multi-density models was proposed together with a SPS-LES turbulence model (GOTOH; SHIBAHARA;

SAKAI, 2001; SHAO; GOTOH, 2005) for 2D simulations. As for the neighboring search algorithm, which is generally done in brute force, the authors present a cell grid alternative to diminish the search time, also only for 2D scenarios. Based on this work, the whole implementation was extended to 3D, including the neighborhood cell grid scheme. CPU and GPU parallelizations were also employed.

This master research benefits from works like (KHAYYER; GOTOH, 2012) (GOTOH, 2013) (SHAKIBAEINIA; JIN, 2010) (SHAKIBAEINIA; JIN, 2012) (HORI et al., 2011) (ZHU et al., 2011) (FERNANDES, 2013) to come up with an optimized (through parallelization) fluid simulation system where users possess control of its parameters and options with the need to have very little or any programming knowledge. The possible options include:

1. Different fluid behaviors such as two levels of compressibility, different levels of viscosity (setting the fluid kinematic viscosity), turbulent flows and multiphase interaction;

2. Switch on improvements such as better momentum conservation, more precise discretization of differential operators and less spurious pressure oscillations;

3. Choosing between different parallelized implementations such as CPU code using OpenMP, GPU code using CUDA or even CPU code running completely sequentially;

4. A few pre-constructed 2D and 3D case scenarios specifically built to test each one of the mentioned features;

5. A user-friendly interface in which all the features above are integrated and can be easily configured.

The above mentioned models and improvements are integrated in one single Microsoft Visual Studio solution.

1.2.1 Publications

The current master research produced six papers and a book chapter:

- (SILVA et al., 2018): a full paper accepted to be published to a journal of Qualis B3 about the acceleration of the improved MPS method and a large review on works in the field along with different methods and adaptations of the method;

- (SILVA et al., 2017): a full paper published in a conference of Qualis B2 proposing a numerically enhanced and GPU-accelerated through CUDA MPS method by extending the work of (GOTOH, 2013);

- (SILVA et al., 2015): a full paper published in a conference evaluating qualitatively and numerically the SPH method in various test cases common in the literature;
- (ALMEIDA et al., 2016): a book chapter about meshless methods used in computer fluid dynamics written and published in the book "Applied Topics in Marine Hydrodynamics" (ASSI et al., 2016b) which was developed in partnership with Universidade de São Paulo (USP);
- (BRITO et al., 2018): a full paper submitted to a journal of Qualis A2 proposing a rendering solution based on ray tracer for large scale fluid simulation;
- (BRITO et al., 2017): a full paper proposing a multiphase SPH formulation by extending the work of Vieira-e-Silva et al. (SILVA et al., 2015) and also proposing a shader based render solution for this kind of simulation to be published in a conference of Qualis B2;
- (BRITO et al., 2017): a full paper about the acceleration of viscoelastic SPH using graphics processing unit using CUDA being able to simulate a large number of particles, up to 1 million, published in a conference of Qualis B2.

To have an even better experience while reading this dissertation, a possible reading order of some of those works is to start by the book chapter (ALMEIDA et al., 2016) to obtain an improved notion of the big picture of meshless methods for fluid simulation. Afterwards, (SILVA et al., 2015) can show how changing some of the formulations of a meshless method can enhance the precision results and, finally, (SILVA et al., 2017; SILVA et al., 2018) will deepen the discussion on the MPS method with topics such as possible applications, corrected differential operators to improve precision and ways to accelerate the simulation execution time.

1.3 OVERVIEW OF THIS DISSERTATION

In the second chapter of this dissertation there is a background context on the area presenting the state of the art and related works. Next, there is a chapter explaining the technique and presenting a set of variations, improvements and applications of it, in which most of them have been implemented in this work. Then, a chapter regarding acceleration structures/strategies and ways of visualizing graphically the generated simulations are presented. Afterwards, the main results are showcased, presenting each test case scenario utilized and its variations, and the purpose of each. Also in the test cases chapter, a discussion is performed through a comparison with results found in the literature. Subsequently, functions time consumption (absolute and relative durations) of CPU and GPU versions, memory usage, as well as the speedups provided by the parallelized

CPU and GPU versions and the rate of generated frames are presented. Lastly, the final remarks are discussed as well as lessons learned and future works suggestions.

2 STATE OF THE ART

Through the years, disasters involving natural phenomena have triggered several researches in many different areas on how to avoid them. Fluid simulation focusing on liquids is one of these areas. To simulate liquids correctly, the fluid flow should be incompressible or weakly compressible, which guarantees that the fluid density oscillations are kept to a minimum. One of the reasons of simulating liquids in general has been on how to solve these kinds of problems. The MPS method has also been providing great assistance in that field, since it was intentionally created for simulating incompressible flows. The work of Chen et al. is a great reference of this area (CHEN et al., 2013). (ALMEIDA et al., 2016) is an interesting introduction on the subject since it provides a thorough explanation on the MPS as well as possible variations of the method and some implementation details.

2.1 MPS AND ITS VARIATIONS

The MPS method and variations of it has already been used for various purposes and in various fields, such as nuclear engineering phenomena applied to molten core solidification behavior in nuclear power plant accidents and others (KAWAHARA; OKA, 2012; SUN et al., 2012; SHIBATA; KOSHIZUKA; OKA, 2004). Another example is chemical engineering phenomena applied to eutectic reactions, as well as multiphase fluid simulation (CHEN et al., 2011; MUSTARI et al., 2015).

As already has been stated, the MPS method was introduced focusing on the modeling of the behavior of incompressible fluids (KOSHIZUKA; OKA, 1996). Other subsequent works apply the method to certain areas of research, such as coastal and mechanical engineering, among others. In 1998, Koshizuka et al. (KOSHIZUKA; NOBE; OKA, 1998) applied the method to wave breaking in a beach. The authors, in this same work presented an optimization in the neighborhood calculation (from $O(n^2)$ to $O(n^{1.5})$). The previous way (naive) provoked a higher computational load, since the algorithm required that each particle position had to be checked with all the others in the system in order to know which ones were its neighbors. Unfortunately, similarly to the other meshfree methods, the MPS technique suffers from instability problems. Some of these issues are related to numerical errors at the boundaries, i.e., at free-surfaces or when interacting with solid boundaries. There are works that describe why those instability problems arise from the MPS method (KONDO et al., 2008) (LEE et al., 2011).

In attempts to overcome these issues, some authors changed the method in order to improve it; one of the biggest issues with the MPS method is the spurious pressure oscillation. Various works already tried successfully to diminish this. In the work of Kondo

et al. (KONDO et al., 2008) an artificial pressure is adopted to stop gradual density change (one of the conditions to express incompressibility). The stabilization process consists in eliminate negative pressure after solving the PPE, setting the negative pressures to zero. This problem is due to the particle number densities near the surface being small, which causes the particles' pressure to be negative, thus causing instability in the system. With the scheme proposed, the authors claimed to obtain smoother pressure variations using a dam break test case for the analysis.

Ataie-Ashtiani and Farhadi (ATAIE-ASHTIANI; FARHADI, 2006) used a meshless numerical approach to solve Euler's equation, which is the governing equation of the irrotational flow of ideal fluids. Since the time integration of the equations of inviscid flow (mass and momentum conservation) presents difficulties when dealing with incompressible, or nearly incompressible fluids, a fractional step method, which consists of splitting each time step in two, was proposed in order to facilitate solving the inviscid flow equations. Regarding the MPS method stability, various kernel functions were considered and applied to the method, and, as a result of this study, the most suitable kernel function was employed so that the method could increase its stability. The authors concluded that the developed method is quite useful for solving problems with irregular free-surface in hydraulic and coastal engineering when an accurate prediction of free water surface is required.

Lee et al. (LEE et al., 2011) stated that the MPS method, when it was initially proposed, had several defects including non-optimal source term of the PPE, gradient and collision models, and search of free-surface particles, which led to less-accurate fluid motions. In that sense, the authors proposed step-by-step improvements in the processes referred above, originating what they called the PNU-MPS method. After analyzing the improvements using the dam break problem and the problem of liquid sloshing inside a rectangular tank, the authors concluded that the numerical results for violent free-surface motions and impact pressures are in good agreement with their respective experimental data.

Duan and Chen (DUAN; CHEN, 2013) discussed the effects of setting up time step and space step on the stability and accuracy of the viscosity term in the MPS method, which is noted to be a very important property of fluids but not quite easy to simulate. In that work, using the MPS method, two conditions for the setup of time step and initial particle distance in a viscous shear flow simulation method are prescribed to be used specially for simulation flows where viscous forces are dominant. The authors concluded that the stability condition of the viscous term can provide a stable simulation. As for the accuracy condition of the viscous term, it is capable of producing the most accurate simulation for steady laminar flow, and can also provide a realistic and accurate simulation of the molecular viscosity term for unsteady turbulent flow at the expense of a high computational cost though.

A set of papers by Khayyer and Gotoh presents valuable insights and improvements

to this problem. Most of them proposed corrected differential operator models (Laplacian and gradient). In one of their first attempts they proposed a Corrected MPS (CMPS) method (KHAYYER; GOTOH, 2008) for the accurate tracking of water surface in breaking waves. Modifications and corrections in gradient operator model used in the standard MPS method are made with the goal to achieve momentum conservation in the calculations of viscous incompressible free-surface flow.

Then, in 2009, Khayyer and Gotoh (KHAYYER; GOTOH, 2009) proposed new modifications to the MPS method in order to diminish spurious pressure fluctuations. The authors introduced a new formulation of the source term of the PPE, which was referred to a Higher order Source term (HS), thus creating the CMPS-HS method after combining this modification with their previous work. Another modification was allowing slight compressibility to the method, that being, adding part of an equation of state to the right hand side of the PPE. The compressible term in the equation would have a stabilizing effect on the particle's pressure calculation. It was shown that the proposed methods are applicable for an approximate estimation of wave impact pressure on a coastal structure.

In 2010, Khayyer and Gotoh (KHAYYER; GOTOH, 2010) focused on the Laplacian model used in the MPS method. They noticed that to further refine and stabilize the pressure calculation, a Higher order Laplacian model (HL) for discretization of the Laplacian operator should be derived. This model was applied in both Laplacian of pressure and the one corresponding to the viscous forces. By merging this new model with previous modifications proposed by the same authors, the CMPS-HS-HL was originated. The authors remarked that, although the improvements enhanced pressure calculations, the numerical results still presented some unphysical numerical oscillation during tests.

After that, in 2011, following the conclusion in their previous work, Khayyer and Gotoh (KHAYYER; GOTOH, 2011) presented two new modifications in order to resolve the shortcomings that were present in the method proposed in their previous work. The first improvement deals with unphysical numerical oscillation caused by the source term in the PPE, so, extra terms were added to it, which are referred by the authors as Error Compensating parts in the Source term of the PPE (ECS) and, by combining with previous works, the CMPS-HS-HL-ECS was conceived. The second change is meant to deal with situations with tensile instability (MONAGHAN, 2000). It consists of a corrective matrix inserted in the pressure gradient calculations to achieve a more accurate approximation of the differential operator in question.

Some of these variations are going to be detailed, exposing its calculations and nomenclatures, further in the section about the MPS technique (chapter 3), especially the ones that were incorporated in the system proposed in this dissertation.

2.2 RELATED WORKS

Since the MPS is fully meshless, the particles are not connected explicitly by any edge, therefore, it is possible to optimize some computational aspects of the simulation, such as by parallelization, by cluster technology or General Purpose GPU (GPGPU) techniques (SILVA et al., 2018).

Tsukamoto (TSUKAMOTO; NISHIMOTO; ASANUMA, 2005) used shared memory parallelization as a way to accelerate the MPS method. His goal was to simulate floating bodies in highly nonlinear waves and he achieved significant performance gains compared with the sequential version of the simulation.

Ikari and Gotoh (IKARI; GOTOH, 2008) compared two problem decomposition methods, one based on particles decomposition and the other on a domain decomposition. They verified that domain decomposition, in most cases, presents a smaller runtime to finish the calculations.

Gotoh et al.(GOTOH et al., 2009) developed a MPS version to be executed in parallel, combining domain decomposition techniques with dynamic boundaries, periodically recalculating based on the center of mass of each sub-domain to enhance load balancing in the processors and also a process of preconditioner matrix restructuring for accomplishing the forward/backward process of the Conjugate Gradient in parallel. The authors concluded that the proposed method successfully simulated the studied models, but the parallelized model still needed further refinement, that being in precision and computational efficiency. This could be achieved through the development of more accurate and consistent numerical models of differential operators, such as time integration.

Iribe et al. (IRIBE; FUJISAWA; KOSHIZUKA, 2010) presented simulation results of the parallelized MPS for a PC cluster. The authors identified that the bottleneck of the iterative solver parallelization in shared memory is cost of the communicating between sub-domains. To minimize this communication, a sophisticated particle renumbering process based in packages and in a communication list was used. With these techniques, they were able to accelerate the communication process. A 237-hour simulation of a tsunami, pictured in Figure 1, with six million particles was generated. The authors concluded that the reordering process proposed can be used to elaborate an efficient scheme of unidimensional decomposition process.

Shakibaeinia and his colleagues produced a set of papers presenting an alternative form of the MPS method, where simpler and effective formulations and fluid behaviors/flows models are incorporated to the method in order to benefit from its numerically preciser formulations without the main efficiency drawback, which is the solution of the Poisson pressure equation.

Shakibaeinia and Jin (SHAKIBAEINIA; JIN, 2010) developed a method based on the MPS interaction model in which a weakly compressible (WC) model replaces the fully incompressible (FI) one. This was motivated by the fact that the Poisson Pressure Equation

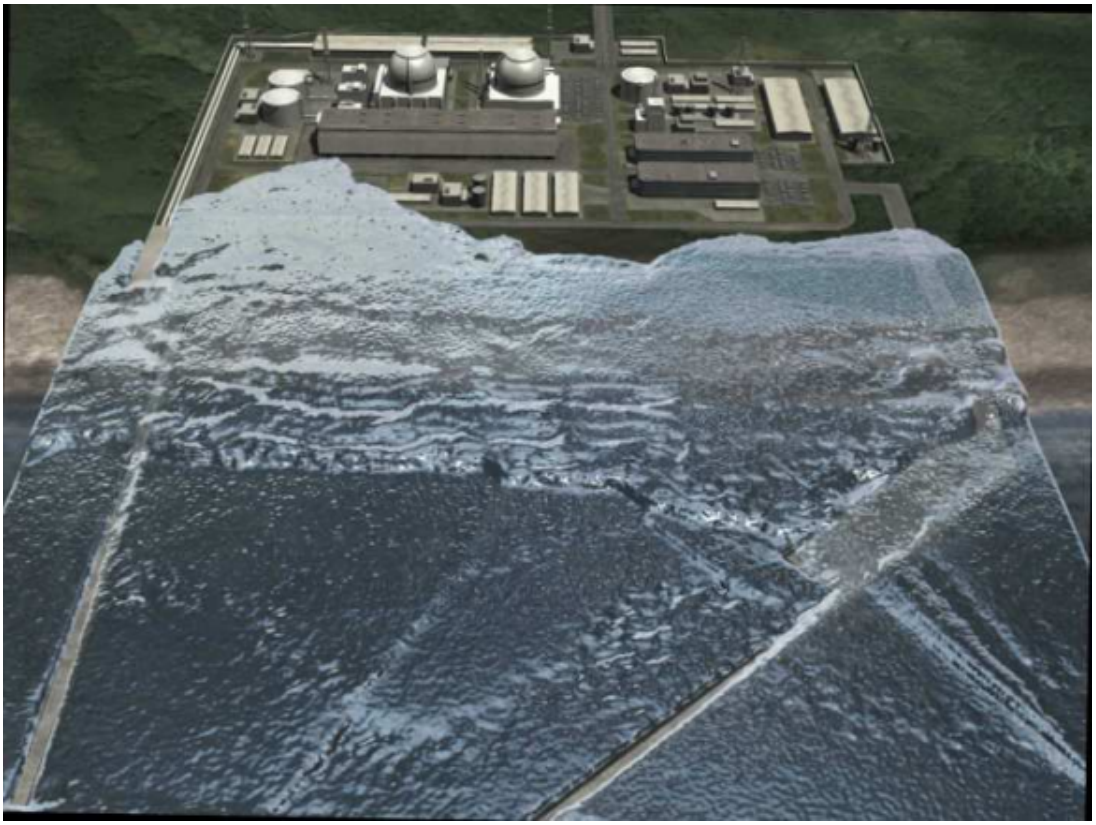


Figure 1 – Tsunami simulation (IRIBE; FUJISAWA; KOSHIZUKA, 2010)

solution is one of the most time-consuming steps of the MPS algorithm. After combining different kernel functions to the method, different numerical precision and computational efficiency are achieved, therefore, an investigation is performed in that sense. The authors claim that the proposed model is able to simulate open-boundary free surface flow in cases even where large deformations and fragmentation are present. There is a clear interest in this work to decrease the computational time of the method, which is an acknowledged issue of it. As it will be shown in this section, there are works that also try to accelerate it by other means.

Shakibaeinia and Jin (SHAKIBAEINIA; JIN, 2012) extended their previous work (SHAKIBAEINIA; JIN, 2010) by proposing a straightforward model of a immiscible multiphase system where, through the WC-MPS, a single set of equations is solved for all the phases. They also employ and investigate different methods for the viscosity model. In order to address turbulence issues in wave dynamics the Large Eddy simulation (LES) concept is employed to formulate a sub-particle scale (SPS) turbulence model. The main test case used for testing the proposed multiphase system was the Poiseuille flow, which showed that the harmonic mean of fluid viscosities for interaction of particles with different property values gives a velocity profile compatible with the analytical solution. Additionally, other tests showed that the model can successfully predict the basic hydrodynamics instabilities and their related features in the high and low density rations.

Hori et al. (HORI et al., 2011) developed a GPU-accelerated version of a MPS code

using NVIDIA's CUDA. The authors focused on the search of neighboring particles and the iterative solution of the linear system generated by the PPE, which generates a large computational load. The optimization of the search for neighboring particles is achieved through a cell grid, in which each particle is stored in a specific cell according to the particle's position. In order to compare accuracy and performance between the CPU and GPU-based codes, 2-dimensional calculations of an elliptical drop evolution and a dam break flow have been carried out. Finally, the reported speedup achieved in that work is about 3 to 7 times.

Zhu et al. (ZHU et al., 2011) developed a GPU-based MPS model using CUDA. To find the neighbors for a specific particle i a similar approach to Hori et al. (HORI et al., 2011) was used, where background grids are employed in order to reduce significantly memory access, taking only $O(kNP)$ times. The authors built four different test cases to evaluate the GPU program optimization, all based on the dam break scenario. To solve the PPE, the Bi-Conjugate Gradient method (BiCG) is used and it is shown that the percentage of time used for solving the pressure equation decreases from 66% to 40% as the total number of particles raises. The authors concluded through a numerical analysis that the models based on CPU and GPU have the same precision and, through a performance comparison, a speedup range from 9 to 26 times can be obtained with the MPS-GPU in contrast to the MPS-CPU, depending on total particle number.

In Fernandes's PhD thesis (FERNANDES, 2013), he developed a computational framework of hybrid parallelization of the MPS method. An altered version of the pressure formulation was used, where the stability is higher at the cost of introducing a small compressibility to the method, something unacceptable in systems with rigorous incompressibility or numerical precision/accuracy requirements. The author concluded that his work contributed to the MPS method consolidation as a practical tool to investigate complex engineering problems, since the method has its applicability extended to scenarios with millions of particles, and could be used, for instance, in the influence of ship movements in waves, phenomena involving fragmentation and dealing with large deformations.

In (TANIGUCHI; SATO; CHENG, 2014), the authors focus on reducing the MPS method runtime by replacing the PPE with an equation of state, as done by (SHAKIBAEINIA; JIN, 2010), which is normally the method's performance bottleneck, and then accelerate it by exploring its parallelization potential through a multi-core CPU, single-node GPU and a multi-node GPU cluster environment. To completely take advantage from the available hardware, two levels of parallelism were implemented: the first one required a simulation domain subdivision among cluster nodes using Message Passing Interface (MPI) for the communications between those sub-domains, and the second, required a in-depth study of the sequential code in order to better port it into a 3D parallel code utilizing the GPU devices. For a 3D dam break scenario with $700,000$ particles the OpenMP solution could reach 5.3 times speedup, while the single-node GPU could reach 14.5 times speedup com-

pared to a single-threaded CPU execution. As for the multi-node GPU with 9 processes, it is able to perform approximately 5.5 times faster than the single-node GPU. The authors claim that the proposed algorithm allows large weakly compressible MPS simulations in distributed memory systems with reduced communication overhead. Also, the lack of a PPE to solve enables higher speedup rates since it is a costly process even parallelized.

Differently from the works here presented, this masters research does not only focus on parallelizing using GPU the standard MPS method but also all the improvements (CMPS-HS-HL-ECS) and models (weak compressibility, turbulence, multiphase and viscoplasticity) that were attached to it as well as benefiting from the CPU multiple cores. This large combination of features cannot be found anywhere in the literature. In addition, the two versions of parallelization are able to achieve relevant speedups (OpenMP and CUDA) of the extensive variations of the MPS method here developed. Exemplifying, one set up of features can ensure a stabler and more accurate method, while another may aim entirely in a higher performance. Equally important, in the fully incompressible set up the characteristics of the PPE are not changed, therefore, maintaining this property completely, if desired.

3 THE MOVING PARTICLE SEMI-IMPLICIT METHOD

The MPS method is detailed in this chapter by showing its governing equations, discretized differential operations, as well as a set of variations, approaches and improvements to the basic MPS and a few illustrations of the MPS algorithm and calculations.

3.1 STANDARD METHOD & GOVERNING EQUATIONS

As described in Koshizuka and Oka (KOSHIZUKA; OKA, 1996), this method models the fluid as an assembly of interacting particles, in which their motion is determined through the interaction with neighboring particles and according to the governing equations of fluid motion. To describe the motion of a viscous fluid flow, there is the continuity equation and Navier-Stokes equation as follows in Equation 3.1 and Equation 3.2, respectively.

$$\frac{1}{\rho}\frac{D\rho}{Dt} + \nabla \cdot \mathbf{u} = 0 \tag{3.1}$$

$$\frac{\partial \mathbf{u}}{\partial t} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} + \frac{1}{\rho}\nabla \cdot \vec{\tau} + F_{ext} \tag{3.2}$$

where $\mathbf{u}$ is the fluid velocity vector, t is the time, ρ is the fluid density, p is the pressure, ν is the laminar kinematic viscosity, $\boldsymbol{\tau}$ is the sub-particle scale (SPS) or turbulence contributed by unresolved small motions (detailed in subsection 3.2.3) and F_{ext} represent external forces like gravity. To adapt these equations so that a fluid can be represented by discrete elements, some of these physical quantities become particles attributes; so $\mathbf{u}$ becomes the velocity vector of a particle, ρ now stands for the density of the particle and p, the pressure of a particle. The left hand side of the continuity equation (Equation 3.1) is represented, in the case of incompressible flow, by a simple volume continuity equation, as presented in Equation 3.3 (GOTOH, 2013):

$$\nabla \cdot \mathbf{u} = 0 \tag{3.3}$$

A particle interacts with its neighbors through a kernel function $w(r)$, r being the distance between two particles. The most common form of kernel function employed in MPS, and used for the implementation in this work, is in Equation 3.4:

$$w\left(|\mathbf{r}_j - \mathbf{r}_i|\right) = \begin{cases} \frac{r_e}{|\mathbf{r}_j - \mathbf{r}_i|} - 1 & , \quad 0 \leq r < r_e \\ 0, & r_e \leq r \end{cases} \tag{3.4}$$

where r_e is the radius of the interaction area and $\mathbf{r}_i$ and $\mathbf{r}_j$ are the positions of particles i and j, respectively. Clearly, a larger kernel size implies in an interaction with more particles, as seen in Figure 2.

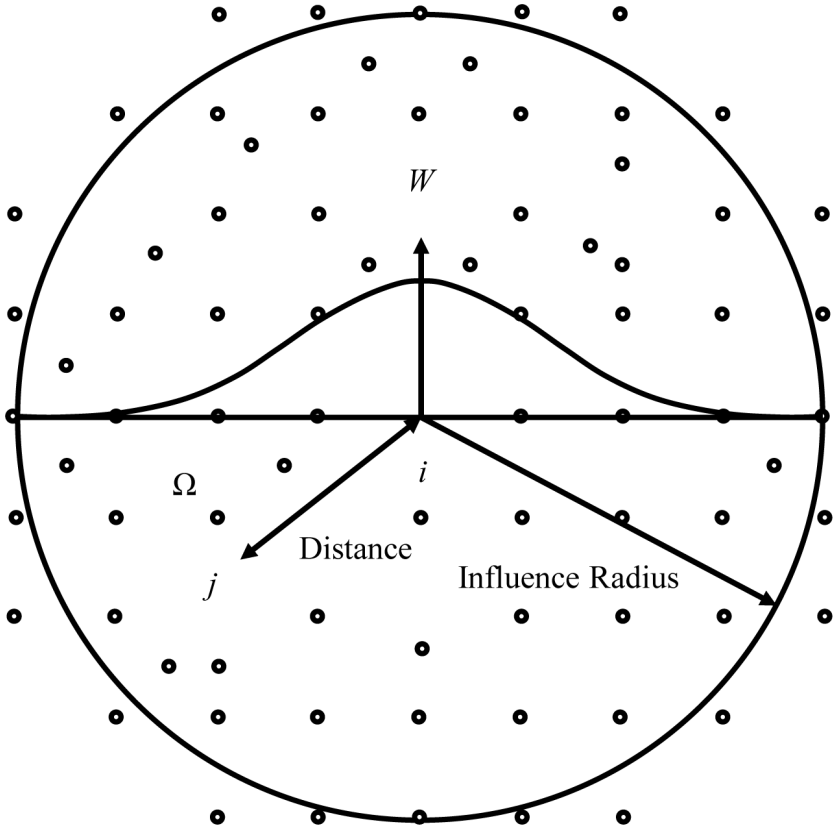


Figure 2 – Influence radius of a particle in a two-dimensional problem

There are also other types of kernel functions and a detailed analysis on the subject can be seen in the work of Ataie-Ashtiani (ATAIE-ASHTIANI; FARHADI, 2006).

To find all the neighboring particles j of each particle i the all-pair search algorithm is used. In this algorithm, the distance between each particle of the simulation is checked to see whether they are in the target's radius of influence and thus determine its neighbors.

The particle number density n_i at the particle's position $\mathbf{r}_i$, which is proportional to the neighbors number of i, is defined in Equation 3.5.

$$n_i = \sum_{j \neq i} w\left(|\mathbf{r}_j - \mathbf{r}_i|\right) \tag{3.5}$$

The continuity equation is satisfied if the particle number density remains constant, and this constant value is denoted by n_0. As stated before, in the original MPS method the derivative of a kernel is not calculated, instead, the gradient or Laplacian are obtained by local weighted averaging of these operators calculated between a pair of particle i and a neighbor, particle j.

The gradient model formulation used in MPS of a physical quantity φ is shown in Equation 3.6.

$$\nabla\varphi_i = \frac{D_S}{n_0} \sum_{j \neq i} \frac{\left(\varphi_j - \varphi_i\right)}{\left|\mathbf{r}_j - \mathbf{r}_i\right|^2} \left(\mathbf{r}_j - \mathbf{r}_i\right) w\left(|\mathbf{r}_j - \mathbf{r}_i|\right) \tag{3.6}$$

In Equation 3.6 and Equation 3.7, D_S is the number of space dimensions in the simulation. This model is ultimately applied to the pressure gradient term. The Laplacian of φ, applied to the pressure and in the viscous stress calculation in this method, is discretized as shown in Equation 3.7.

$$\nabla^2 \varphi_i = \frac{2D_S}{n_0 \lambda} \sum_{j \neq i} \left(\varphi_j - \varphi_i \right) w \left(|\mathbf{r}_j - \mathbf{r}_i| \right) \tag{3.7}$$

where λ is the weighted average of the squared distance between particles i and j (or r_{ij}^2), as can be seen in (KOSHIZUKA; OKA, 1996).

The satisfaction of the continuity equation is indispensable to model incompressibility, so, the fluid density must remain constant. When the particle number density n^* calculated in an intermediate step is not equal to n_0, it is implicitly adjusted to n_0.

Differently from many SPH-based calculations where the equations are solved explicitly, the pressure in MPS is implicitly calculated by solving a PPE. The other terms are approximated explicitly, thus giving the name of the method. To solve the PPE it is necessary a two step prediction-correction process. In the first step there is the explicit integration in time, while, in the second step, the implicit computation of a divergence-free velocity field occurs. The calculation of the intermediate velocity field $\mathbf{u}^*$ is derived from the implicit pressure gradient term as:

$$\mathbf{u}_i^* = \mathbf{u}_i^k + \frac{\Delta t}{\rho_i^*} \nabla p_i^{k+1} \tag{3.8}$$

where k indicates the current time step in the simulation, ρ_i^* is the density calculated at time step k of the particle i and p_i indicates the particle pressure. The velocity and particle densities in Equation 3.8 satisfy the mass conservation law as in Equation 3.9.

$$\frac{1}{\rho} \frac{D\rho}{Dt} + \nabla \cdot \left(\mathbf{u}_i^{k+1} - \mathbf{u}_i^* \right) = 0 \tag{3.9}$$

By representing the derivative of the ρ as $\frac{\rho_0 - \rho_k^*}{\Delta t}$ and substituting ρ for n, it is possible to deduce the PPE (KOSHIZUKA; OKA, 1996):

$$\nabla^2 p_i^{k+1} = -\frac{\rho}{\Delta t^2} \frac{n_i^* - n_0}{n_0} \tag{3.10}$$

The Incomplete Cholesky Conjugate Gradient (ICCG) method is usually employed to solve the linear system (KOSHIZUKA; OKA, 1996; KHAYYER; GOTOH, 2009). By solving the PPE, the velocity in time step $k+1$ ($\mathbf{u}_{k+1}$) can be calculated, and, at last, the particle positions, denoted by r in Equation 3.11, are updated through a simple first-order Euler integration.

$$\mathbf{r}_i^{k+1} = \mathbf{r}_i^k + \mathbf{u}_i^{k+1} \Delta t \tag{3.11}$$

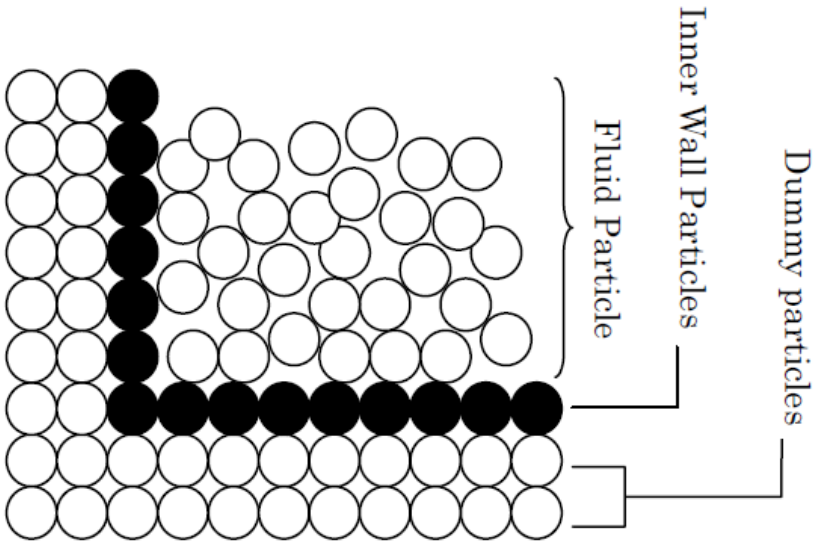


Figure 3 – Dummy boundary scheme

The solid boundaries in standard MPS, as walls and fixed obstacles, are represented by fixed particles with no velocity. Some of these particles, however, are considered to solve the PPE. To tell which will be used for the pressure calculations, it is important to explain that there are two layers of wall particles. One of these layers will be referred to inner wall particles (those that, initially, come into direct contact with the fluid particles) and the other as dummy particles (which complement the solid boundary). Usually, just some lines (often two) of dummy particles are used (KOSHIZUKA; NOBE; OKA, 1998). A model can be seen in Figure 3. The PPE is solved by taking into account the inner wall particles only to repel the fluid from the solid boundaries, while the dummy particles were introduced so that the particle number density at the inner wall particles is not small and that they are not recognized as free-surface.

To identify a free-surface particle, the particle number density of the ith particle just needs to satisfy the condition presented in Equation 3.12 since on the free-surface the particle number density drops abruptly.

$$n_i < \beta n_0 \tag{3.12}$$

The bigger β is, the bigger will be the number of particles recognized as free-surface. Koshizuka and Oka (KOSHIZUKA; OKA, 1996) recommends that it should be set to 0.97. An overview of the MPS algorithm can be seen in Figure 4.

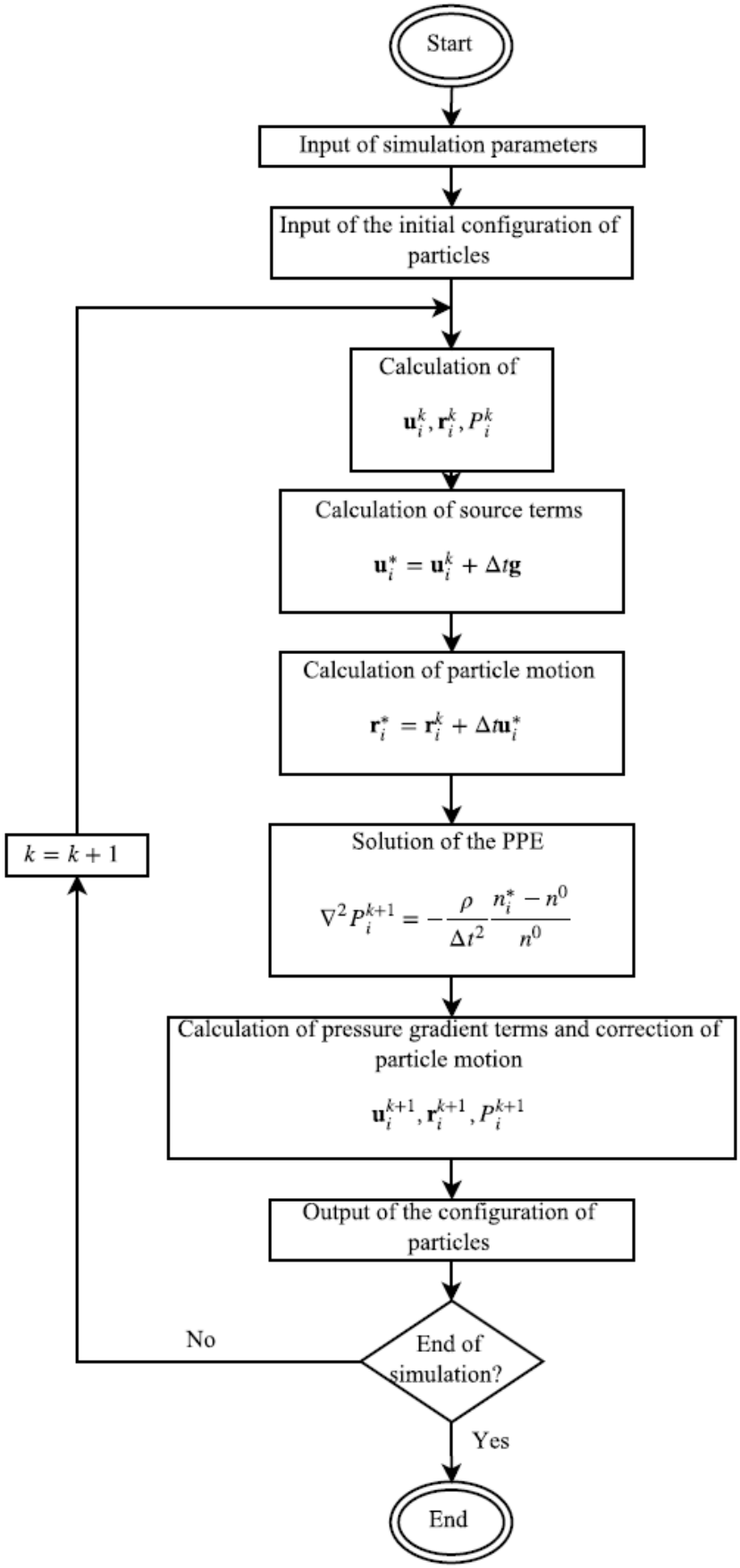


Figure 4 – Algorithm of MPS method

3.2 FLUID BEHAVIORS & FLUID FLOWS

This section presents a set of models of fluid behaviors and fluid flows that were covered in this work. Those models try to improve or even make possible the simulation of fluids in certain situations.

3.2.1 Compressibility

Generally, two approaches can be used when simulating liquids to obtain the pressure of the particles: a weakly compressible (WC) one and a fully incompressible (FI) one, where each one of them has its advantages. In this work both approaches are implemented since the goal is to produce a system that provides a wide variety of features so the users decide the one that best meets their needs.

3.2.1.1 State equation

The WC model prioritizes performance since it can be coded relatively quick and integrated to the MPS method in order to severely diminish computational load in exchange of some numerical precision (TAYEBI; JIN, 2015; TANIGUCHI; SATO; CHENG, 2014). In the work of (SHAKIBAEINIA; JIN, 2010), the traditional incompressible model is replaced by a weakly compressible one on the grounds that assembling and solving the PPE in each step takes a considerable amount of computation time: About two thirds of the computational time in each step for a case where the number of particles in the simulation is in the order of 10^3. In the mentioned work, the PPE is replaced by an explicit relation, more specifically an equation of state described by (BATCHELOR, 1970) and modified by (MONAGHAN, 1994) that is shown below.

$$p_i^{k+1} = \frac{\rho c_0^2}{\gamma}\left(\left(\frac{n_i^*}{n_0}\right)^{\gamma} - 1\right) \tag{3.13}$$

The typical value used for $\gamma = 7$. c_0 is the speed of sound in the reference density. By keeping a small compressibility value, the fluid is treated as a weakly incompressible fluid. This study in fact shows a decrease in process time per time step while the simulation has similar precision to the fully incompressible method. Authors refer to this modified MPS as WC-MPS. The work of (KHAYYER; GOTOH, 2009) proposes another compressible form for the PPE, which is presented in Equation 3.14:

$$\nabla^2 p_i^{k+1} = -\frac{1}{\Delta t^2 c_0^2}\left(p_i^{k+1} - p_i^k\right) + \frac{\rho}{\Delta t}\left(\nabla . \mathbf{u}_i^*\right) \tag{3.14}$$

where c_0 is ,

0the speed of sound. In this work, the compressible term, which is the first term on the right hand side of Equation 3.14, works as a stabilizer for the pressure calculation, soften-

ing part of the noise caused by the second term on the right hand side in Equation 3.14, thus resulting in a somewhat lower fluctuation in the pressure field.

In this work, the approach proposed by (SHAKIBAEINIA; JIN, 2010) was adopted due to its similarity to the weakly compressible approach adopted in the most common and used WC-SPH methods (MONAGHAN, 2005).

3.2.1.2 Poisson pressure equation

As opposed to the weakly compressible approach, the fully incompressible approach tries to obtain the particles' pressure with higher accuracy even though this leads to a high computational load. In this case, it is necessary the solution of a linear system of equations of the type shown in Equation 3.15, yielded by the PPE, shown in Equation 3.10.

$$Ax = b \tag{3.15}$$

where A is a square sparse matrix of size $N \times N$ which N is the total particle number in the simulation, the Right Hand Size (RHS) vector b of size N stores the source terms and x, also of size N, represents the desired pressures of the particles. Equation 3.16 shows a hypothetical square sparse matrix M.

$$M = \begin{bmatrix} a_{11} & a_{12} & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & \ddots & & & & \vdots \\ 0 & a_{32} & a_{33} & a_{34} & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & a_{76} & a_{77} & a_{78} & 0 \\ \vdots & & & & \ddots & a_{87} & a_{88} & a_{89} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & a_{98} & a_{99} \end{bmatrix} \tag{3.16}$$

The assembly of the coefficient matrix A makes use of the discretized Laplacian model, as shown in Equation 3.7 and a simple discretization of the derivative of n, the particle number density, as explained in Equation 3.10. The solution of the PPE in this work is given by the method shown in (KOSHIZUKA; OKA, 1996), the ICCG, as previously mentioned in section 3.1. It consists in submitting the coefficient matrix A into the incomplete Cholesky factorization, which is generally used as preconditioning for iterative numerical methods. A preconditioner is usually related to the diminish of the condition number of a matrix. A function with a high condition number can produce significant changes in the output with small changes in the input, something that can have a huge negative impact on numerical iterative solvers. After preconditioning the matrix A, the conjugate gradient method is applied to solve the linear system of equations iteratively. It is noteworthy that

the incomplete Cholesky decomposition is an intrinsically sequential method, making it not a simple task to parallelize it and get significant speedups (KIM et al., 2016).

3.2.2 Multiphase flow

A model that significantly increases the number of possible applications for the MPS method is the one that supports multi-density fluids interaction, the multiphase flow. Here, an enhanced stabilized MPS method for simulation of multiphase flows characterized by high density ratios is discussed. This method benefits from previous enhancements also suggested by Khayyer & Gotoh and a new one for accurate, consistent modeling of density at the phase interface. One of the challenging issues in simulation of multiphase flows characterized by high density ratios, corresponds to the mathematical discontinuity of density at the phase interface. The simplest way, according to (KHAYYER; GOTOH, 2013), to deal with discontinuity is to evaluate the calculated density at a target particle i based on a simple spatial averaging. So, two schemes referred to the Zeroth-order and First-order accurate Density Smoothening schemes, abbreviated as ZDS and FDS, are shown in Equation 3.17 and Equation 3.18, respectively.

$$\rho_i = \frac{1}{\sum_{j \in I} W_{ij}} \sum_{j \in I} \rho_j W_{ij} \tag{3.17}$$

$$\rho_i = \frac{1}{\sum_{j \in I} W_{ij}} \sum_{j \in I} \left(\rho_j - \frac{\partial \rho_i}{\partial x_{ij}} x_{ij} - \frac{\partial \rho_i}{\partial y_{ij}} y_{ij} \right) W_{ij} \tag{3.18}$$

where I corresponds to the target particle i and all its neighboring particles j, x_{ij} is the distance between particles i and j in the x-axis direction, y_{ij} is the distance between particles i and j in the y-axis direction and W_{ij} represents a different kernel function from the standard MPS kernel called Wendland kernel adopted for all test cases in that study, as can be seen in (WENDLAND, 1995).

(SHAKIBAEINIA; JIN, 2012) proposed a quite straightforward model of a multiphase system based on the MPS method for incompressible multiphase flow, in which the system is treated as a multi-density multi-viscosity fluid. The model is applied to the author's previously documented weakly compressible MPS (SHAKIBAEINIA; JIN, 2010) to solve a single set of equations for all phases. In this model, the density differences of particles of different phases are automatically taken care of, since that, when calculating a particle's velocity, its density appears directly in the equations. The main issue of this approach arises when dealing with the interface between the fluids, which can result in pressure field discontinuities near the interface. The strategy followed by the authors was to use a smoothed value of density $\langle \rho \rangle_i$ instead of the real density of the particles, which was set for each particle before the start of the simulation. Figure 5 shows an image extracted from (SHAKIBAEINIA; JIN, 2012) sketching the smooth density scheme adopted. ρ_I an ρ_{II} are the interacting particles' density values initially set.

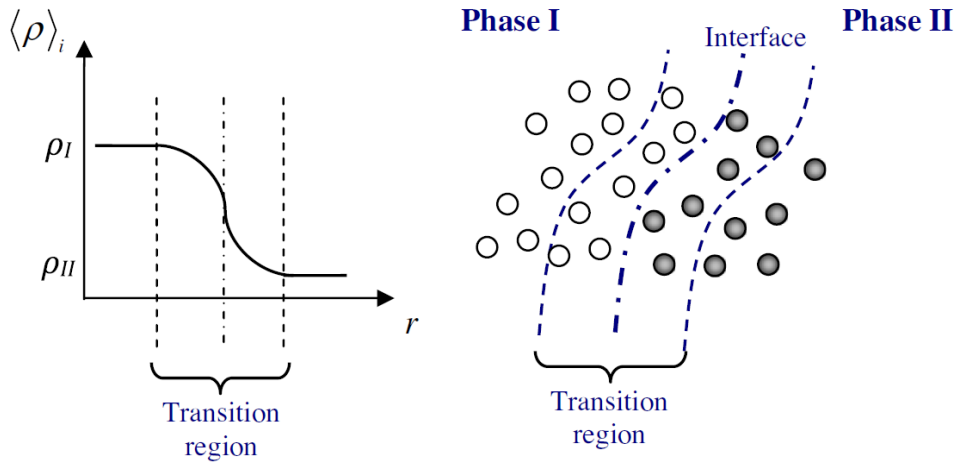


Figure 5 – Multiphase interface scheme extracted from (SHAKIBAEINIA; JIN, 2012)

The density of a certain particle is directly used for calculating its pressure. Equation 3.19 shows the multi-density pressure term in the weakly compressible approach, which was applied in this study.

$$\begin{aligned} \frac{1}{\rho_i}\langle\nabla p\rangle_i &= \frac{d}{n_0}\sum_{j\neq i}\left(\frac{(\rho_j/\rho_i)\alpha_j-\alpha_i}{r_{ij}^2}\mathbf{r}_{ij}w(|r_j-r_i|)\right) \\ \alpha_i &= \frac{c_0^2}{\gamma}((\langle n^*\rangle_i/n_0)^\gamma-1) \end{aligned} \tag{3.19}$$

When there are also differences in the phases' kinematic viscosity, a similar problem happens: viscosity discontinuity near the interfaces. The viscous term is written as in Equation 3.20.

$$\nabla\tau = \nabla(\mu\nabla\cdot\mathbf{u}) = \nabla\mu\nabla\cdot\mathbf{u}+\mu\nabla^2\mathbf{u} \tag{3.20}$$

Despite of the first part of viscous term have low impact in the viscous forces, it is not neglected since its appearance near the interfaces can have some small effects on the momentum exchange in this area. For the second part of the viscous term, which is the most significant to the calculations, the MPS approximation can be written as in Equation 3.21.

$$\langle\mu\nabla^2\mathbf{u}\rangle_i = \frac{2D_S}{n_0\lambda}\sum_{j\neq i}\mu_{ij}\left(\mathbf{u}_j-\mathbf{u}_i\right)w\left(|\mathbf{r}_j-\mathbf{r}_i|\right) \tag{3.21}$$

The μ_{ij} term is the particles' kinematic viscosity values near the interfaces of fluids with different phases and viscosity. Its value is somewhere between μ_i and μ_j and it is determined by the harmonic mean of the two friction factors of interacting fluids, which is the appropriate mean for rates.

Despite the multiphase model proposed by (SHAKIBAEINIA; JIN, 2012) only be applied to the weakly compressible MPS, it was the adopted multiphase model in this work given its relative simplicity combined with its stability and good performance.

3.2.3 Turbulent flow

To calculate the influence of the turbulence term $\boldsymbol{\tau}$, referred to the unresolved small motion term in (SHAO; GOTOH, 2005), the large eddy simulation (LES) mathematical model for turbulence (SMAGORINSKY, 1963) (ROGALLO; MOIN, 1984) was employed. According to the original LES conception, eddies capable of being resolved by the computational grid are allowed to evolve according to the Navier-Stokes equations and a model is employed to represent the turbulence at sub-grid scales (mesh-based methods). Clearly, a sub-particle scale (SPS) model was made necessary for meshless methods. By introducing the turbulence eddy viscosity ν_t, the unresolved SPS turbulence stress τ_{ij} in Equation 3.2 can be written as shown in Equation 3.22.

$$\frac{\tau_{ij}}{\rho} = 2\nu_t S_{ij} - \frac{2}{3}k\delta_{ij} \tag{3.22}$$

where δ_{ij} is Kronecker's operator; and S_{ij} is the strain rate and k is the turbulence kinetic energy, which can be incorporated into the pressure term when solving the momentum equation Equation 3.2. The widely used model by (SMAGORINSKY, 1963) is employed here to formulate the turbulence eddy viscosity as in Equation 3.23.

$$\nu_t = (C_s \Delta X)^2 |\overline{S}| \tag{3.23}$$

where Cs is the Smagorinsky constant (taken as 0.1 in the computations); X is the initial particle spacing and $|\overline{S}|$ is the norm on the local strain rate, which can be calculated from the resolved variables.

3.3 NUMERICAL IMPROVEMENTS

In this section, improvements of the standard MPS that were implemented in this work are described. It is noteworthy that the universe of variations is much larger and the ones that were selected stand between the improvement impact size and implementation cost until, finally, a version that was considered sufficiently stable and physically accurate was achieved. It is also shown other modifications to the standard method which expand the range of applications of the MPS method.

3.3.1 Kernel functions

As discussed before, the motion of each particle depends on the interaction with its neighbors, and this relation is ruled by the kernel function. So, along the years, various kernel functions were suggested for the purpose of achieving better performance and numerical accuracy in the simulation. In (ATAIE-ASHTIANI; FARHADI, 2006) six kernel functions previously proposed are considered and applied to study the simulation behavior

Table 1 – Different kernel functions.

Kernel function formulation	Reference
$w(r) = \begin{cases} e^{-\left(\frac{r}{r_e}\right)^2}, & 0 \leq r \leq r_e \\ 0, & r_e < r \end{cases}$	(BELYTSCHKO et al., 1996b)
$w(r) = \begin{cases} \frac{2}{3} - 4\left(\frac{r}{r_e}\right)^2 + 4\left(\frac{r}{r_e}\right)^3, & 0 \leq r \leq \frac{r_e}{2} \\ \frac{4}{3} - 4\left(\frac{r}{r_e}\right) + 4\left(\frac{r}{r_e}\right)^2 + \\ \quad \frac{4}{3}\left(\frac{r}{r_e}\right)^3, & \frac{r_e}{2} < r \leq r_e \\ 0, & r_e < r \end{cases}$	(BELYTSCHKO et al., 1996b)
$w(r) = \begin{cases} 1 - 6\left(\frac{r}{r_e}\right)^2 + 8\left(\frac{r}{r_e}\right)^3 - \frac{4}{3}\left(\frac{r}{r_e}\right)^4, & 0 \leq r \leq r_e \\ 0, & r_e < r \end{cases}$	(BELYTSCHKO et al., 1996b)
$w(r) = \begin{cases} -2\left(\frac{r}{r_e}\right)^2 + 2, & 0 \leq \frac{r}{r_e} < \frac{1}{2} \\ \left(2\frac{r}{r_e} - 2\right)^2, & \frac{1}{2} \leq \frac{r}{r_e} < 1 \\ 0, & r_e \leq r \end{cases}$	(KOSHIZUKA; OKA, 1996)
$w(r) = \begin{cases} \frac{r_e}{r} - 1, & 0 \leq r < r_e \\ 0, & r_e \leq r \end{cases}$	(KOSHIZUKA; NOBE; OKA, 1998)
$w(r) = \begin{cases} \frac{40}{7\pi r_e^2}\left(1 - 6\left(\frac{r}{r_e}\right)^2 + 6\left(\frac{r}{r_e}\right)^3\right), & 0 \leq r < \frac{r_e}{2} \\ \frac{10}{7\pi r_e^2}\left(2 - 2\frac{r}{r}\right)^3, & \frac{r_e}{2} < r < r_e \\ 0, & r > r_e \end{cases}$	(SHAO; LO, 2003)

and computational performance in order to reveal which one enhances numerical stability best. The kernels considered in this work are presented in Table 1.

This study shows that the kernel function proposed by (SHAO; LO, 2003) was found to improve the most the stability of the MPS method, in way that the collapse of a water column simulation was successful, including the loss of the water momentum to the point where it stands still inside the container, a significant feature amongst other particle methods. For the system developed in this work a set of kernel functions were implemented in order to give the possibility of the user to choose between them, however, it is recommended that the kernel function proposed by (KOSHIZUKA; NOBE; OKA, 1998) is used in the fully incompressible version and the one proposed by (SHAKIBAEINIA; JIN, 2010) is used for the weakly compressible version, as shown in Equation 3.24.

$$w(r) = \begin{cases} (1 - \frac{r}{r_e}) & 0 \leq \frac{r}{r_e} < 1 = 0 \\ 0 & \frac{r}{r_e} \geq 1 \end{cases} \tag{3.24}$$

3.3.2 Momentum conservation

Suzuki et al. (SUZUKI; KOSHIZUKA; OKA, 2007) developed the Hamiltonian MPS (HMPS) in which the momentum and mechanical energy of the system are preserved. However, HMPS carries heavy theory to its calculations making it extremely complicated to implement in comparison to the standard MPS method. A simple way to achieve a consistent conservation of linear momentum is to ensure a better discretization of the gradient model, which is directly connected to the linear momentum conservation.

Equation 3.25 shows the suggested alteration in Equation 3.6 on the pressure gradient formulation by Khayyer and Gotoh (KHAYYER; GOTOH, 2008).

$$\nabla\varphi_i = \frac{D_S}{n_0}\left(\sum_{j\neq i}\frac{\left(\varphi_i+\varphi_j\right)-\left(\hat{\varphi}_i+\hat{\varphi}_j\right)}{\left|\mathbf{r}_j-\mathbf{r}_i\right|^2}\left(\mathbf{r}_j-\mathbf{r}_i\right)w\left(\left|\mathbf{r}_j-\mathbf{r}_i\right|\right)\right) \tag{3.25}$$

$$\hat{\varphi}_i = min_{j\in J}(\varphi_i,\varphi_j), J=\{j : w(|\mathbf{r}_j-\mathbf{r}_i|)\neq 0\} \tag{3.26}$$

When the anti-symmetric Equation 3.25 is applied, linear momentum is exactly conserved. This method is referred to by the authors as Corrected MPS (CMPS).

3.3.3 Pressure calculation

One of the major issues of the MPS method, and consequently widely explored, is the spurious pressure oscillation. Recent works that presented substantial improvements in this area, making few and simple modifications to the method, have been proposed (KHAYYER; GOTOH, 2009; KHAYYER; GOTOH, 2010). The first one is called by the authors as the MPS method with a Higher order Source term (MPS-HS), since it basically presents a new formulation for the calculation of the derivative of the particle number density ($\frac{Dn}{Dt}$). Using this method, the Equation 3.10 is replaced by the Equation 3.27 (KHAYYER; GOTOH, 2012).

$$\nabla^2 p_i^{k+1} = -\frac{\rho}{n_0\Delta t}\left(\sum_{i\neq j}\frac{r_e}{r_{ij}^3}\left(x_{ij}u_{ij}\ +\ y_{ij}v_{ij}+z_{ij}w_{ij}\right)\right)^* \tag{3.27}$$

where r_{ij} is the distance between particles i and j. x_{ij}, y_{ij} and z_{ij} represent the distance between particles i and j in each dimension and u_{ij}, v_{ij} and w_{ij} the velocity difference of particles i and j in each dimension. It is important to note that all the enhancements shown so far can be, and most of them normally are (specially when they are suggested by the same authors), combined in one single method to produce a more robust outcome.

The other improvement to the method's pressure calculation implemented was the proposition of a higher order Laplacian model for both two and three (Equation 3.28)

dimensional simulations (KHAYYER; GOTOH, 2010; KHAYYER; GOTOH, 2012).

$$\nabla^2 \varphi_i = \frac{1}{n_0} \sum_{i \neq j} \left(\frac{2 \varphi_{ij} r_e}{r_{ij}} \right) \tag{3.28}$$

where φ is a generic physical quantity. This new derivation was named by the authors as MPS with a Higher order Laplacian of pressure (MPS-HL).

3.3.4 Numerical stability

Khayyer and Gotoh (KHAYYER; GOTOH, 2011) came up with a PPE's source term with error-compensating parts to enhance even further pressure and velocity field calculations. The error-compensating terms should be measures for instantaneous and accumulative violations of fluid incompressibility. Equation 3.29 shows the suggested terms to be added to the source term of the PPE and Equation 3.30 shows the complete modified PPE.

$$ECS = \left| \left(\frac{n^k - n_0}{n_0} \right) \right| \left[\frac{1}{n_0} \left(\frac{Dn}{Dt} \right)_i^k \right] + \left| \left(\frac{\Delta t}{n_0} \left(\frac{Dn}{Dt} \right)_i^k \right) \right| \left[\frac{1}{\Delta t} \frac{n^k - n_0}{n_0} \right] \tag{3.29}$$

$$\nabla^2 p_i^{k+1} = \frac{\rho}{n_0 \Delta t} \left(\frac{Dn}{Dt} \right)_i^* + ECS \tag{3.30}$$

The combination of the refinements shown so far gives as outcome the Corrected MPS with a Higher order Source term - Higher order Laplacian of pressure - Error Compensating parts in the Source term (CMPS-HS-HL-ECS) method. According to Gotoh (GOTOH, 2013), the said method ensures satisfactory accuracy and stable computation, more specifically, under the absence of tensile stress. A comparison between CMPS-HS-HL-ECS and the standard MPS can be seen in (GOTOH, 2013) which presents a standard test case found in the literature, the breaking waves. This comparison is abridged in Figure 6, where different shades of gray represent different pressure levels.

A set of important acronyms is summarized in Table 2.

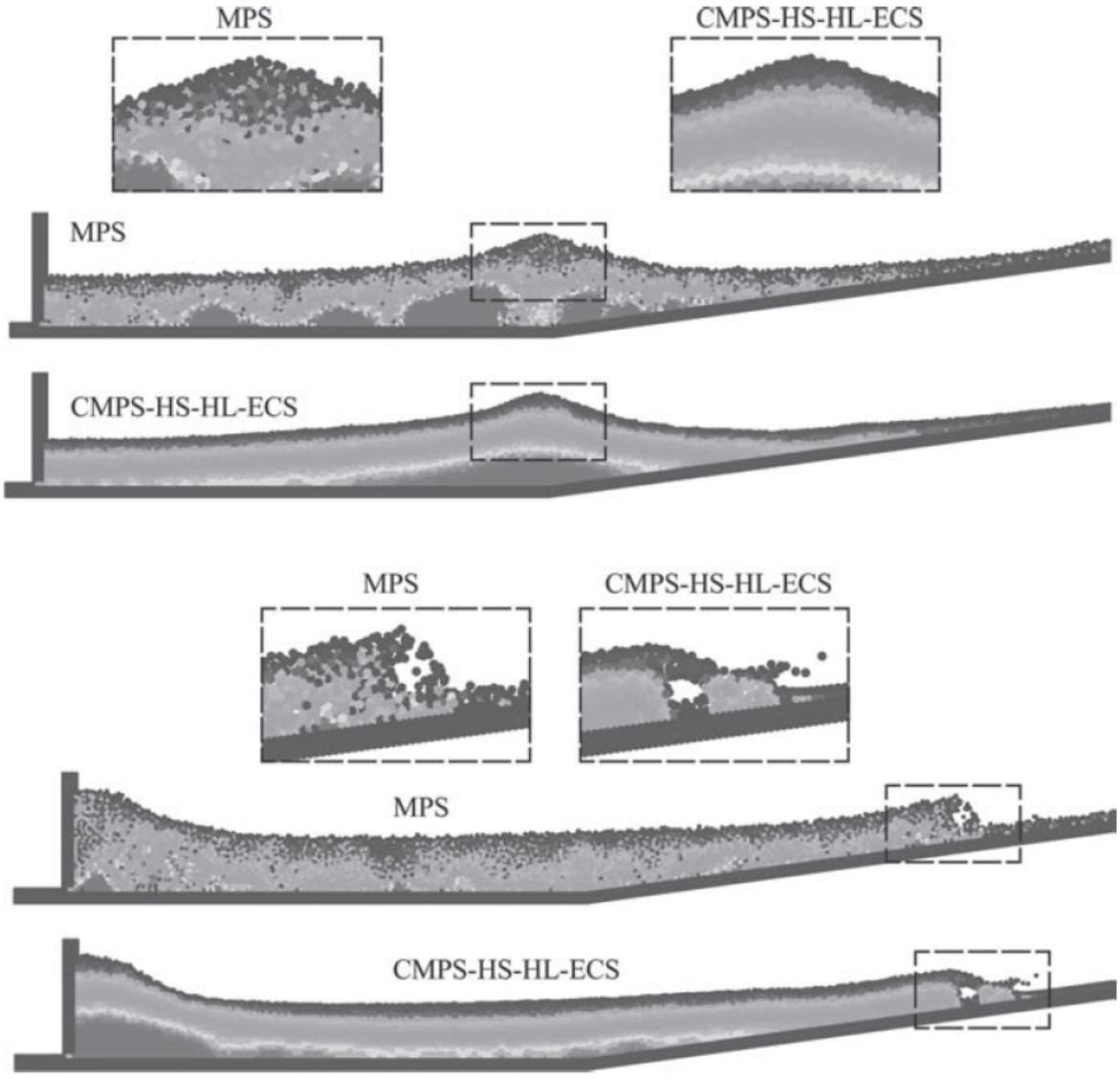


Figure 6 – Comparison between standard MPS and CMPS-HS-HL-ECS through breaking waves test case (GOTOH, 2013)

Table 2 – Description of acronyms.

Acronym	Meaning	Description
MPS	Moving Particle Semi-implicit	Fluid simulation method
SPH	Smoothed Particle Hydrodynamics	Fluid simulation method
PPE	Poisson Pressure Equation	Yields a system of linear equations that needs to be solved to obtain the particles' pressure values
FI	Fully Incompressible	Approach that solves a PPE to obtain the particles' pressure values
WC	Weakly Compressible	Approach that solves a equation of state to obtain the particles' pressure values
CMPS	Corrected MPS	MPS with improved momentum conservation by using a new gradient model
HS	Higher order Source term	Improvement in pressure calculation numerical precision by using a new derivative model for n
HL	Higher order Laplacian model	Improvement in pressure calculation numerical precision by using a new Laplacian model
ECS	Error Compensating parts in the Source term	Improvement in numerical stability by adding terms in the PPE's source term
SPS	Sub-Particle Scale	Turbulence model
LES	Large Eddy Simulation	Numerical technique that describes turbulence

4 SIMULATION & VISUALIZATION IMPLEMENTATION

In order to implement the MPS method and put it into practice through visualization, it is not only necessary to code it using a programming language and environment, but also make sure that the outcome is available in, at least, a feasible amount of time. For that, acceleration structures and application programming interfaces (APIs) for code parallelization were utilized.

Of course, the utility and practicality of the method would diminish drastically if the simulation could not be displayed graphically to seek replicate a real-world fluid simulation. To make it possible, general purpose visualization softwares and rendering solutions have been used. This chapter will detail the implemented structures and APIs, softwares and solutions used in this work.

The integrated development environment (IDE) used to write, run and debug all the code created in this work was the Microsoft Visual Studio 2015, which ran on the Microsoft Windows 10 operating system. The C/C++ programming languages were used to implement the method. The OpenMP and CUDA APIs were used in order to explore the many cores in the CPU and the GPU, hence accelerating the system runtime. Lastly, the Windows Forms graphical class library was explored to build a graphical user interface (GUI) for the system.

4.1 ACCELERATION IMPLEMENTATION

This section addresses the performance improvements employed to the system such as acceleration structures and the explored APIs in order to reduce the computational time of the system as a whole.

4.1.1 Neighboring search algorithms

The search of neighbors of a particle has generally been a computationally expensive process in particle-based methods such as the SPH and MPS (TANIGUCHI; SATO; CHENG, 2014) (HORI et al., 2011). The brute force approach has a complexity of $O(n^2)$ (with n being the total number of particles in the simulation domain) since that, to check whether a particle j is inside the influence radius of a particle i (in other words, check if j is a neighbor of i), one must compute and compare the distances of all the particles in the simulation domain to a particle i in focus. This approach was employed in early versions of the particle-based methods, such as the initial versions of the MPS method (KOSHIZUKA, 1995) (KOSHIZUKA; OKA, 1996).

With that in mind, a couple years after proposing the MPS method, Koshizuka and his colleagues proposed an optimized all-pair search algorithm (KOSHIZUKA; NOBE; OKA,

Listing 4.1 – Optimized all-pair search algorithm

```
(...)

/* Getting position of potential neighbors, particles i and j */
Part_j = (*particles)[j].get_po();
Part_i = (*particles)[i].get_po();

/* Getting distance in x and y-axis between both particles */
x = Part_j.x - Part_i.x;
y = Part_j.y - Part_i.y;

/* If their euclidean distance is less or equal to the radius
    of interaction value, i and j are neighbors */
if(((x*x)+(y*y)) <= r2)
{
    /* Increments neighbors number of each particle */
    neighbors[i][0]++;
    neighbors[j][0]++;

    /* Stores the neighbor index in the array of neighbors
        of the target particles */
    neighbors[i][neighbors[i][0]] = j;
    neighbors[j][neighbors[j][0]] = i;
}


(...)
```

1998). Given the standard all-pair search algorithm explanation above, the optimized version of this algorithm allows that, while a particle j is being registered as a neighbor of the target particle i, inside the same loop iteration the particle i is also set as a neighbor of particle j. This makes the computational complexity of the algorithm go from $O(n^2)$ to $O(n^{1.5})$. Listing 4.1 shows a code snippet of the optimized all-pair search algorithm implementation.

Still, with the optimized all-pair search algorithm, the search of neighbors remained a bottleneck in the method's computation. For the present study, a strategy called "cell-linked list" was adopted (SHAKIBAEINIA; JIN, 2010). It relies on a background Cartesian grid that divides the whole domain in squares, for two-dimensional scenarios, or cubes, for three-dimensional simulations. The squares or cubes are referred to cells and they have sides equal to r_e, which is the influence radius of a particle. In every iteration the particles of the simulation are allocated in a specific cell depending on its position. Thereby, when searching for interacting partners (neighbors) of a particle, it is only necessary to look for them inside the cell of the particle itself and also in adjacent cells, generating, for each particle, a list of particles which remains constant for that entire step. To simplify the understanding, Figure 7 shows a 2D domain divided in square cell in which the algorithm performs the search for neighbors of the particle on focus (in red) only in the cells inside the highlighted square (in gray), which offers a considerable decrease on the complexity of the neighboring search function to $O(n\ log\ n)$. This strategy was adopted in this work.

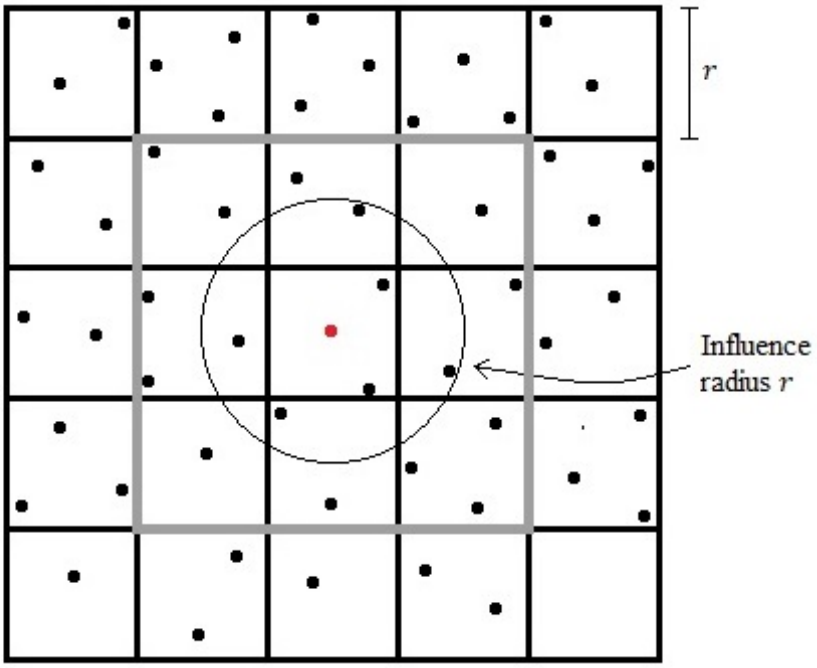


Figure 7 – Cell-linked list scheme

4.1.2 OpenMP

(DAGUM; MENON, 1998) proposed the Open Multi-Processing (OpenMP) API and defined it as a set of compiler directives and callable runtime library routines that extend a limited set of programming languages to express shared-memory parallelism. Today, OpenMP provides a simple and flexible interface an makes possible developing parallel applications for platforms ranging from the standard desktop computer to the supercomputer. For this work, OpenMP was used with the same end as CUDA: to benefit from the multiple processing cores available. In the case of OpenMP, It is benefiting from the multiple cores in the CPU. The machine used in the present study is an Intel® Core™ i7-4790 CPU @ 3.60 GHz (INTEL..., 2013) with 8 GB of installed RAM and a 64-bit operating system (x64) and can run simultaneously up to 8 threads. During the development of the OpenMP version of the system here proposed, very little obstacles were found thanks to OpenMP's straightforwardness and high level of abstraction. Listing 4.2 shows an example where by just adding a single line of code (highlighted in red, in line 2), a function's main loop (responsible for calculation the particle number density of each particle) is now running in parallel.

4.1.3 CUDA

The GPU code was developed based on the fully sequential and the OpenMP versions previously presented. CUDA C/C++ was used to write and optimize even further the system. Each one of the functions, from the calculation of the time step value and the particle number density of the particles, to the assemble of the coefficient matrix and the

Listing 4.2 – C code of the particle number density calculation benefiting from OpenMP

```
2  #pragma omp parallel for schedule (guided)
   for (int I = 1; I <= NUM; I++)
4  {
       double sum = 0.0;
6      for (int l2 = 2; l2 <= neighb[I][1]; l2++)
       {
8          int J = neighb[I][l2];
           (...)
10         /*sum of all kernel function values given the the particle I and each of its
               neighbors J*/
           (...)
12     }
       (...)
14     n[I] = sum;
       (...)
16 }
   return(n);
```

source term was ported to run in CUDA kernels. Inside these kernels, through the use of specific keywords and commands, CUDA allowed the assignment of tasks to various threads inside the GPU, so independent tasks, which would be executed sequentially, can occur simultaneously but now using a larger number of threads, differently from the OpenMP version.

NVIDIA GPUs are organized in grids. As it is pictured in Figure 8, each grid has a number of blocks, which each block contains a certain number of threads and all of these components have two or three dimensions, depending on the GPU compute capability. The number of grids, blocks per grid, threads per block, etc., also varies according to the GPU compute capability. The compute capability of the NVIDIA GeForce GTX 760 used in this work is 3.0, so it allows three dimensions in the components and has 1152 CUDA cores (NVIDIA..., 2013). The maximum number of threads per block is 1024 in the x and y-dimensions, while in the z-dimension this number is lowered to 64. As for the maximum x-dimension of a grid of thread blocks, $2^{31} - 1$ is the corresponding value, while in the y and z-dimensions this number is 65535. This information is important in the development of a CUDA efficient program because, when calling a kernel, the number of blocks and threads that will be used has to be specified as configuration parameters. Inside the CUDA kernel, each thread inside a block and each block has its own index, in all dimensions. That is indispensable to identify and assign a task for each block's threads.

Regarding the implementations, while the process to parallelize a portion of the functions was quite straightforward, some of them had to be adapted in order to be possible to develop a parallelized version of each of them.

One example of a straightforward function parallelization is the particle number density calculation, which in the test case used in this work is represented by n. Listing 4.3 and Listing 4.4 show the CPU and GPU implementation of this calculation, respectively.

It is possible to note that, in lines 4 and 5 of Listing 4.4, the calculation of the index

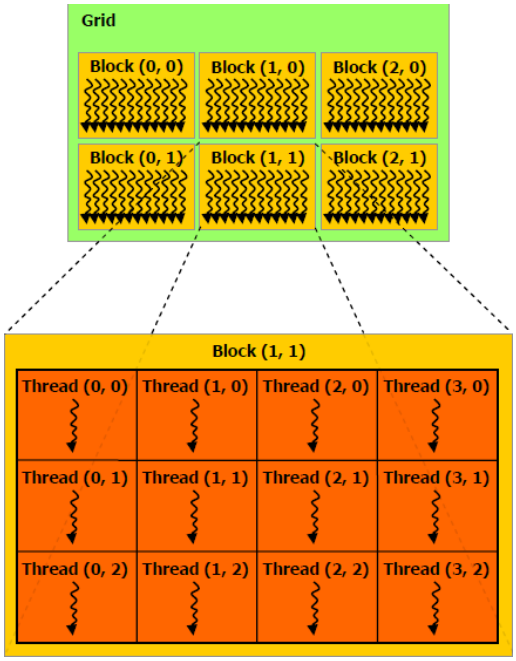


Figure 8 – High-level NVIDIA GPU architecture (NVIDIA, 2007)

Listing 4.3 – CPU code of the particle number density calculation

```
for (int I = 1; I <= NUM; I++)
{
    double sum = 0.0;
    for (int l2 = 2; l2 <= neighb[I][1]; l2++)
    {
        int J = neighb[I][l2];
        (...)
        /*sum of all kernel function values given the the particle I and each of its
            neighbors J*/
        (...)
    }
    (...)
    n[I] = sum;
    (...)
}
return (n);
```

of each particle takes into consideration the case when the total particle number of the simulation exceeds the maximum number of threads in a block, so threads in other blocks are triggered and the thread count proceeds from where it stopped in the previous block.

One example where the code had to be adapted, is in the neighboring particle search algorithm, where cell-linked list approach by (SHAKIBAEINIA; JIN, 2010) had to be extended to 3D cases and subdivided into three different functions to be possible extract the necessary information in each step of the function.

A commonly tricky parallelization step in the fully incompressible version of the MPS is the PPE solution. For this work, the standard ICCG solver used by Koshizuka and his colleagues (KOSHIZUKA; NOBE; OKA, 1998) to solve it was parallelized in the OpenMP version as well as in the CUDA version using raw array formats for the linear system ele-

Listing 4.4 – GPU code of the particle number density calculation

```
1
unsigned int I = offset + (blockDim.x * blockIdx.x + threadIdx.x);
3
double sum = 0.0;
5 for (int l2 = 2; l2 <= neighb[I][1]; l2++)
{
7     int J = neighb[I][l2];
      (...)
9     /*sum of all kernel function values given the the particle I and each of its
          neighbors J*/
      (...)
11 }
(...)
13 n[I] = sum;
(...)
```

Listing 4.5 – PPE assemble GPU code

```
1
unsigned int I = offset + (blockDim.x * blockIdx.x + threadIdx.x);
3
double val = 0.0;
5 if (bcon[I] == 0) {
      poiss[I*NEIGHBORS + 1] = 0.0;
7     for (int l = 2; l <= neigh[I*NEIGHBORS + 1]; l++) {
          int J = neigh[I*NEIGHBORS + l];
9         if (I == J) continue;
          if (bcon[J] == -1) poiss[I*NEIGHBORS + l] = 0.0;
11        else {
              (...)
13            /* Calculating the distance between particles */
              (...)
15            #ifdef MATRIX_OPTMIZATION
              if (dim == 2) val = 3.0 * re;
17            else val = 2.0 * re;
              val = val / (d*d*d) / n0 / Rho;
19            #else
              val = 2.0*dim / lambda*W(d, KTYPE, dim, re) / n0 / Rho;
21            #endif
              poiss[I*NEIGHBORS + l] = -val;
23            poiss[I*NEIGHBORS + 1] += val;
          }
25    }
      poiss[I*NEIGHBORS + 1] += (1.00e-7) / DT / DT;
27 }
```

ments, such as the coefficient matrix and right-hand side source vector. Also to decrease memory usage by data structures, the total number of possible neighbors of a single particle was limited to 300. Thanks to this, it was possible to assemble a coefficient matrix with $N \times 300$ elements with N being the total number of particles, rather than an $N \times N$ matrix, thus, increasing the simulation's total number of particles in the fully incompressible version of the code. Listing 4.5 shows a code snippet from the PPE assemble.

All the functions developed in the CPU version were successfully ported to CUDA C/C++ to run in the GPU.

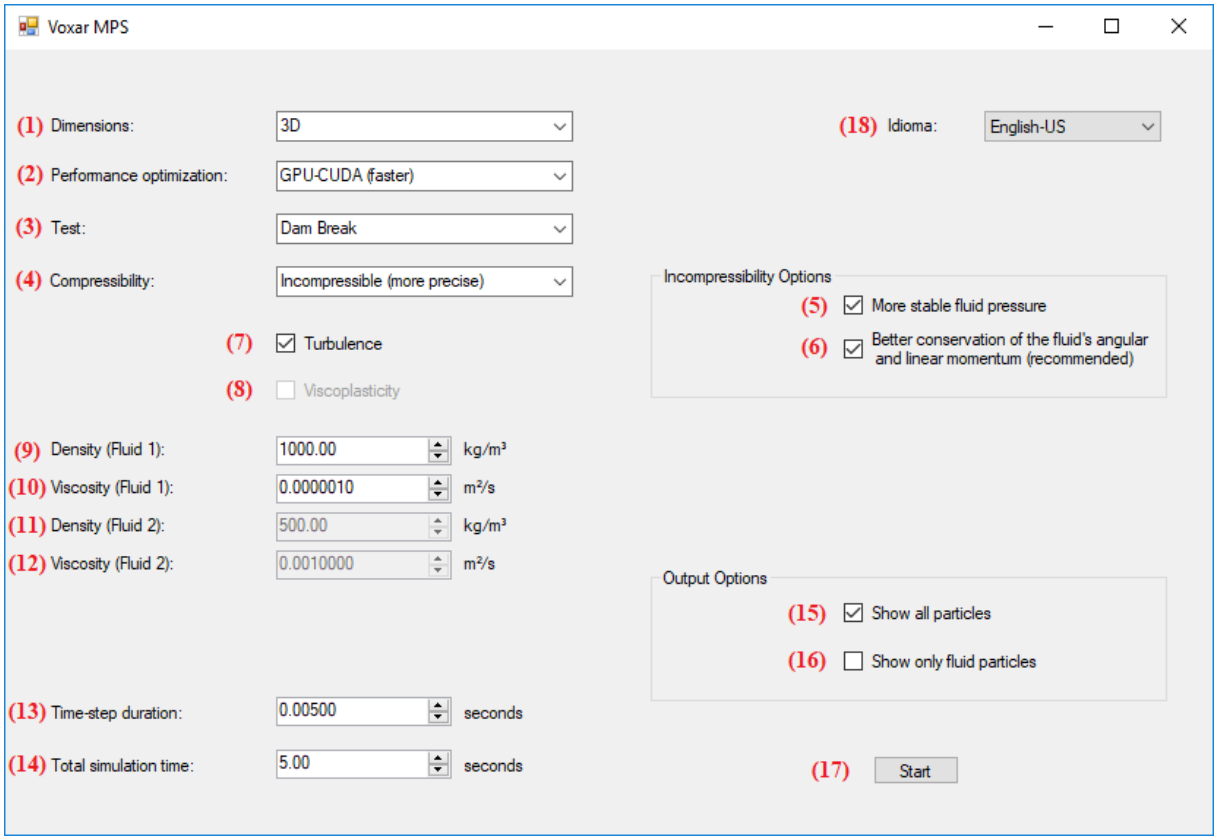


Figure 9 – User interface main screen

4.2 GRAPHICAL USER INTERFACE

As informed in the beginning of this chapter, the Windows Forms graphical class library from the Microsoft .NET framework was explored to build a graphical user interface (GUI) for the system. Figure 9 shows the implemented interface, which facilitates the interaction and usage of the developed system.

In Figure 9, (1) is a combo box from which the user can choose whether the simulation will have two or three dimensions; in (2) the user can choose how the code will be executed: sequentially, parallelized through OpenMP or parallelized through CUDA; in (3) the user will select a previously built simulation scenario; in (4) two types of fluid compressibility can be chosen: weakly compressible or fully incompressible. By selecting the latter, (5) and (6) will be enabled, which are options that, if checked, will further stabilize the pressure calculations of the simulation and better conserve the linear and angular momentum of the fluid (by using the improvements presented in section 3.3); By checking (7), the SPS-LES turbulence model will be employed; (8) will only the enabled if the test scenario chosen allows a multiphase system. If checked, the second fluid in the simulation will present viscoplastic properties; (9) and (11) are, the density values in $[Kg/m^3]$ of the two fluids in the simulation, and (10) and (12) are their viscosity values in $[m^2/s]$; (13) sets the time duration between two steps of the simulation and (14) sets how long it will last, both in seconds; if (15) is checked, a folder will be created in the same directory of the

executable file containing the generated frames showing all the particles present in the simulation and, if (16) is checked, another folder will be created with frames showing only fluid particles; (17) starts the generating the simulation and (18) allows the user to switch between English and Portuguese languages.

4.3 VISUALIZATION

In order to facilitate the simulation analysis and also enable a wider range of applications for the developed system, softwares and rendering solutions are not only helpful but, in some cases even necessary. This section shows the software used in this work as well as some rendering solutions that can potentially be integrated to the system (BRITO et al., 2017; BRITO et al., 2018).

4.3.1 Paraview

At the end of the execution of each time step, the the code outputs a set of information regarding that step, such as position, velocity components, the pressure value and the type of each particle in the simulation that is being generated, in a XML-like file referred to a VTU file (SCHROEDER; LORENSEN; MARTIN, 2004). To visualize the simulation, the ParaView software (AHRENS; GEVECI; LAW, 2005) is used. It is an open-source, multi-platform data analysis and visualization application that enables quick data analysis using qualitative and quantitative techniques.; it reads all the VTU files generated by the program and shows graphically their information. The ParaView interface in shown in Figure 10, where a vtu example of a dam break is loaded.

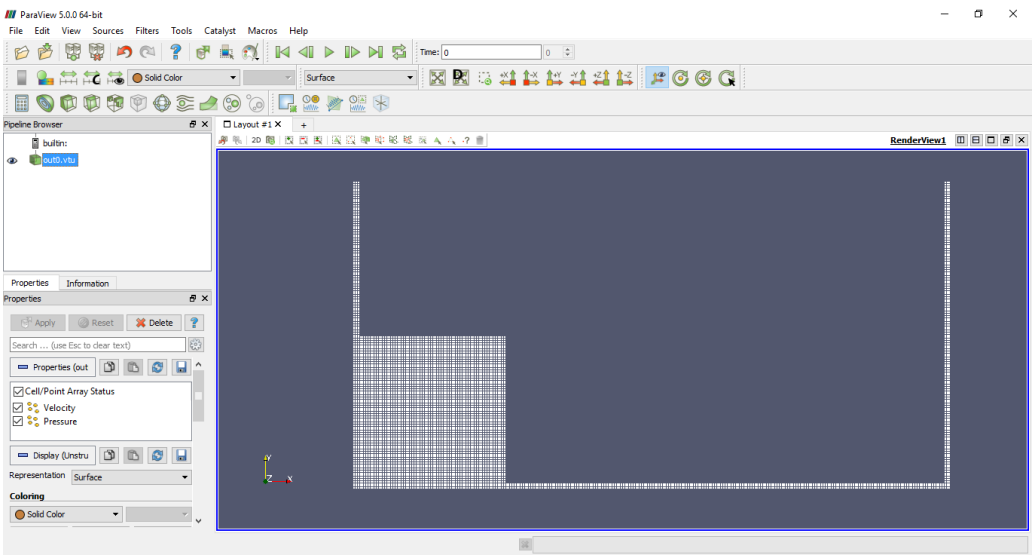


Figure 10 – Paraview software

4.3.2 CG & Rendering

Particle-based methods have been used in many Computer Graphics (CG) applications, such as for simulating granular materials (BELL; YU; MUCHA, 2005), smoke and explosions (SELLE; RASMUSSEN; FEDKIW, 2005), and, of course, for fluid dynamics simulation (MüLLER; CHARYPAR; GROSS, 2003). The SPH method is widely adopted by the community in fluid animation (IHMSEN et al., 2014). Despite of its usage in CAE softwares (Fuji Technical Research Inc., 2013) (Prometech Software, 2014), the MPS method is not commonly seen in the literature applied to CG applications, whether for movies, games or real-time applications. This study attempts to reduce this gap providing a decent method with a good variety of settings and parameter tuning. For instance, it is possible to obtain a configuration where computational performance is a priority by activating GPU optimization and choosing the weakly compressible approach. Hence, larger MPS simulation cases could meet real-time requirements that CG applications request.

By coupling the developed method with rendering solutions like (BRITO et al., 2017) (BRITO et al., 2017), decent CG applications with interactive frame-rates could be achieved.

5 RESULTS

The aim in this chapter is to validate some models and properties of the developed method such as incompressibility, numerical precision, turbulent flow and multiphase interaction. Another important goal here is to assess the performance gain of the parallelized implementations of the weakly compressible and fully incompressible versions of the MPS method and discuss them.

5.1 WATER DROP

The examination of the evolution of an elliptic water drop is one of the most common scenarios to validate incompressibility models of a particle-based fluid simulation method; examples of its use can be found in the literature in the works of (MONAGHAN, 1994) (BONET; LOK, 1999) (SHAKIBAEINIA; JIN, 2010). The usual test consists on a two-dimensional water drop, beginning the simulation in the shape of a circle Figure 11, with a predefined velocity field of $(-100x, 100y)$ m/s so that the its format evolves into an elliptical shape over time. The circle radius is 9×10^{-2} m and the average particle distance is 5×10^{-2} m in both directions, implying in a total of 1257 particles. The influence radius was taken as $r_e = 3l_0$, where l_0 is the average particle distance, and the adopted time step for the simulation was 10^{-5} s. The used method for this test was the turbulent and viscous CMPS-HS-HL-ECS, as well.

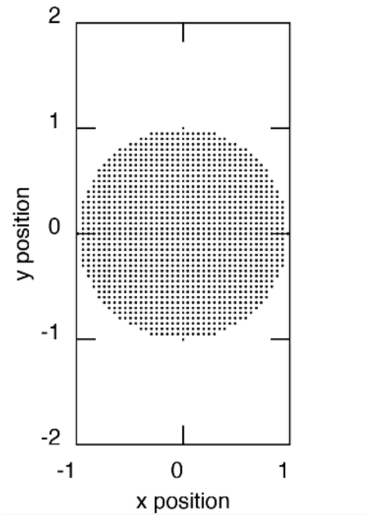
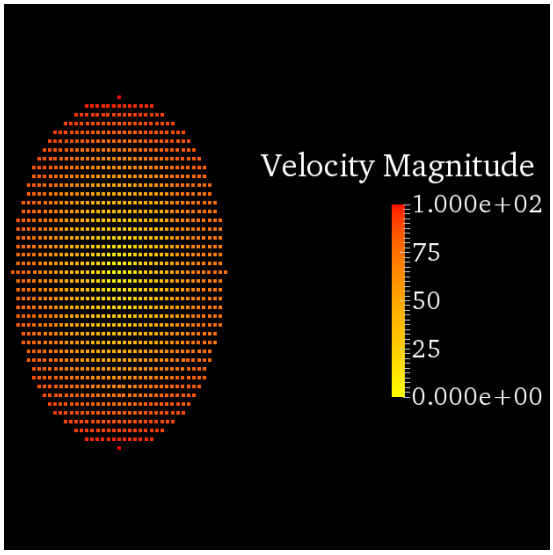
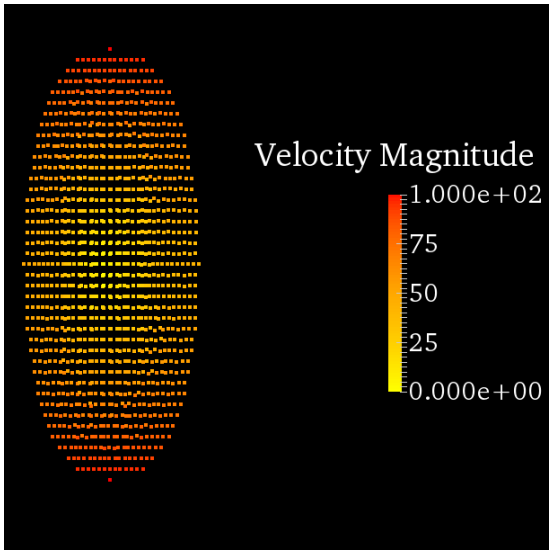


Figure 11 – Initial state for the simulation in the water drop test (scale in meters)

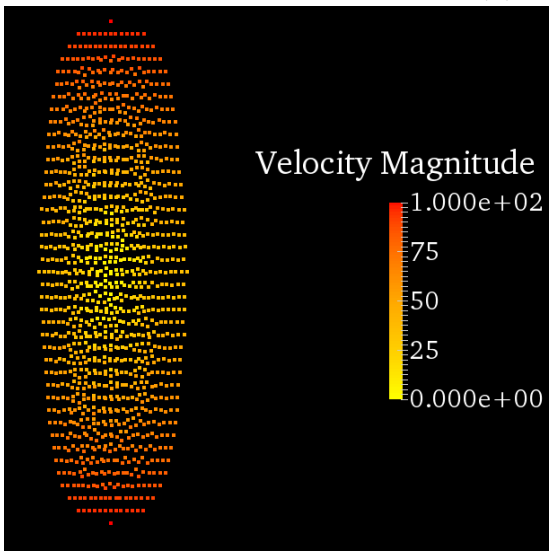
As in (SHAKIBAEINIA; JIN, 2010), the three time steps of the simulation are shown in Figure 12. The recommended kernel function by the authors is the polynomial spiky



(a) $t = 0.0025\ s$



(b) $t = 0.0050\ s$



(c) $t = 0.0075\ s$

Figure 12 – Velocity magnitude profiles (in m/s) of the water drop in three different time steps

function, presented in Equation 5.1.

$$w = \begin{cases} (1 - r/r_e)^3 & 0 \leq r/r_e < 1 \\ 0 & r/r_e \geq 1 \end{cases} \tag{5.1}$$

As noted by Monaghan (MONAGHAN, 1994), the condition in this test to measure the incompressibility of the method is that ab is constant throughout the simulation, where a is the semi-minor axis and b is the semi-major axis of the ellipse. This case ends when the size of b becomes twice the value it were initially, and in that moment the ab value is compared to the its initial value. Errors are kept within less than 0.3 %. Table 4 compares this result to (MONAGHAN, 1994) which keep errors in less than 2 % and (SHAKIBAEINIA; JIN, 2010) which keep errors in less than 0.2 %. This shows the achieved result matches

Table 4 – Comparison of water drop dimensions throughout the simulation

	$t = 0.0\ s$	$t = 0.01\ s$	Difference in %
b (meters)	0.875	1.75	100
ab (meters)	0.766	0.764	0.26

those of the state-of-the-art works.

5.2 DAM BREAK FLOW

The collapse of a water column has been widely used in the literature to validate the numerical precision and the similarity to real-life experiments of the various fluid simulation methods. The dam break problem, as it is known, usually is modeled with the water column initially located on the left side of the recipient, against the left vertical wall. When the simulation starts, the fluid portion collides with the right vertical wall, which generates fragmentation and coalescence of the fluid itself. A variation of the problem, which is modeled with just taking out the right vertical wall and extending the horizontal boundary (floor) has already been used to verify codes for free-surfaces (RAMASWAMY; KAWAHARA, 1987) (STAROSZCZYK, 2010). Originally, Koshizuka and Oka (KOSHIZUKA; OKA, 1996) modeled the dam break problem as shown in Figure 13. In their work, the method proposed is compared to three different experiments conducted in equal manner.

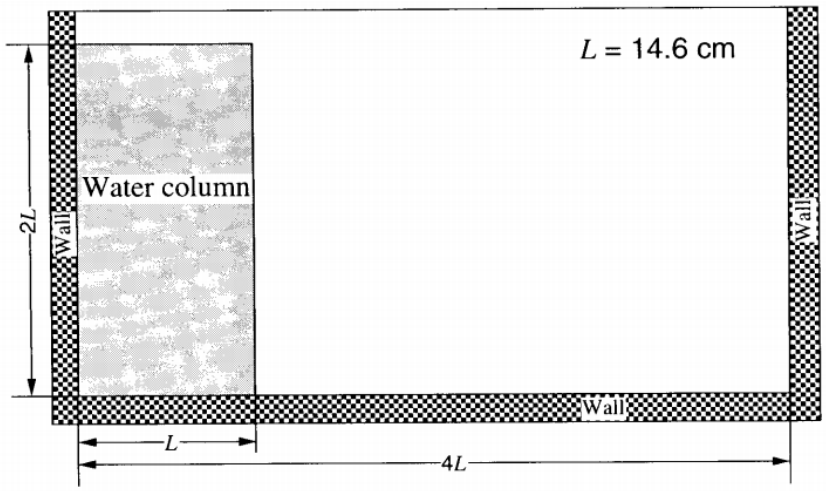


Figure 13 – One possible dam break model. Extracted from (KOSHIZUKA; OKA, 1996)

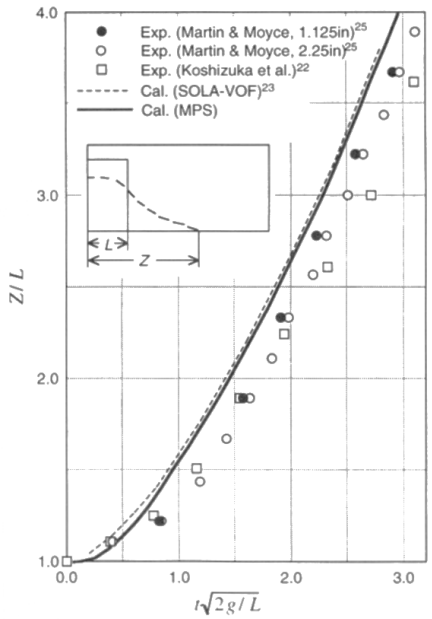
As the test performed by (KOSHIZUKA; OKA, 1996), the water column height is two times bigger then the water column length L. The floor in the model employed here is also four times the length L of the water column. The size of the water column varies depending on how many particles the simulation has. The average particle distance is $10^{-2}\ m$ and the time step of the simulation is $5 \times 10^{-3}\ s$. The total number of particles is 1122. The parameter β is used to judge whether a particle belongs to the free-surface

or not, as explained in the MPS chapter and shown in Equation 3.12. The value used for β is 0.92 as recommended by the authors of the original MPS method. The kernel size is represented by the parameter r_e and is used as the radius of influence of a particle. Koshizuka and Oka show in (KOSHIZUKA; OKA, 1996) that the kernel size should be $< 3.0l_0$, where l_0 is the average particle distance, otherwise the particles will gather near the free-surface. On the other hand, they also show that the discretization of the Laplacian model is more accurate when the kernel size has a higher value. In order to satisfy this, two different kernel sizes were employed, $r_{eP} = 2.1l_0$ and $r_{eLap} = 4.0l_0$.

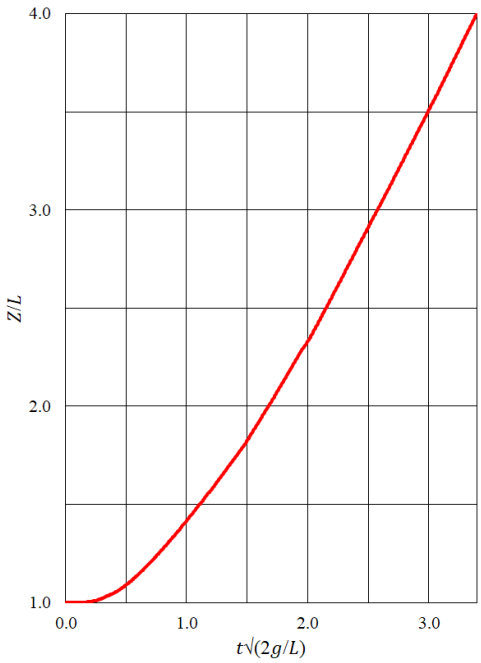
When Koshizuka and Oka first proposed the MPS method, referred to here as standard MPS method, they put it to the test by comparing it to a traditional volume of fluid (VOF) approach (HIRT; NICHOLS, 1981) and experimental data of three different experiments, which are described in (KOSHIZUKA, 1995) and (MARTIN et al., 1952). In this work the dam break model used by Koshizuka and Oka is adopted and the simulation results obtained are compared with the initial results of the MPS and, more importantly, with the experimental results.

To achieve this comparison, the wave front position (its absolute value is represented by Z) must be tracked as a function of time since the beginning of the dam burst. The wave front position is represented by a dimensionless format, Z/L, where L is the water column initial length, which for this test is equal to 0.18 m. As said before, the size of the floor is four times the size of L, which implies that the maximum value Z can reach is $4L$ (that being the reason that the y-axis values ends when $Z/L = 4$ in Figure 14 and Figure 15). The time in the chart is represented by a dimensionless format as well: $t\sqrt{2g/L}$, where t is the time in seconds and g the gravitational acceleration, which is equal to 9.8 m/s^2. Figure 14a shows the data from (KOSHIZUKA; OKA, 1996) in which the standard MPS is compared to experimental data and another technique and Figure 14b shows the data from the developed method. The configuration of the developed method used for this comparison was the turbulence and viscosity-enabled CMPS-HS-HL-ECS (each one of these variations and models were previously explained in section 3.2 and section 3.3). In Figure 15 the two images from Figure 14 are overlapped in order to enable a direct comparison.

Figure 15 shows that the developed method gets a lot closer to the experimental data than the other methods. Interestingly, it almost overlaps the experiment represented by the small empty squares. It is important to note that, differently from the method, the experiments do not become outdated since the configuration did not change over the years and that they match the reality, therefore, this corroborates the assumption of high precision and high physical coherence of the method proposed in this study.



(a) Two methods compared to three different experiments. Extracted from (KOSHIZUKA; OKA, 1996)



(b) The developed method

Figure 14 – Evolution of wave front position of a few methods and experiments in the left and evolution of wavefront position provided by the developed method

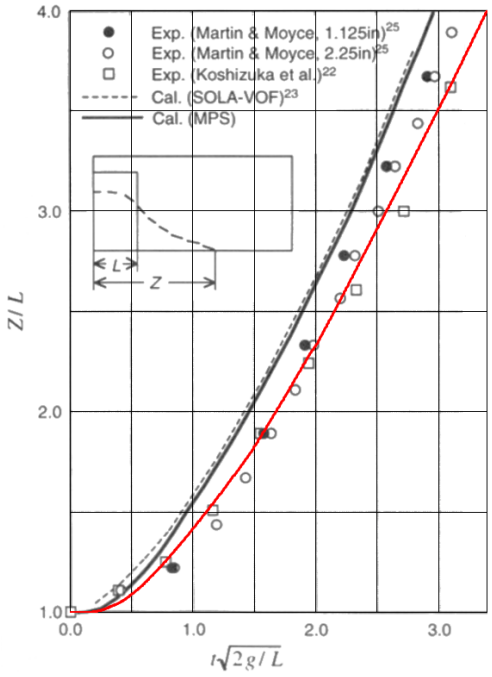


Figure 15 – Direct comparison of the developed method (in red) with the experimental results and other methods

5.3 R-T INSTABILITY

Here, the Rayleigh-Taylor instability (YOUNGS, 1984) will be demonstrated through the developed method. This test's initial state simplicity is valuable since it provides an opening into the mathematical study of stability. Its main goal is to show the multiphase model ability to handle the density stratification and to evolve a linear perturbation into a nonlinear hydrodynamic turbulence.

The test consists in placing the same amount of two immiscible fluids with different densities and same kinematic viscosity, one on the top of the other. The heavier one will stay on top of the lighter, only influenced by the gravitational force g inside a two dimensional rectangular box. So, when the simulation starts the heavier fluid will tend to go downwards, pushing the lighter fluid upwards. The interface between the fluids will become unstable and the format of the pattern generated at the interface is critical to judge whether the test has been successful or not. The lighter fluid forms a bubble in the shape of a mushroom cap that breaks eventually. The problem is characterized by the Atwood number, calculated as shown in Equation 5.2.

$$A_t = \frac{\rho_h - \rho_l}{\rho_h + \rho_l} \tag{5.2}$$

ρ_h and ρ_l refer to the densities of the heavier and lighter fluid, respectively. For the test used in this study, $A_t = 1/3$ and the kinematic viscosity $\nu = 0.010$ for both fluids, following (SHAKIBAEINIA; JIN, 2012). The test has 3066 particles. Figure 16 shows a few steps of the generated simulation and it is possible to visually compare it to Figure 17 a mesh-based approach extracted from (LI; LI, 2006). Note that Figure 17 has a quite large number of small vortices at the interface. This is another instability phenomenon, called the Kelvin-Helmholtz instability (SHAKIBAEINIA; JIN, 2012), but it will not be addressed here since it is not the focus.

It is clear that the generated simulation by the implemented approach has a similar

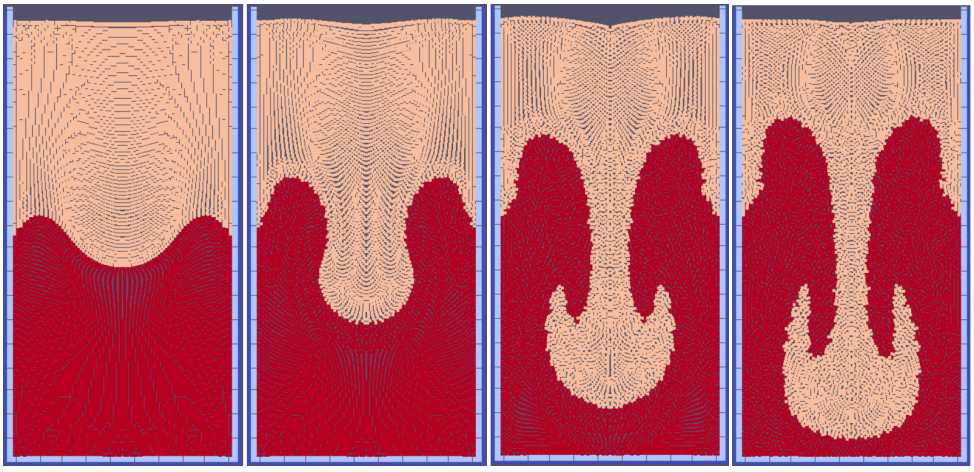


Figure 16 – Rayleigh-Taylor instability generated by proposed method

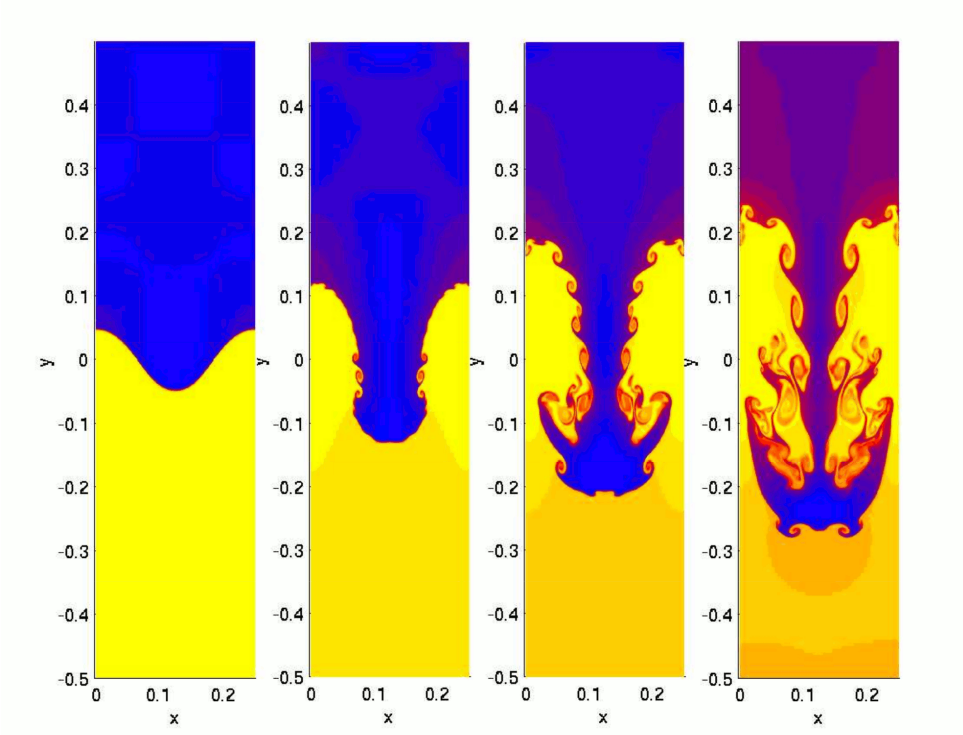


Figure 17 – Rayleigh-Taylor instability by a mesh-based approach. Extracted from (LI; LI, 2006)

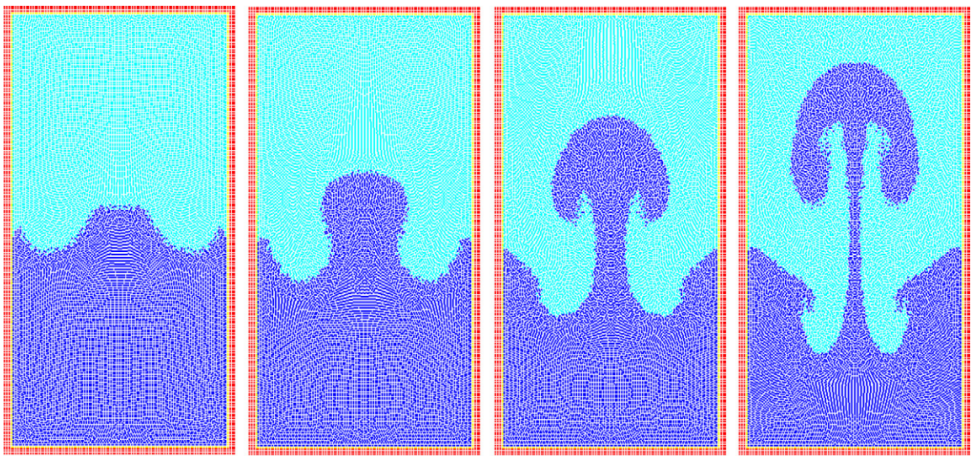


Figure 18 – Rayleigh-Taylor instability by a basic multiphase MPS method. Extracted from (SHAKIBAEINIA; JIN, 2012)

formation of the mushroom cap of the heavier fluid going downwards. The implemented method shows generally good agreement regarding the Rayleigh-Taylor instability with the advantage of much simpler formulations and less computational cost.

For a better context on to how other particle methods behave in this simulation, Figure 18 shows the same phenomenon in a similar test case (SHAKIBAEINIA; JIN, 2012). In this scenario there is already an initial perturbation to the fluids' interface, which causes the bubble (mushroom cap) to go upwards.

5.4 OIL SPILL

To validate the multiphase system, treated as a multi-density and multi-viscosity fluid, a qualitative analysis and comparison with an experiment and another method is conducted. (DUAN et al., 2017) provides a replicable test of a continuous oil spill due to a damaged tank. Figure 19, extracted from the mentioned paper, describes the test in details.

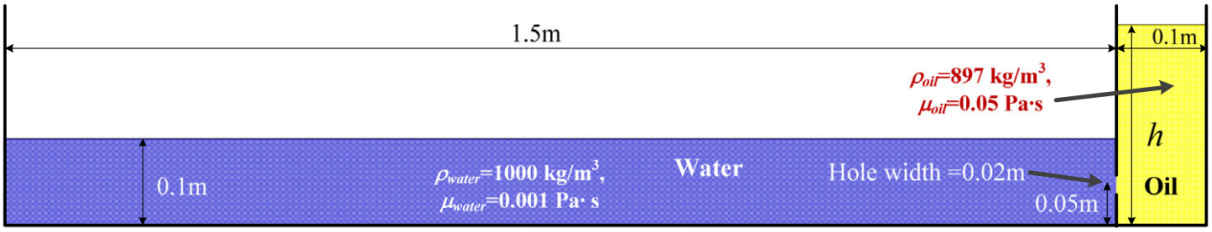
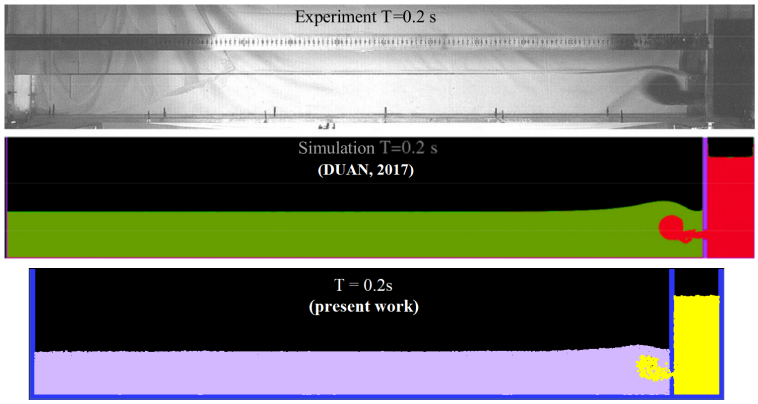


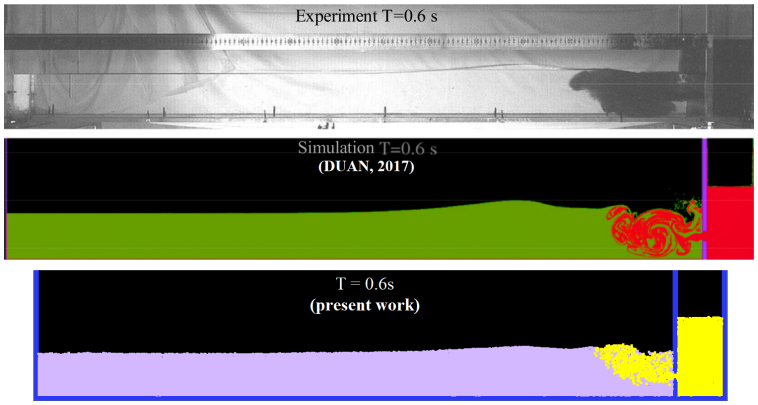
Figure 19 – Description of oil spilling scenario extracted from (DUAN et al., 2017)

The value of h in Figure 19 is 0.25 m, as it is the adopted value in (DUAN et al., 2017) to show a detailed simulation. The average particle distance is 4×10^{-3} m, the water and oil's kinematic viscosity ν_{water} and ν_{oil} are, respectively, 10^{-6} m^2/s and 5×10^{-5} m^2/s and the used method is the multiphase and turbulent WC-CMPS. This test has a total particle number of 12779.The generated leakage simulation is qualitatively compared to experimental results and to the proposed method by Duan and his colleagues, a fully incompressible multiphase MPS. Figure 21 shows five time frames of the experiments and the simulations. In each time frame there is an image of the experiment, the simulation by (DUAN et al., 2017) and the simulation by the present work, in that order.

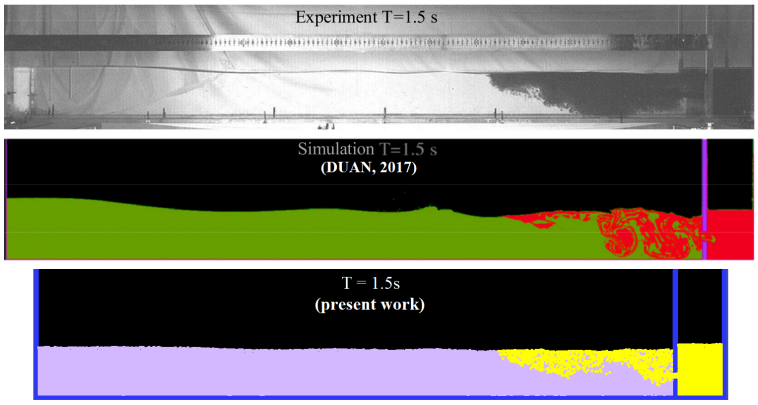
It is important to note that the FS-MMPS is assumed to be quite precise and accurate since a Poisson pressure equation is solved to obtain each particle pressure. However, it is possible to observe that the implemented multiphase WC-MPS can better predict the pointy behavior of the left-end of the oil (in yellow), mainly in Figure 21d and Figure 21e. In a general manner, the oil profile and the waves generated by it are in reasonable agreement with the generated by the experiment, with far less computations compared to



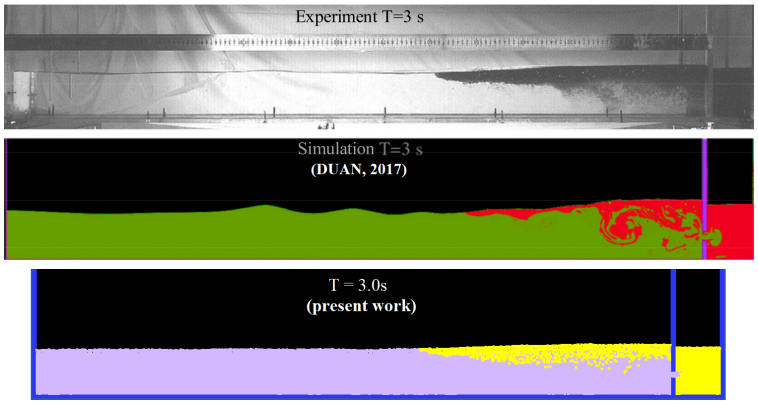
(a) $t = 0.2$ s



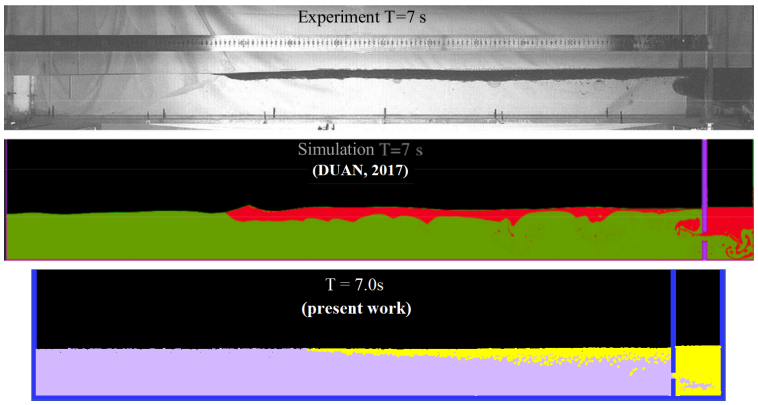
(b) $t = 0.6\ s$



(c) $t = 1.5\ s$



(d) $t = 3\ s$



(e) $t = 7\ s$

Figure 21 – Comparison between experiment, FS-MMPS (DUAN et al., 2017) and the implemented multiphase WC-MPS at different moments of the oil spilling

the FS-MMPS, since the the multiphase WC-MPS only solves a single set of equations, which significantly diminishes the amount of computational operations performed.

5.5 MEMORY USAGE ANALYSIS

In order to evaluate strong and weak spots of the system's performance, this section will detail the time duration and memory usage of each simulation approach combined with three different types of execution: completely sequential; parallelized through OpenMP; and parallelized through CUDA. Finally, the absolute time spent by each version will be computed for a specific test to calculate the speedup reached by the parallelized implementations of the method.

The Performance and Diagnostics tool of Visual Studio 2015 (MICROSOFT, 2018) was used to obtain details related to the sequential and OpenMP executions and to evaluate CPU memory usage in all versions. As for the GPU memory, the GPU-Z software (TECHPOWERUP, 2018) provided the desired information in real-time as well as the minimum and maximum usage throughout the simulation runtime. The NVIDIA Visual profiler (NVIDIA, 2018) provided the majority of informations regarding the GPU such as the whole simulation runtime, time spent in each CUDA kernel during the execution and temperature of the GPU throughout the process.

Table 5 and Table 6 show memory usage of the WC approach of the MPS method for a standard 2D dam break test, like the one presented in section 5.2. For this case, there is a total of $152,332$ particles which $145,800$ of those are fluid particles. 152k particles is considered a medium amount of particles for a weakly compressible approach assuming that a GPU is used.

Table 5 – Absolute memory usage of different executions of the WC-MPS for 152k particles

	Sequential	OpenMP	CUDA
Required CPU Memory (MB)	222	222	290
Required GPU Memory (MB)	-	-	463

Table 6 – Occupancy in the CPU and GPU memories of different executions of the WC-MPS for 152k particles

	Sequential	OpenMP	CUDA
Required CPU Memory (%)	2.9 %	2.9 %	3.8 %
Required GPU Memory (%)	-	-	23.15 %

The memory usage relative to the FI approach is shown in Table 7 and Table 8 for the same test described above and in section 5.2. 152k particles is considered a high amount of particles for a FI approach, mainly if the GPU is used, since a linear system of size $N \times N$, which N is the total particle number, has to be fully loaded in the memory. This is also considered an improvement, since previous works (SILVA et al., 2017; SILVA et al., 2018) could not even achieve this order of magnitude.

Table 7 – Absolute memory usage of different executions of the FI-MPS for 152k particles

	Sequential	OpenMP	CUDA
Required CPU Memory (MB)	967	967	997
Required GPU Memory (MB)	-	-	1174

Table 8 – Occupancy in the CPU and GPU memories of different executions of the FI-MPS for 152k particles

	Sequential	OpenMP	CUDA
Required CPU Memory (%)	12.6 %	12.6 %	13.0 %
Required GPU Memory (%)	-	-	59.0 %

Note that, in both approaches, the CUDA versions require not only the GPU memory but also CPU memory, which is also a slightly bigger amount compared to OpenMP and the fully sequential executions. This happens due to that most arrays and vectors must exist in host memory (CPU) as well as in the device memory (GPU) in order to transfer the particle input information from the CPU to the GPU and to be able to access the information being processed in the GPU.

A noteworthy information related to memory is the maximum number of particles reached by the fully incompressible approach, in which the size of the simulation turned out to be an issue. So, when running in the CPU with 8 GB of available memory, $1,054,122$ particles was the limit before the program crashed, and, in the GPU with 2 GB of available memory, $212,522$ particles was the limit also before a crash occurred. Considering the linear system, this is an interesting amount of particles for the used PC configurations and the accuracy and precision of the fully incompressible approach. This also shows that the memory usage is at almost a linear rate, since four times of more memory enabled a simulation with approximately four times more particles.

5.6 PERFORMANCE ANALYSIS

Following the goal of evaluating the system's performance, the execution time of each one of the execution possibilities were measured as well as the duration details of each function in the execution.

5.6.1 Functions

This subsection aims to expose the system's bottleneck in its different executions and approaches. By comparing different approaches in similar execution context it will be possible to better evaluate the functions performance and which of them deserve a further study in order to diminish the impact to the system's computation. As in section 5.5, the

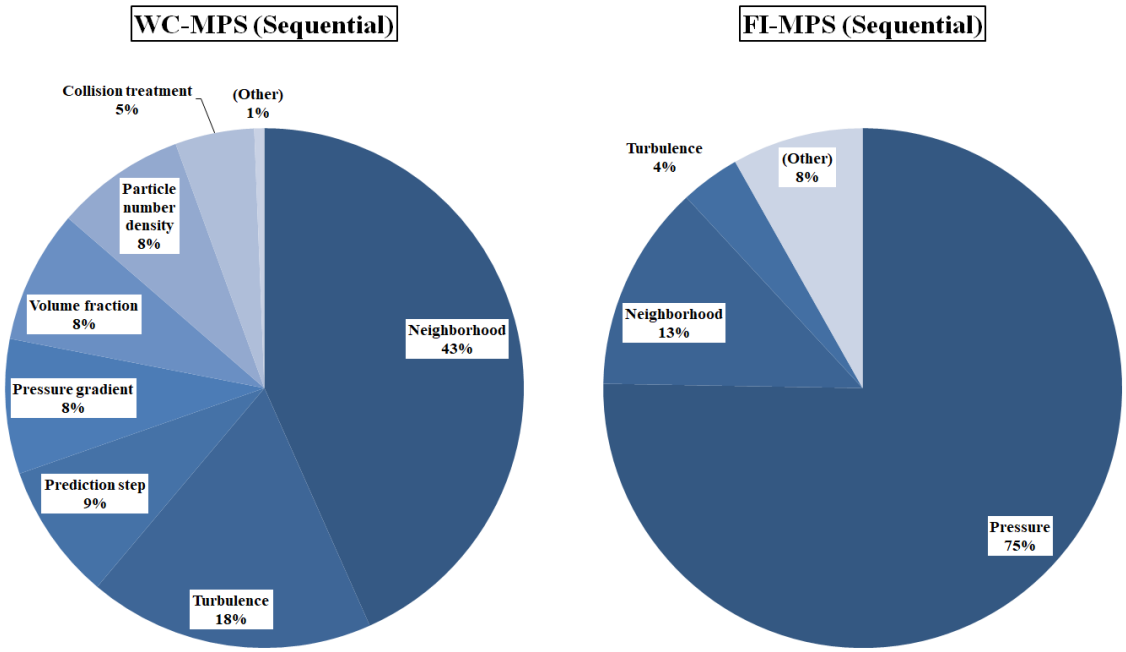


Figure 22 – Functions profile from sequential execution of WC-MPS and FI-MPS

same test case with 152k particles was used for profiling the functions and the speedup of the parallelized versions.

Figure 22 shows the immense computational drawback that is the need to solve the Poisson pressure equation. It is relevant to note that the only difference from the weakly compressible MPS to the fully incompressible MPS is inside the pressure calculation function. All other functions remain the same except for minor changes in certain functions that do not affect performance relevantly. Figure 22 also shows that, despite of the improvement in the neighborhood calculation related to (SILVA et al., 2017) and (SILVA et al., 2018), it still is, in both versions, an execution bottleneck even after the implementation of the cell linked-list scheme. Th PPE solution in the FI-MPS remains a challenge also despite of the very important improvement of expanding the simulation size in approximately 32 times (SILVA et al., 2017) by decreasing the linear system size.

Regarding Figure 23, it is possible to see that relatively, almost nothing has changed from the totally sequential execution despite of the performance gain that will be detailed in subsection 5.6.2. That was an interesting result because the performance gain was relevant as it is going to be detailed, however, there are not any significant changes in the execution profile, which is initially a good sign, since it shows that there is balance in this parallelization scale: all functions are equally speeding up.

Finally, Figure 24 presents the CUDA kernels profiles. Here, the PPE solution process and neighborhood search performance problems emerges. The reasons for both functions behave in that way are somewhat apparent.

For the PPE solution a numerical iterative method is used to solve the linear system, the ICCG method, in which it consists in performing an incomplete-Cholesky factorization

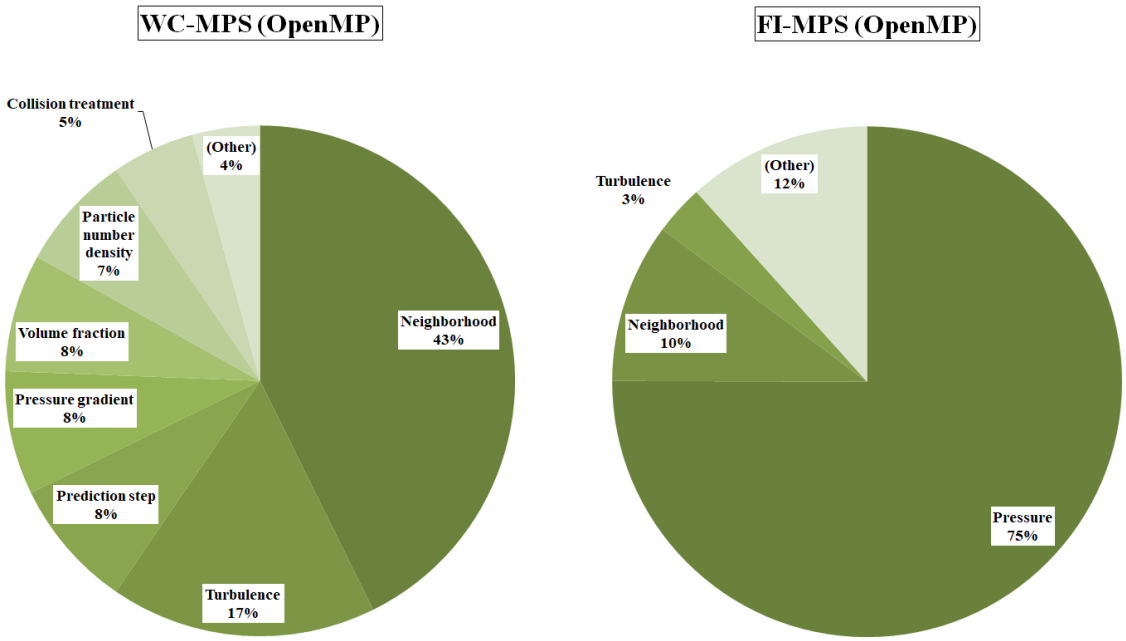


Figure 23 – Functions profile from OpenMP execution of WC-MPS and FI-MPS

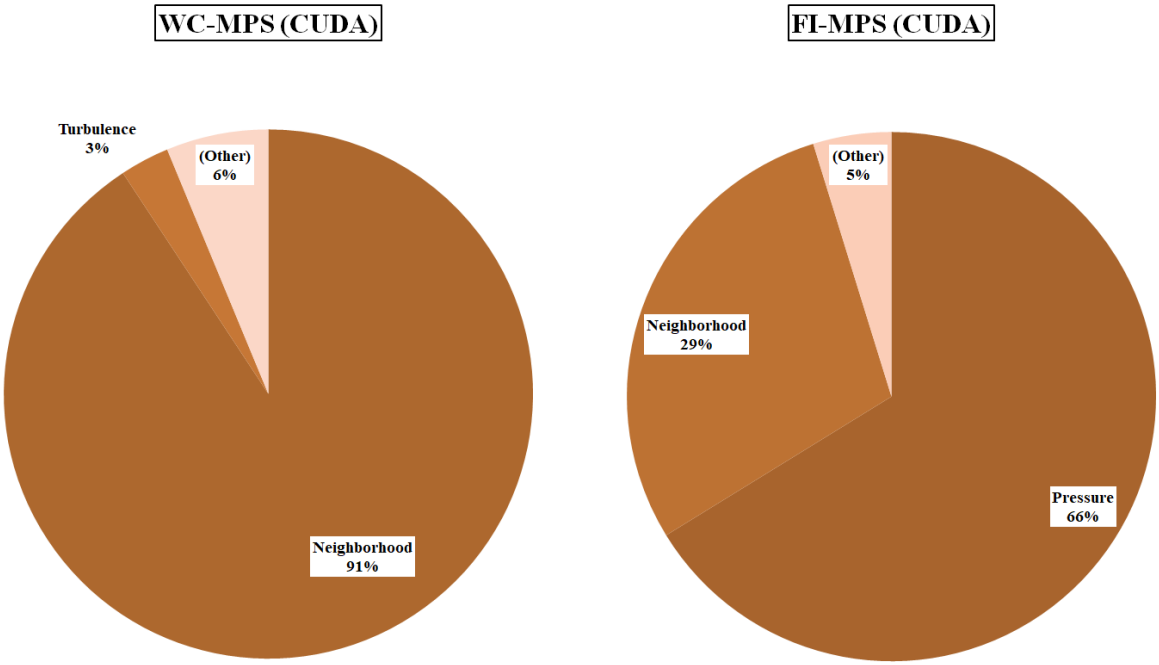


Figure 24 – Functions profile from CUDA execution of WC-MPS and FI-MPS

of the coefficient matrix before it is used by the conjugate gradient method. The referred factorization serve as a preconditioner for the matrix, with the purpose of decreasing its condition number so less significant numerical errors are produced. The time performance problem that arises in the CUDA version from the IC factorization method is that it is a intrinsically sequential method, with few and complex alternatives on how to diminish its computational time. The reason of the neighborhood search bottleneck has a similar justification to the PPE's; a small part of the neighborhood search algorithm is intrinsically sequential and can not be parallelized. So, since the rest of the other functions receive a great performance boost, the neighborhood search and the PPE solution stay far behind.

For those issues, a proper solution would be reimplement the cell linked-list scheme in a manner parallelization can be achieved more easily by the whole algorithm.

5.6.2 Speedup

In spite of all the above presented performance issues, speedups could be achieved. Table 9 shows the absolute time duration for each method combined with the different executions.

Table 9 – Execution duration for WC-MPS and FI-MPS in different implementation versions

	Sequential	OpenMP	CUDA
WC-MPS (seconds)	1362	675	371
FI-MPS (seconds)	1196	657	535

Table 10 – Speedups of WC-MPS and FI-MPS in different implementations versions

	Sequential	OpenMP	CUDA
WC-MPS	1	2.02	3.15
FI-MPS	1	1.82	2.23

5.6.3 Simulation limits

This subsection aims to show data regarding performance and memory limitations, such as maximum frame rate achieved with some specific number of particles and maximum number of particles that can be loaded in memory during a simulation. Table 11 and Table 12 show how much frames per second (fps) can be generated depending on the number of particles in the simulation. In Table 11 the neighborhood is recalculated every step of the simulation and in Table 12 the neighborhood is recalculated every other step. The simulation used for this test was a classic 2D dam break with initial particle distance of 4×10^{-3} m, varying the total particle number.

To illustrate, the test with $18.4k$ particles has a liquid column with height of 0.72 m and length equivalent to half the height size. It is important to note that, increasing

Table 11 – Fps achieved given number of particles and steps to neighborhood recalculation

Particles	fps
4.5×10^4	2
1.84×10^4	10
10^4	23

Table 12 – Fps achieved given number of particles and steps to neighborhood recalculation

Particles	fps
1.41×10^5	3.3
4.5×10^4	10
1.84×10^4	24

the amount of steps between neighborhood recalculations the simulation loses a bit in precision, however, gains significantly in performance, since this calculation has great impact on this system execution time, as shown in subsection 5.6.1.

5.6.4 Temperature

In order verify whether the hardware was also behaving appropriately, the temperature information of the GPU card from both executions with the highest performance was extracted. For this, the NVDIA Visual Profiler was also used. Table 13 shows the lowest, highest and average temperatures reached by the GPU card during the executions.

Table 13 – GPU card temperature during the CUDA-enabled implementations

	Lowest	Highest	Average
WC-MPS (°C)	38	69	62
FI-MPS (°C)	39	74	69

NVIDIA says that, in general the maximum temperature of GeForce GTX is 105 °C (NVIDIA, 2010). Therefore, the displayed temperatures are inside that boundary, which shows that the hardware is operating accordingly.

6 CONCLUSIONS

As previously discussed, the study of fluid flow simulation is of great importance in helping mitigate the consequences of environmental disasters and accidents. It has applications in a wide range of engineering problems and virtual and augmented reality technologies. Meshless methods like the MPS are a great alternative to deal with large deformations and free-surface flow, situations where the traditional mesh-based methods usually perform inefficiently.

Throughout the development of his research it became clear that, despite the discussed method suffers from certain instability problems, there have been constant improvements (both performance or accuracy-related enhancements) described throughout the literature that justify its use in a large number of scenarios. The considerable amount of referenced works also show the complexity of this task, the importance of the method to the community and the great potential it has to simulate, increasingly more realistically, fluid flows. Regarding the MPS optimization, it is a delicate subject since this method is commonly used for replicating real phenomena more reliably when taking into consideration other meshless approaches. Some authors are able to enhance its performance with acceleration structures and algorithms without losing the precision the method offers (or even accelerating it with numerical improvements) (HORI et al., 2011) (ZHU et al., 2011) (SILVA et al., 2017) (SILVA et al., 2018), while other authors prefer, despite the precision loss, replace the clear performance and memory bottleneck of the method, which is the assembly and solution of the PPE, with a state equation to solve the pressures (SHAKIBAEINIA; JIN, 2010) (TAYEBI; JIN, 2015) and then, parallelize it through GPU (TANIGUCHI; SATO; CHENG, 2014) (FERNANDES, 2013). Also, the hardware development must be considered since even more powerful CPUs and GPUs are being manufactured every year, increasing the parallelized systems performance.

After a research in the literature, it was observed that only the standard MPS method (KOSHIZUKA; OKA, 1996) or the basic weakly compressible version (SHAKIBAEINIA; JIN, 2010) has been parallelized. This master research provides a wide variety of models, improvements and approaches to the MPS technique in which both are entirely parallelized, through OpenMP and CUDA, all integrated in a single Visual Studio solution. This enables a practical selection between models, approaches and parameter tuning, from a physically coherent free-surface incompressible fluid flow simulation, to a GPU-accelerated multiphase free-surface weakly compressible flow simulation. Regarding the numerical improvements, the techniques proposed by (GOTOH, 2013) and (SHAKIBAEINIA; JIN, 2012) were combined and extended, which generated important enhancements in physical coherence, as presented in the Dam Break, Oil Spill and Water Drop tests. In these tests, results such as the better matching of the fluid's wavefront with experiments and the

matching of the oil profile in water with the experiments, show the numerical advances achieved. Also, the proposed method shows to be compatible with the state of the art in the Oil Spill test, by qualitatively comparing the proposed method with (DUAN et al., 2017). The OpenMP-enabled weakly compressible approach achieves a speedup of 2.02 times and the fully incompressible approach of 1.82 times. The CUDA-enabled weakly compressible approach achieves a speedup of 3.15 times while the fully incompressible approach of 2.23 times.

Throughout the development of this work, six papers were produced, four of them were published (SILVA et al., 2017) (SILVA et al., 2015) (BRITO et al., 2017) (BRITO et al., 2017), (SILVA et al., 2018) is accepted for publication and (BRITO et al., 2018) was submitted and waiting for acceptance. A book chapter was also published disserting about about meshless methods for fluid simulation while in a partnership with Universidade de São Paulo (USP) (ASSI et al., 2016a).

6.1 FUTURE WORK

There is an interesting number of possibilities for future developments of this work. Surely, refinements in every small part of the code, mainly where improvement possibilities had not been seen, will lead to a more optimized version; this is considered the path to a notable real-time simulation.

One possibility that would enhance even further the method's performance is to improve the GPU implementation of the fully incompressible MPS to be able to run on multiple GPU simultaneously inside the the same machine or even in a GPU cluster, which could raise the speedup values to new levels. (TANIGUCHI; SATO; CHENG, 2014) shows the performance of weakly compressible MPS aided by a GPU cluster; fully incompressible versions of the MPS could benefit greatly from such structures. (FERNANDES et al., 2015) presents a domain decomposition strategy for a parallelized MPS for running in a cluster of computers. This enables larger simulations since the simulation domain can be loaded and solved separately in each one of the pc's memories and then integrated back.

Another clear possibility is the issue presented in cell linked-list neighborhood algorithm implemented. The implementation of this acceleration structure enhanced the performance of the sequential and the OpenMP versions of the code, however it turned out to be a problem in the CUDA version since it was developed to run in parallel and became a bottleneck for the GPU versions. An important next step is to rethink and develop a new cell linked-list scheme from scratch that is able to run in different threads simultaneously (DOMÍNGUEZ; CRESPO; GÓMEZ-GESTEIRA, 2013).

Still on the optimization field, since the PPE presents a great deal of issues regarding computational performance, an alternative to the code parallelization could be the usage of neural networks that learns the usual results of commonly yielded linear systems

by incompressible fluid simulation systems. The work of (TOMPSON et al., 2016) shows promising fluid simulations using this approach.

The addition of a model that allows the interaction between fluids and solids such as floating bodies, deformable bodies and viscoelastic fluid would increase drastically the number of application possibilities of the method, mainly if a graphical user interface is implemented and fully tested and experimented.

The present work was supported by the CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil (National Council of Scientific and Technological Development - Brazil), process number: 456332/2013-8.

REFERENCES

AHRENS, J.; GEVECI, B.; LAW, C. 36 paraview: An end-user tool for large-data visualization. *The Visualization Handbook*, Citeseer, p. 717, 2005.

ALMEIDA, M.; BRITO, C.; SILVA, A. L. B. Vieira e; TEICHRIEB, V.; BARBOSA, J. M. Meshless methods. In: ASSI, G.; BRINATI, H.; CONTI, M. de; SZAJNBOK, M. (Ed.). *Applied Topics in Marine Hydrodynamics*. São Paulo: Escola Politécnica da Universidade de São Paulo (ISBN 978-85-86686-89-4), 2016. chap. 8, p. 8.1–8.38.

ASSI, G.; BRINATI, H.; CONTI, M.; SZAJNBOK, M. Meshless methods. In: ASSI, G.; BRINATI, H.; CONTI, M.; SZAJNBOK, M. (Ed.). *Applied Topics in Marine Hydrodynamics*. São Paulo: [s.n.], 2016.

ASSI, G.; BRINATI, H.; CONTI, M. de; SZAJNBOK, M. *Applied Topics in Marine Hydrodynamics*. [S.l.]: EPUSP, 2016. ISBN 978-85-86686-89-4.

ATAIE-ASHTIANI, B.; FARHADI, L. A stable moving-particle semi-implicit method for free surface flows. *Fluid Dynamics Research*, Elsevier, v. 38, n. 4, p. 241–256, 2006.

BATCHELOR, G. *K. 1967 an introduction to fluid dynamics*. [S.l.]: Cambridge University Press, 1970.

BELL, N.; YU, Y.; MUCHA, P. J. Particle-based simulation of granular materials. In: *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. New York, NY, USA: ACM, 2005. (SCA '05), p. 77–86. ISBN 1-59593-198-8. Available at: <http://doi.acm.org/10.1145/1073368.1073379>.

BELYTSCHKO, T.; KRONGAUZ, Y.; ORGAN, D.; FLEMING, M.; KRYSL, P. Meshless methods: an overview and recent developments. *Computer methods in applied mechanics and engineering*, Elsevier, v. 139, n. 1, p. 3–47, 1996.

BELYTSCHKO, T.; KRONGAUZ, Y.; ORGAN, D.; FLEMING, M.; KRYSL, P. Meshless methods: An overview and recent developments. *Computer Methods in Applied Mechanics and Engineering*, v. 139, n. 1, p. 3 – 47, 1996. ISSN 0045-7825. Available at: <http://www.sciencedirect.com/science/article/pii/S004578259601078X>.

BONET, J.; LOK, T.-S. Variational and momentum preservation aspects of smooth particle hydrodynamic formulations. *Computer Methods in applied mechanics and engineering*, Elsevier, v. 180, n. 1, p. 97–115, 1999.

BRITO, C. J. dos S.; SILVA, A. L. B. V. e; ALMEIDA, M. W. S.; TEIXEIRA, J. M. X. N.; TEICHRIEB, V. Large viscoelastic fluid simulation on GPU. In: *2017 16th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. [S.l.: s.n.], 2017. p. 134–143. ISSN 2179-2259.

BRITO, C. J. dos S.; SILVA, A. L. B. V. e; TEIXEIRA, J. M. X. N.; TEICHRIEB, V. Ray tracer based rendering solution for large scale fluid rendering. *Computers & Graphics (submitted; under review)*, 2018.

BRITO, C. J. S.; ALMEIDA, M. W. S.; SILVA, A. L. B. V. e; TEIXEIRA, J. M. X. N.; TEICHRIEB, V. Screen space rendering solution for multiphase SPH simulation. In: *2017 19th Symposium on Virtual and Augmented Reality (SVR)*. [S.l.: s.n.], 2017. p. 309–318.

CHEN, G.; ONISHI, Y.; ZHENG, L.; SASAKI, T. *Frontiers of Discontinuous Numerical Methods and Practical Simulations in Engineering and Disaster Prevention*. [S.l.]: Taylor & Francis Group, London, 2013. ISBN 978-1-138-00110-7.

CHEN, R.; TIAN, W.; SU, G.; QIU, S.; ISHIWATARI, Y.; OKA, Y. Numerical investigation on coalescence of bubble pairs rising in a stagnant liquid. *Chemical Engineering Science*, Elsevier, v. 66, n. 21, p. 5055–5063, 2011.

CLEARY, P.; PRAKASH, M.; HA, J. Novel applications of smoothed particle hydrodynamics (sph) in metal forming. *Journal of materials processing technology*, Elsevier, v. 177, n. 1, p. 41–48, 2006.

DAGUM, L.; MENON, R. Openmp: an industry standard api for shared-memory programming. *IEEE Computational Science and Engineering*, v. 5, n. 1, p. 46–55, Jan 1998. ISSN 1070-9924.

DOMÍNGUEZ, J. M.; CRESPO, A. J.; GÓMEZ-GESTEIRA, M. Optimization strategies for cpu and gpu implementations of a smoothed particle hydrodynamics method. *Computer Physics Communications*, v. 184, n. 3, p. 617 – 627, 2013. ISSN 0010-4655. Available at: <http://www.sciencedirect.com/science/article/pii/S001046551200358X>.

DUAN, G.; CHEN, B. Stability and accuracy analysis for viscous flow simulation by the moving particle semi-implicit method. *Fluid Dynamics Research*, IOP Publishing, v. 45, n. 3, p. 035501, 2013.

DUAN, G.; CHEN, B.; ZHANG, X.; WANG, Y. A multiphase mps solver for modeling multi-fluid interaction with free surface and its application in oil spill. *Computer Methods in Applied Mechanics and Engineering*, v. 320, p. 133 – 161, 2017. ISSN 0045-7825. Available at: <http://www.sciencedirect.com/science/article/pii/S0045782516317819>.

FERNANDES, D. T. *Implementação de framework computacional de paralelização híbrida do Moving Particle Semi-implicit Method para modelagem de fluidos incompressíveis*. Phd Thesis (PhD Thesis) — Universidade de São Paulo, 2013.

FERNANDES, D. T.; CHENG, L.-Y.; FAVERO, E. H.; NISHIMOTO, K. A domain decomposition strategy for hybrid parallelization of moving particle semi-implicit (mps) method for computer cluster. *Cluster Computing*, v. 18, n. 4, p. 1363–1377, Dec 2015. ISSN 1573-7543. Available at: <https://doi.org/10.1007/s10586-015-0483-3>.

FREY, P.-J.; ALAUZET, F. Anisotropic mesh adaptation for cfd computations. *Computer methods in applied mechanics and engineering*, Elsevier, v. 194, n. 48, p. 5068–5082, 2005.

FU, L.; JIN, Y.-C. Investigation of non-deformable and deformable landslides using meshfree method. *Ocean Engineering*, v. 109, p. 192 – 206, 2015. ISSN 0029-8018. Available at: <http://www.sciencedirect.com/science/article/pii/S0029801815004497>.

Fuji Technical Research Inc. *MPS-RYUJIN*. 2013. Accessed: 2018-04-23. Available at: <http://www.ftr.co.jp/n/eng/products/mps_ryujin/index.html>.

GINGOLD, R. A.; MONAGHAN, J. J. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly notices of the royal astronomical society*, Oxford University Press, v. 181, n. 3, p. 375–389, 1977.

GOTOH, H. Advanced particle methods for accurate and stable computation of fluid flows. *Frontiers of Discontinuous Numerical Methods and Practical Simulations in Engineering and Disaster Prevention*, CRC Press, p. 113, 2013.

GOTOH, H.; IKARI, H.; MEMITA, T.; SAKAI, T. Lagrangian particle method for simulation of wave overtopping on a vertical seawall. *Coastal Engineering Journal*, v. 47, n. 02n03, p. 157–181, 2005. Available at: <https://www.worldscientific.com/doi/abs/10.1142/S0578563405001239>.

GOTOH, H.; KHAYYER, A. Current achievements and future perspectives for projection-based particle methods with applications in ocean engineering. *Journal of Ocean Engineering and Marine Energy*, v. 2, n. 3, p. 251–278, Aug 2016. ISSN 2198-6452. Available at: <https://doi.org/10.1007/s40722-016-0049-3>.

GOTOH, H.; KHAYYER, A. On the state-of-the-art of particle methods for coastal and ocean engineering. *Coastal Engineering Journal*, Taylor & Francis, v. 0, n. 0, p. 1–25, 2018. Available at: <https://doi.org/10.1080/21664250.2018.1436243>.

GOTOH, H.; KHAYYER, A.; IKARI, H.; HORI, C. 3d-cmps method for improvement of water surface tracking in breaking waves. In: WORLD SCIENTIFIC. *Proceedings of 4th SPHERIC Workshop. Nantes, France,:[sn]*. [S.l.], 2009. p. 265–272.

GOTOH, H.; SAKAI, T. Lagrangian simulation of breaking waves using particle method. *Coastal Engineering Journal*, v. 41, n. 03n04, p. 303–326, 1999. Available at: <https://www.worldscientific.com/doi/abs/10.1142/S0578563499000188>.

GOTOH, H.; SAKAI, T. Key issues in the particle method for computation of wave breaking. *Coastal Engineering*, v. 53, n. 2, p. 171 – 179, 2006. ISSN 0378-3839. Coastal Hydrodynamics and Morphodynamics. Available at: <http://www.sciencedirect.com/science/article/pii/S037838390500133X>.

GOTOH, H.; SHIBAHARA, T.; SAKAI, T. Sub-particle-scale turbulence model for the MPS method - Lagrangian flow model for hydraulic engineering. *Advanced Methods for Computational Fluid Dynamics*, v. 9, p. 339–347, 2001.

HARADA, E.; GOTOH, H.; IKARI, H.; KHAYYER, A. Numerical simulation for sediment transport using mps-dem coupling model. *Advances in Water Resources*, 2017. ISSN 0309-1708. Available at: <http://www.sciencedirect.com/science/article/pii/S0309170817300428>.

HIRT, C.; NICHOLS, B. Volume of fluid (vof) method for the dynamics of free boundaries. *Journal of Computational Physics*, v. 39, n. 1, p. 201 – 225, 1981. ISSN 0021-9991. Available at: <http://www.sciencedirect.com/science/article/pii/0021999181901455>.

HORI, C.; GOTOH, H.; IKARI, H.; KHAYYER, A. Gpu-acceleration for moving particle semi-implicit method. *Computers & Fluids*, Elsevier, v. 51, n. 1, p. 174–183, 2011.

HWANG, S.-C.; KHAYYER, A.; GOTOH, H.; PARK, J.-C. Development of a fully lagrangian mps-based coupled method for simulation of fluid-structure interaction problems. *Journal of Fluids and Structures*, v. 50, p. 497 – 511, 2014. ISSN 0889-9746. Available at: <http://www.sciencedirect.com/science/article/pii/S0889974614001595>.

IHMSEN, M.; ORTHMANN, J.; SOLENTHALER, B.; KOLB, A.; TESCHNER, M. Sph fluids in computer graphics. The Eurographics Association, 2014.

IKARI, H.; GOTOH, H. Parallelization of mps method for 3d wave analysis. In: *Advances in Hydro-science and Engineering, 8th International Conference on Hydro-science and Engineering (ICHE)*. [S.l.: s.n.], 2008.

INTEL Processor i7 4790 Specifications. 2013. Accessed: 2018-04-22. Available at: <https://ark.intel.com/products/80806/Intel-Core-i7-4790-Processor-8M-Cache-up-to-4_00-GHz>.

IRIBE, T.; FUJISAWA, T.; KOSHIZUKA, S. Reduction of communication in parallel computing of particle method for flow simulation of seaside areas. *Coastal Engineering Journal*, World Scientific, v. 52, n. 04, p. 287–304, 2010.

JOHNSON, A. A.; TEZDUYAR, T. E. Advanced mesh generation and update methods for 3d flow simulations. *Computational Mechanics*, Springer, v. 23, n. 2, p. 130–143, 1999.

KAWAHARA, T.; OKA, Y. Ex-vessel molten core solidification behavior by moving particle semi-implicit method. *Journal of Nuclear Science and Technology*, Taylor & Francis, v. 49, n. 12, p. 1156–1164, 2012.

KHAYYER, A. *Improved particle methods by refined models for free-surface fluid flows*. Phd Thesis (PhD Thesis) — Kyoto University, 2008.

KHAYYER, A.; GOTOH, H. Development of cmps method for accurate water-surface tracking in breaking waves. *Coastal Engineering Journal*, World Scientific, v. 50, n. 02, p. 179–207, 2008.

KHAYYER, A.; GOTOH, H. Modified moving particle semi-implicit methods for the prediction of 2d wave impact pressure. *Coastal Engineering*, Elsevier, v. 56, n. 4, p. 419–440, 2009.

KHAYYER, A.; GOTOH, H. A higher order laplacian model for enhancement and stabilization of pressure calculation by the mps method. *Applied Ocean Research*, Elsevier, v. 32, n. 1, p. 124–131, 2010.

KHAYYER, A.; GOTOH, H. Enhancement of stability and accuracy of the moving particle semi-implicit method. *Journal of Computational Physics*, Elsevier, v. 230, n. 8, p. 3093–3118, 2011.

KHAYYER, A.; GOTOH, H. A 3d higher order laplacian model for enhancement and stabilization of pressure calculation in 3d mps-based simulations. *Applied Ocean Research*, Elsevier, v. 37, p. 120–126, 2012.

KHAYYER, A.; GOTOH, H. Enhancement of performance and stability of mps mesh-free particle method for multiphase flows characterized by high density ratios. *Journal of Computational Physics*, Elsevier, v. 242, p. 211–233, 2013.

KIM, K.; RAJAMANICKAM, S.; STELLE, G.; EDWARDS, H. C.; OLIVIER, S. L. Task parallel incomplete cholesky factorization using 2d partitioned-block layout. *CoRR*, abs/1601.05871, 2016. Available at: <http://arxiv.org/abs/1601.05871>.

KONDO, M.; SUTO, K.; SAKAI, M.; KOSHIZUKA, S. Incompressible free surface flow analysis using moving particle semi-implicit method. *Joint International Workshop: Nuclear Technology and Society – Needs for Next Generation, Berkeley, California, January 6-8, 2008, Berkeley Faculty Club, UC Berkeley Campus*, 2008.

KOSHIZUKA, S. A particle method for incompressible viscous flow with fluid fragmentation. *Comput. Fluid Dynamics J.*, v. 4, p. 29–46, 1995.

KOSHIZUKA, S.; NOBE, A.; OKA, Y. Numerical analysis of breaking waves using the moving particle semi-implicit method. *International Journal for Numerical Methods in Fluids*, Citeseer, v. 26, n. 7, p. 751–769, 1998.

KOSHIZUKA, S.; OKA, Y. Moving-particle semi-implicit method for fragmentation of incompressible fluid. *Nuclear science and engineering*, American Nuclear Society, v. 123, n. 3, p. 421–434, 1996.

KOSHIZUKA, S.; TAMAKO, H.; OKA, Y. A particle method for incompressible viscous flow with fluid fragmentation. *Comput. Fluid Dynamics J.*, 1995.

LEE, B.-H.; PARK, J.-C.; KIM, M.-H.; HWANG, S.-C. Step-by-step improvement of mps method in simulating violent free-surface motions and impact-loads. *Computer methods in applied mechanics and engineering*, Elsevier, v. 200, n. 9, p. 1113–1125, 2011.

LI, S.; LI, H. Parallel amr code for compressible mhd or hd equations. *Los Alamos National Laboratory*, 2006.

LUCY, L. B. A numerical approach to the testing of the fission hypothesis. *The astronomical journal*, v. 82, p. 1013–1024, 1977.

MARTIN, J. C.; MOYCE, W. J.; MARTIN, J. C.; MOYCE, W. J.; PENNEY, W. G.; S., F. R.; PRICE, A. T.; THORNHILL, C. K. Part iv. an experimental study of the collapse of liquid columns on a rigid horizontal plane. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, The Royal Society, v. 244, n. 882, p. 312–324, 1952. ISSN 0080-4614. Available at: <http://rsta.royalsocietypublishing.org/content/244/882/312>.

MICROSOFT. *Visual Studio Enterprise 2015*. 2018. Accessed: 2018-05-01. Available at: <https://www.visualstudio.com/>.

MONAGHAN, J. Sph without a tensile instability. *Journal of Computational Physics*, v. 159, n. 2, p. 290 – 311, 2000. ISSN 0021-9991. Available at: <http://www.sciencedirect.com/science/article/pii/S0021999100964398>.

MONAGHAN, J. J. Simulating free surface flows with sph. *Journal of computational physics*, Elsevier, v. 110, n. 2, p. 399–406, 1994.

MONAGHAN, J. J. Smoothed particle hydrodynamics. *Reports on Progress in Physics*, v. 68, n. 8, p. 1703, 2005. Available at: <http://stacks.iop.org/0034-4885/68/i=8/a=R01>.

MüLLER, M.; CHARYPAR, D.; GROSS, M. Particle-based fluid simulation for interactive applications. In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003. (SCA '03), p. 154–159. ISBN 1-58113-659-5. Available at: <http://dl.acm.org/citation.cfm?id=846276.846298>.

MUSTARI, A. P. A.; OKA, Y.; FURUYA, M.; TAKEO, W.; CHEN, R. 3d simulation of eutectic interaction of pb–sn system using moving particle semi-implicit (mps) method. *Annals of Nuclear Energy*, Elsevier, v. 81, p. 26–33, 2015.

NVIDIA. *CUDA Zone | NVIDIA Developer*. 2007. Accessed: 2016-01-09. Available at: <https://developer.nvidia.com/cuda-zone>.

NVIDIA. *NVIDIA GPU Maximum operating temperature*. 2010. <http://nvidia.custhelp.com/app/answers/detail/a_id/2752/related/1>. Accessed: 2018-05-02.

NVIDIA. *NVIDIA Visual Profiler*. 2018. Accessed: 2018-05-01. Available at: <https://docs.nvidia.com/cuda/profiler-users-guide/index.html>.

NVIDIA GPU GeForce GTX 760 Specifications. 2013. Accessed: 2018-04-22. Available at: <http://www.geforce.com/hardware/desktop-gpus/geforce-gtx-760/specifications>.

OÑATE, E.; IDELSOHN, S. R.; PIN, F. D.; AUBRY, R. The particle finite element method—an overview. *International Journal of Computational Methods*, World Scientific, v. 1, n. 02, p. 267–307, 2004.

Prometech Software. *Particleworks*. 2014. Accessed: 2018-04-23. Available at: <https://www.prometechsoftware.com/particleworks.html>.

RAMASWAMY, B.; KAWAHARA, M. Lagrangian finite element analysis applied to viscous free surface fluid flow. *International Journal for Numerical Methods in Fluids*, Wiley Online Library, v. 7, n. 9, p. 953–984, 1987.

ROGALLO, R. S.; MOIN, P. Numerical simulation of turbulent flows. *Annual Review of Fluid Mechanics*, v. 16, n. 1, p. 99–137, 1984. Available at: <https://doi.org/10.1146/annurev.fl.16.010184.000531>.

SCHROEDER, W. J.; LORENSEN, B.; MARTIN, K. *The visualization toolkit*. [S.l.]: Kitware, 2004.

SELLE, A.; RASMUSSEN, N.; FEDKIW, R. A vortex particle method for smoke, water and explosions. In: *ACM SIGGRAPH 2005 Papers*. New York, NY, USA: ACM, 2005. (SIGGRAPH '05), p. 910–914. Available at: <http://doi.acm.org/10.1145/1186822.1073282>.

SHAKIBAEINIA, A. *MPARS - Mesh-free Particle Simulator*. 2012. Accessed: 2018-04-09. Available at: <http://web.uvic.ca/~shakiba/mpars/MPARS-H.htm>.

SHAKIBAEINIA, A.; JIN, Y. A weakly compressible mps method for modeling of open-boundary free-surface flow. *International Journal for Numerical Methods in Fluids*, Wiley Online Library, v. 63, n. 10, p. 1208–1232, 2010. Available at: <https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.2132>.

SHAKIBAEINIA, A.; JIN, Y.-C. Mps mesh-free particle method for multiphase flows. *Computer Methods in Applied Mechanics and Engineering*, v. 229-232, p. 13 – 26, 2012. ISSN 0045-7825. Available at: <http://www.sciencedirect.com/science/article/pii/S0045782512000898>.

SHAO, S.; GOTOH, H. Turbulence particle models for tracking free surfaces. *Journal of Hydraulic Research*, Taylor & Francis, v. 43, n. 3, p. 276–289, 2005. Available at: <https://doi.org/10.1080/00221680509500122>.

SHAO, S.; LO, E. Y. Incompressible sph method for simulating newtonian and non-newtonian flows with a free surface. *Advances in Water Resources*, v. 26, n. 7, p. 787 – 800, 2003. ISSN 0309-1708. Available at: <http://www.sciencedirect.com/science/article/pii/S0309170803000307>.

SHIBATA, K.; KOSHIZUKA, S. Numerical analysis of shipping water impact on a deck using a particle method. *Ocean Engineering*, Elsevier, v. 34, n. 3, p. 585–593, 2007.

SHIBATA, K.; KOSHIZUKA, S.; OKA, Y. Numerical analysis of jet breakup behavior using particle method. *Journal of nuclear science and technology*, Taylor & Francis, v. 41, n. 7, p. 715–722, 2004.

SHIBATA, K.; KOSHIZUKA, S.; SAKAI, M.; TANIZAWA, K. Lagrangian simulations of ship-wave interactions in rough seas. *Ocean Engineering*, v. 42, p. 13 – 25, 2012. ISSN 0029-8018. Available at: <http://www.sciencedirect.com/science/article/pii/S0029801812000315>.

SILVA, A. L. B. Vieira e; ALMEIDA, M. W.; BRITO, C. J.; TEICHRIEB, V.; BARBOSA, J. M.; SALHUA, C. A qualitative analysis of fluid simulation using a sph variation. In: *Proceedings of the Congress on Numerical Methods in Engineering*. [s.n.], 2015. Available at: <http://www.dem.ist.utl.pt/cmn2015/html/CD-Proceedings/PDF/Papers/CMN_2015_submission_289.pdf>.

SILVA, A. L. B. Vieira e; ALMEIDA, M. W. S.; BRITO, C.; TEICHRIEB, V. Improved meshless method for simulating incompressible fluids on GPU. In: *2017 19th Symposium on Virtual and Augmented Reality (SVR)*. [S.l.: s.n.], 2017. p. 297–308.

SILVA, A. L. B. Vieira e; ALMEIDA, M. W. S.; BRITO, C.; TEICHRIEB, V. Improved MPS method and its variations for simulating incompressible fluids on GPU. *Journal on 3D Interactive Systems*, (Submitted), (Submitted), 2018.

SMAGORINSKY, J. General circulation experiments with the primitive equations. *Monthly Weather Review*, v. 91, n. 3, p. 99–164, 1963. Available at: <https://doi.org/10.1175/1520-0493(1963)091<0099:GCEWTP>2.3.CO;2>.

STAROSZCZYK, R. Simulation of dam-break flow by a corrected smoothed particle hydrodynamics method. *Archives of Hydro-Engineering and Environmental Mechanics*, v. 57, n. 1, p. 61–79, 2010.

SUN, X.; SAKAI, M.; SHIBATA, K.; TOCHIGI, Y.; FUJIWARA, H. Numerical modeling on the discharged fluid flow from a glass melter by a lagrangian approach. *Nuclear Engineering and Design*, Elsevier, v. 248, p. 14–21, 2012.

SUZUKI, Y.; KOSHIZUKA, S.; OKA, Y. Hamiltonian moving-particle semi-implicit (hmps) method for incompressible fluid flows. *Computer Methods in Applied Mechanics and Engineering*, Elsevier, v. 196, n. 29, p. 2876–2894, 2007.

TANIGUCHI, D.; SATO, L.; CHENG, L. Explicit moving particle simulation method on gpu clusters. *Blucher Mech. Eng. Proc. 1*, v. 1, p. 1155, 2014.

TAYEBI, A.; JIN, Y. chung. Development of moving particle explicit (mpe) method for incompressible flows. *Computers & Fluids*, v. 117, p. 1 – 10, 2015. ISSN 0045-7930. Available at: <http://www.sciencedirect.com/science/article/pii/S0045793015001462>.

TECHPOWERUP. *GPU-Z Video card GPU Information Utility*. 2018. Accessed: 2018-05-01. Available at: <https://www.techpowerup.com/gpuz/>.

TOMPSON, J.; SCHLACHTER, K.; SPRECHMANN, P.; PERLIN, K. Accelerating eulerian fluid simulation with convolutional networks. *arXiv preprint arXiv:1607.03597*, 2016.

TSUKAMOTO, M. M.; NISHIMOTO, K.; ASANUMA, T. Development of particle method representing floating bodies with highly non-linear waves. In: *18th International Congress of Mechanical Engineering, COBEM*. [S.l.: s.n.], 2005.

WENDLAND, H. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in computational Mathematics*, Springer, v. 4, n. 1, p. 389–396, 1995.

XU, R.; STANSBY, P.; LAURENCE, D. Accuracy and stability in incompressible sph (isph) based on the projection method and a new approach. *Journal of Computational Physics*, Elsevier, v. 228, n. 18, p. 6703–6725, 2009.

YOUNGS, D. L. Numerical simulation of turbulent mixing by rayleigh-taylor instability. *Physica D: Nonlinear Phenomena*, v. 12, n. 1, p. 32 – 44, 1984. ISSN 0167-2789. Available at: <http://www.sciencedirect.com/science/article/pii/0167278984905128>.

ZHU, X.; CHENG, L.; LU, L.; TENG, B. Implementation of the moving particle semi-implicit method on gpu. *SCIENCE CHINA Physics, Mechanics & Astronomy*, Springer, v. 54, n. 3, p. 523–532, 2011.