UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

An adaptive Particle-In-Cell method for liquid simulation using RBF-FD

Rafael Umino Nakanishi

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

Rafael Umino Nakanishi

An adaptive Particle-In-Cell method for liquid simulation using RBF-FD

Thesis submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP – in accordance with the requirements of the Computer and Mathematical Sciences Graduate Program, for the degree of Doctor in Science. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Afonso Paiva Neto

USP – São Carlos
March 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Nakanishi, Rafael Umino
N163a An adaptive Particle-In-Cell method for liquid simulation using RBF-FD / Rafael Umino Nakanishi; orientador Afonso Paiva Neto Paiva. -- São Carlos, 2021.
72 p.

Tese (Doutorado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2021.

1. Radial Basis Function. 2. Computational Fluid Dynamics. 3. Liquid Animation. I. Paiva, Afonso Paiva Neto, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Rafael Umino Nakanishi

Um método Particle-In-Cell adaptativo para simulação de líquidos utilizando RBF-FD

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Paiva, A.: Prof. Dr. Afonso Paiva Neto

USP – São Carlos
Março de 2021

ACKNOWLEDGEMENTS

Os agradecimentos principais são direcionados à minha família, meus pais e minha irmã, amigos e todos aqueles que contribuíram para o desenvolvimento desse trabalho. Em especial, Bruno, Kelly, Filipe, Camila e Matheus, pelo suporte técnico e emocional durante os longos anos de doutorado.

Agradecimentos especiais são direcionados ao Instituto de Ciências Matemáticas e de Computação, ao grupo de pesquisa VICG e ao LMACC. Agradecimentos aos técnicos responsáveis pelos equipamentos do laboratório, Leonardo e Gabriel, e ao meu orientador, Afonso, pelo suporte e auxílio durante o desenvolvimento do projeto dessa tese.

Agradeço à CAPES pelo auxílio financeiro e ao CNPq pelo financiamento do período sanduíche realizado no Japão sob orientação do professor Dr. Ryoichi Ando. Agradecimentos à equipe do ICMC, Aline, Carol e Marcos pela ajuda com os procedimentos do estágio sanduíche. Agradecimentos também à equipe do *National Institute of Informatics* pelo suporte e auxílio durante o período de estágio.

RESUMO

NAKANISHI, R. **Um método Particle-In-Cell adaptativo para simulação de líquidos utilizando RBF-FD**. 2021. 69 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Estruturas de dados adaptativas têm se tornado de interesse para processamento de grandes quantidades de informações, principalmente pela sua capacidade de se modelar automaticamente ao formato dos dados. Em se tratando de simulação de líquidos, introduzimos uma nova abordagem que combina o método de projeção da pressão com o método *Particle-In-Cell* (PIC). O método se baseia em uma versão generalizada do método de diferenças finitas (FD) para aproximar o campo de pressão e seus gradientes em discretizações de malha baseadas em árvore, possivelmente não balanceada. Em nossa abordagem, os pontos de amostra para diferenças finitas são usados para calcular as interpolações sem malha, fornecidas por uma variante da técnica de Funções de Base Radial (RBF), conhecida como *RBF-Finite-Difference* (RBF-FD). Esta versão sem malha produz pesos de diferenciação em nós espalhados com alta precisão. Nosso método adapta um *quadtree/octree* dinamicamente em uma faixa estreita ao redor da superfície do líquido, fornecendo uma amostra adaptativa de partículas para a etapa de advecção. Além disso, o RBF fornece um esquema preciso para transferência de velocidade entre a grade e as partículas, mantendo a estabilidade do sistema e evitando dissipação numérica. Apresentamos também uma estrutura de dados que conecta a subdivisão espacial de uma *quadtree/octree* com a topologia de seu grafo dual correspondente. A estrutura de dados apresentada torna a configuração *stencils* simples, permitindo sua atualização sem a necessidade de reconstruí-los do zero a cada passo de tempo. Os resultados obtidos mostram a eficácia e a precisão de nosso solver ao simulat fluidos invíscidos incompressíveis e comparando os resultados com métodos que utilizam malha regular, também baseados em PIC, disponíveis na literatura.

Palavras-chave: Animação compucional, simulação de fluidos, RBF, adaptive.

ABSTRACT

NAKANISHI, R. **An adaptive Particle-In-Cell method for liquid simulation using RBF-FD**. 2021. 69 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Adaptive data structures have become of interest for processing large amounts of information, mainly due to their ability to automatically model the data format. We introduced a new approach in the liquid simulation that combines the pressure projection method with the Particle-In-Cell (PIC) method. The solver relies on a generalized version of the Finite Difference (FD) method to approximate the pressure field and its gradients in tree-based grid discretizations, possibly non-graded. In our approach, FD stencils are computed using mesh-free interpolations provided by a variant of Radial Basis Function (RBF), known as RBF-Finite-Difference (RBF-FD). This mesh-free version of the FD produces differentiation weights on scattered nodes with high-order accuracy. Our method adapts a quadtree/octree dynamically in a narrow-band around the liquid interface, providing an adaptive particle sampling for the PIC advection step. Furthermore, RBF affords an accurate scheme for velocity transfer between the grid and particles, keeping the system's stability and avoiding numerical dissipation. We also present a data structure that connects the spatial subdivision of a quadtree/octree with the topology of its corresponding dual-graph. Our data structure makes the setup of stencils straightforward, allowing its updating without the need to rebuild it from scratch at each time-step. We show our solver's effectiveness and accuracy by simulating incompressible inviscid fluids and comparing results with regular PIC-based solvers available in the literature.

Keywords: Computer animation, Fluid simulation, rbf, adaptive.

LIST OF FIGURES

LIST OF CHARTS

LIST OF TABLES

CONTENTS

CHAPTER

1

INTRODUCTION

Fluid simulation is a broad study field in numerical simulation of physical phenomena, interesting for the Computer Graphics community with notable animations being produced in recent years (STOMAKHIN *et al.*, 2013; STOMAKHIN; SELLE, 2017). From small-scale raindrops to large-scale tsunami scenes, FX artists and directors strive for realistic visual effects capable of reproducing liquids' intricate motion in different scales, motivating the search for efficient fluid simulation methods. The most exciting methods, in this case, are spatially adaptive methods over the computational domain to reduce the computational burden and the memory load on empty regions (i.e., without fluid) and regions far from the liquid surface. Since the first use of the Navier-Stokes equations by Foster and Metaxas (1996), many researchers have developed a variety of techniques to reproduce realistic and computationally efficient fluid behavior, with a notable presence in animations and interactive graphics (Figure 1).

Fluid simulation is a complex problem to solve, and therefore there are still many open challenges to be adressed. Some examples in Computer Graphics to successfully tackle the problem include the Stable fluids (STAM, 1999), using a regular uniform grid to simulate the smoke, and level set fluids (ENRIGHT; MARSCHNER; FEDKIW, 2002; ENRIGHT *et al.*, 2003). These methods are preferred due to their capability to extract a symmetric positive definite pressure matrix, resulting in a stable linear system solution. These configurations' counterpart results from a poor advection of the values describing the fluid, like in the level sets case. A way to overcome this problem is using particle advection methods, known as Lagrangian methods, like Smoothed Particle Hydrodynamics methods or mesh surface methods. The downside is that Lagrangian formulations suffer from ensuring divergence free constraints due to the unstructured nature, making them less attractive to implement. Besides, methods with mesh representation (DA *et al.*, 2016) of the liquids add more complexity when dealing with topological changes. A common point on the methods listed above is the amount of generated data and the need for high computational processing power, raising the interest in reducing these costs.

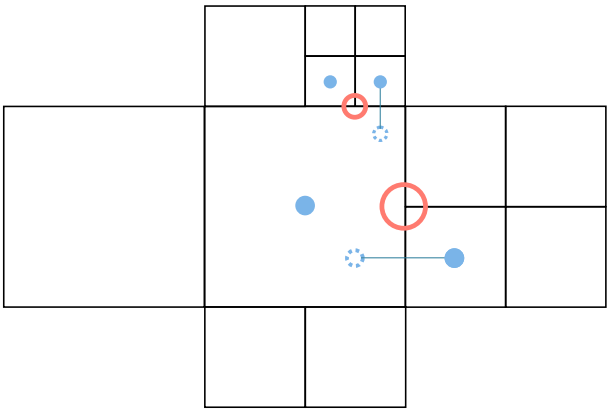
Adaptive meshes have become attractive in physically-based modeling, primarily for the

Figure 1 – Animation scenes involving special effects generated by computer algorithms simulating fluids realistic behavior. ©Disney

Source: Adapted from Stomakhin and Selle (2017).

pressure projection step in incompressible fluid flow simulations. Usually, in these simulations, the spatial adaptivity is achieved by replacing uniform grids with quadtree (in 2D) or octree (in 3D) grids (LOSASSO *et al.*, 2004; LOSASSO *et al.*, 2006). Tree-based grids combine the best of both worlds: the ability to use fast and compact Cartesian discretizations, such as finite differences (FD) and finite volumes (FV), and the versatility and accuracy of local mesh refinement. However, the refinement of quadtrees/octrees can produce non-conforming meshes containing undesirable T-junctions at regions where the mesh resolution changes (Figure 2). This configuration causes a severe limitation in traditional numerical methods because we cannot directly use the standard FD or FV stencils, causing the adaptive creation of the stencil closer to coarse-to-fine grid interfaces a delicate task, as it can ruin the entire simulation due to numerical instabilities that can occur in these regions. A way to overcome this problem is to build stencils using local high-order geometrical interpolations. These interpolations depend on the arrangement of the cells in the evaluation node's neighborhood and can become quite complicated in 3D simulations, mainly in staggered discretizations that store pressure samples at cell centers and velocity components at cell faces. Another alternative to computing generalized FD stencils is the use of meshfree approximations. This class of methods requires solving a small least-squares problem at the T-junctions to determine the suitable stencil.

Figure 2 – Grid resolution change on T-Junctions



Source: Elaborated by the author.

1.1 Thesis statement

This thesis's main idea is that differential operators can be efficiently computed on adaptive meshes using Radial Basis Functions with a sparse particle representation of the fluid, resulting in a hybrid adaptive method. Pressure and velocity samples can be computed the same way as in the standard regular grid methods over regular regions. Special treatment using RBF can compute their values correctly on T-junctions, either for interpolating valuer os approximating numerical derivatives. Although the resulting matrix is not symmetric, using the Biconjugate Gradients method over a smaller amount of data improves the overall simulation performance. Simulating fluid behavior concepts with fewer data and improved computational performance leads to efficiently creating more significant scenes' animation and saving simulation time considerably.

This thesis describes a staggered tree-based grid discretization, which provides a high-order approximation of the pressure field and its gradients even in non-graded quadtree/octree configurations, i.e., the difference of the resolution levels (depth) between adjacent cells is not constrained. Our method relies on meshfree interpolations computed via the Radial Basis Function (RBF) to enhance and generalize the FD method, providing differentiation weights on scattered node stencils in spatial discretizations. This meshfree version of the FD method is known as RBF-Finite-Difference (RBF-FD), and to our best knowledge, this is the first time RBF-FD has been used to simulate free-surface flows in the Computer Graphics and Computational Physics literature.

Additionally, we combine our robust and accurate tree-based pressure projection solver with the Particle-in-Cell (PIC) method straightforwardly and efficiently. Our fluid solver gracefully adapts a quadtree/octree dynamically during the simulation in a narrow-band around the liquid surface; this strategy also provides an adaptive particle sampling for the PIC advection step, drastically reducing the number of PIC particles inside the fluid without compromising the

visual details of the liquid surface. Moreover, the RBF interpolation gives a higher-order scheme for velocity transfer between the grid and particles. This procedure keeps the system's stability, avoids numerical dissipation even in coarse grid cells or regions with a low density of particles, and is visually competitive with Affine Particle-in-Cell (APIC). The figure shows our method in action.

In summary, the contributions of this thesis are:

- presenting a novel fully adaptive fluid solver that uses both tree-based grids and adaptive particle sampling using an approach which generalizes FD stencils in graded and non-graded grids;
- a pressure projection solver on adaptive grids that allows second-order accuracy for the pressure field and its gradients in infinity-norm;
- a stable and accurate scheme for PIC advection tailored to adaptive grids and non-uniform particle distribution;
- a tree-based data structure that can adapt itself along with the simulation, providing the stencil layouts automatically, i.e., without queries.

Also, the results of this thesis were published in the SIGGRAPH Asia 2020 conference (NAKANISHI *et al.*, 2020).

1.2 Thesis organization

The following chapters will present a more detailed introduction to the Navier-Stokes equations and how researchers have found their numerical solution, presenting spatial discretization and temporal integration concepts and their approach to solve the problem. We will also show how the equations' solution has been used in Computer Graphics for rendering realistic models. Next, we introduce the concepts of radial-based functions used for interpolation and their extension to compute finite differences values to solve partial differential equations numerically. In the following chapter, the method's core structure is described, showing how the structure can adapt itself and how it incorporates the RBF scheme to the staggered adaptive grid. Finally, we will summarize how the proposed method can be used and the published results of the SIGGRAPH Asia 2020 conference.

CHAPTER

2

RELATED WORKS

In this chapter, we are focusing on the Navier-Stokes equations concept introduction, as they mostly guide the fluid behavior during the flow. We describe here its origin from the momentum conservation equation, derived from physics laws. As it follows, we present the numerical methods for temporal-space discretization used in the literature for solving the equations.

2.1 Numerical simulation of the physics equations

It is widespread to use Physics equations to describe the behavior of different natural phenomena, like wind, climate, fluid motions, and fluid flow. For instance, we are more interested in how values, such as pressure and the fluid velocity, evolves from those fluid equations. This thesis focuses on the inviscid Navier Stokes equations, also known as Euler equations, usually written as

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \mathbf{f} \tag{2.1}$$

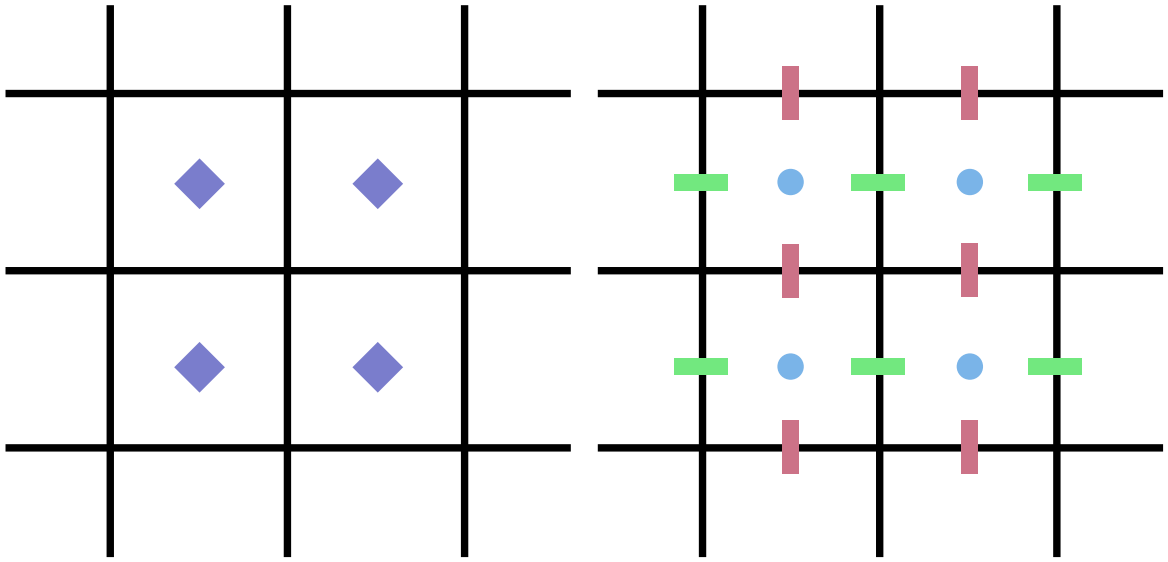
$$\nabla \cdot \mathbf{u} = 0, \tag{2.2}$$

where we have the momentum conservation and the mass conservation equation. The momentum equation's left-hand side refers to momentum exchange within the liquid due to local acceleration and convection. In contrast, the right-hand side is the external forces applied to them. The mass equation ensures that the liquid remains incompressible. In the equations, $\mathbf{u}$ is commonly used as the velocity vector; the greek letter ρ represents the fluid density; p is for pressure; $\mathbf{f}$ is the external forces applied to the fluid.

The analytical solution for these set of equations are still unknown, so the motivation for accurate numerical methods arises. We can find different approaches to tackle this problem in the literature, with their advantages and disadvantages. Usually, we consider that the fluid

Figure 3 – Structured grid models. Colored icons store simulation data.

(a) Collocated grid (b) *Mark-and-Cell*



Source: Elaborated by the author.

is a continuum body instead of directly simulating the molecules' behavior. We can find some techniques to solve the equations in literature: Eulerian, Lagrangian, and hybrid methods.

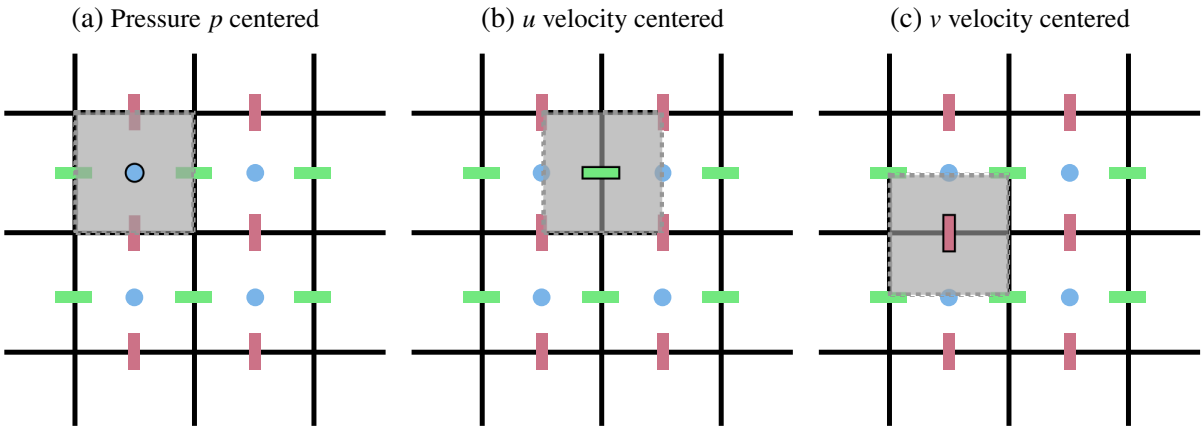
2.1.1 Eulerian methods

The Eulerian models, named after the Swiss mathematician Euler, solve the Navier-Stokes equations numerically through a fixed grid in the problem domain (STAM, 1965; HONG; KIM, 2003; LOSASSO *et al.*, 2006). The fixed grid points can rest on the cell center, cell vertices, or cell faces, and we measure how the fluid quantities (density, velocity, pressure, temperature) change over time. These grids may vary in shape and purpose, but the most common are the coupled grid and the *Mark-And-Cell* model presented by Harlow and Welch (1965). Notice in Figure 3a an example of spatial grid discretization.

The information needed for the simulation is stored in the mesh in two different locations: the pressure is in the cell center, denoted by p, while the speeds are stored in the face centers. Each velocity component, u, v, and w, is stored on the face corresponding to its coordinate, as shown by Figure 3b. This type of separation of the unknowns allows for a more stable solution of the Poisson equation. It is also necessary that each cell be categorized according to its content, whether it contains fluid, air, or solid, facilitating pressure calculation since only cells categorized as fluid are used to solve the equations. Although the MAC method can handle fluids' complex interface, it is computationally expensive due to every marker position storage and oscillations on the interface that may occur when the interface is reconstructed.(MCKEE *et al.*, 2008)

Another Eulerian method to solve the fluid simulation equation is the Level Set Method, which uses signed-distance functions to describe the free surface. This function is a piece of

Figure 4 – Control volume



Source: Elaborated by the author.

additional information stored on the grids' points and helps improve the Eulerian methods' advection problem. Due to the nature of the signed distance field, merging and splitting the surface is automatic, but it may lose information when reinitializing the Level Set function (SUSSMAN; FATEMI, 1999; PENG *et al.*, 1999; OLSSON; KREISS, 2005; OLSSON; KREISS; ZAHEDI, 2007). Several attempts on improving the reinitialization part of the method are being reported during the years, but they may never completely satisfy the mass conservation equation (PIJL *et al.*, 2005).

2.1.2 Lagrangean methods

Lagrangean models, named after the French mathematician Lagrange, are methods based on data structures that follow the fluid flow, like particle systems, connected or disconnected meshes. Fluid equation variables are stored on the vertices or particles that move as time progresses during the simulation. Standard methods that uses this approach are the *Smoothed Particle Hidrodynamics* (MONAGHAN, 1992) (MÜLLER; CHARYPAR; GROSS, 2003) (ADAMS; WICKE, 2009) and Surface-Only Fluids (DA *et al.*, 2016). The Lagrangian approach solves the advection problem found in the Eulerian models due to its nature. However, pressure solution becomes more difficult because there are unstructured data scattered throughout the space, and the discretization of differential values depends on interpolations and approximations, leaving the method potentially unstable, resulting in shorter time intervals to reduce instability.

Surface-Only Liquids (DA *et al.*, 2016) aims to simplify problems caused by excessive storage in volumetric data or grid data. In this way, all numerical values calculated and used during the simulation are stored at the liquid mesh's vertices, introducing the advantage of simplified calculation of variables linked to the fluid's surface, such as surface tension and magnetic fields (HUANG; MICHELS, 2020). In exchange for this advantage, this approach inserts difficulties in capturing vector fields' behavior inside the fluid and considers that such fields are irrotational, characteristics of inviscid flows. Besides, treatments to change the fluid

topology are also necessary since they are not directly implemented in the mesh representation.

On the other hand, particle-based methods are typically based on Smoothed Particle Hidrodynamics (SPH) (IHMSEN *et al.*, 2014), do not employ meshes, and store all the simulation data on the moving particles. The main advantage of this kind of algorithm is representing the free surface of the fluid more reliably. Also, as the particles are independent, parallelizing the processes makes it more computationally efficient. Besides that, the disadvantage relies on setting boundary conditions and solving the pressure system of the simulation. Also, increasing the resolution of the simulation means increasing the particle count significantly.

2.1.3 Hybrid methods

As presented, the Eulerian and Lagrangian models show individual features that could be improved if used together. With that in mind, the hybrid model *Particle In Cell*(PIC) (HARLOW, 1962) comes to mind. The PIC uses a structured mesh to perform the pressure calculations, while the advection of the fluid is done using particles, which store the velocities. As the Eulerian part of the model does not store fluid information, the velocity values must be interpolated on the cells' faces to solve the Poisson equation numerically. The velocities, then corrected, are interpolated back to the particles so that the advection is done.

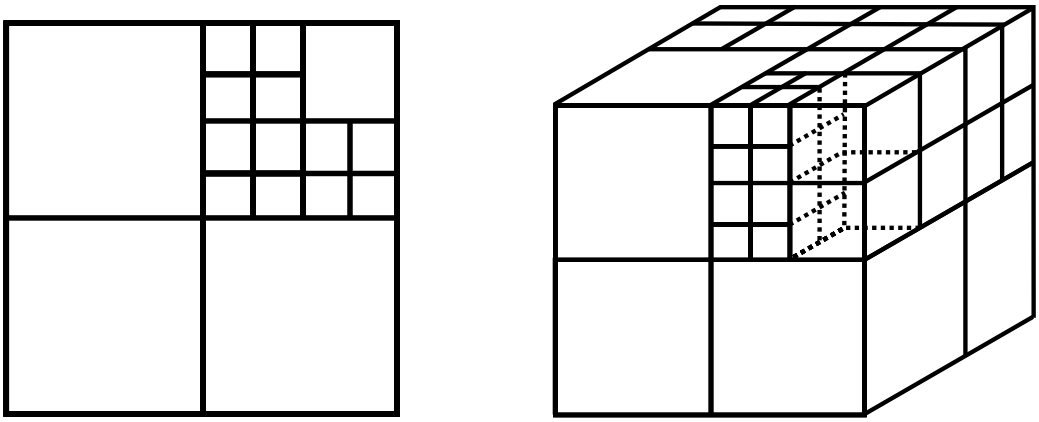
One of the algorithms' weaknesses is the numerical dissipation caused by the quantities interpolations between the grid and particles. In order to reduce losing much information, a correction weight is introduced in the Fluid Implicit Particles (FLIP) method to the Computer Graphics community Zhu and Bridson (2005) and has been one of the most used algorithms for realistic simulation of liquids, mainly by the entertainment industry (BUDSBERG *et al.*, 2013) (AVRAMOUSSIS; WARNER, 2015). The proposed approach is based on a ratio between the new velocities and the old velocities, so liquids' simulation has a smaller viscous effect (BRIDSON, 2007).

In the spirit of the Level Set Methods, (ENRIGHT *et al.*, 2002) has introduced a Particle Level Set Method, a combination of the Lagrangian surface tracking and the grid-based Level Set approach's efficiency. The method is based on correcting the Level Set values using escaping particles near the surface. The method's downside relies on choosing the particle reseeding approach, which may produce different results depending on the chosen algorithm.

2.2 Adaptive methods

The amount of fluid cells and particles drives the simulation's computational complexity in the previous methods presented. Over small domains, the complexity remains manageable, but as the problem increases, the cost increases exponentially due to the number of cells to process. Adaptive meshes have become attractive in physically-based modeling (MANTEAUX *et al.*, 2017), primarily for the *pressure projection* (STAM, 1999) step in incompressible fluid flow

Figure 5 – *quadtree*(2D) and *octree*(3D) adaptive structures



(a) Quadtree (b) Octree

Source: Elaborated by the author.

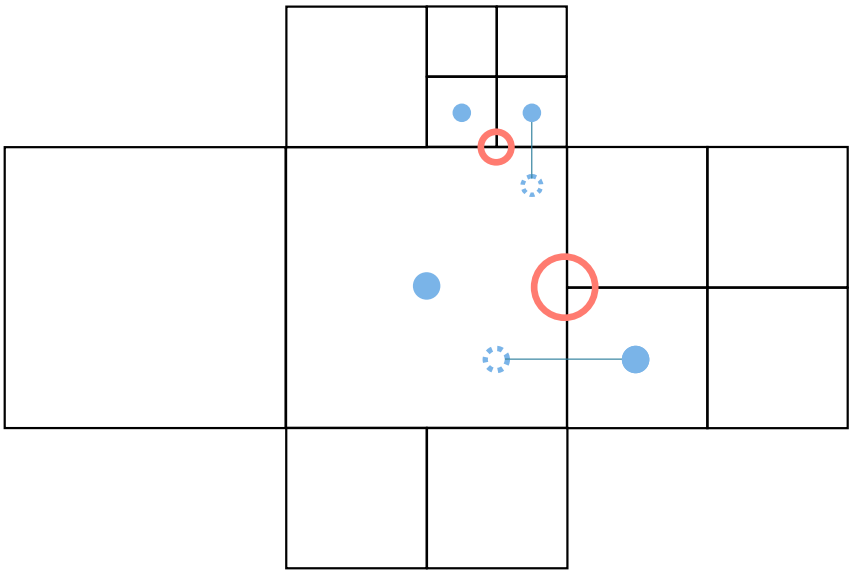
simulations. Usually, in these simulations, the spatial adaptivity is achieved by replacing uniform grids by quadtree (in 2D) or octree (in 3D) grids. Tree-based grids combine the best of both worlds: the ability to use fast and compact Cartesian discretizations, such as finite differences (FD) and finite volumes (FV), and the versatility and accuracy of local mesh refinement.

Octree structures are common used in Computer Graphics since it is capable of subdividing the domain region into octants according to some given parameter, as can be seen in Figure 5. Their discretization of the fluid flow equations are typically built around a staggered Finite Volume discretization combined with level set advection. However, the refinement of quadtrees/octrees can produce non-conforming meshes that contain the undesirable *T-junctions* at regions where the mesh resolution changes. This configuration causes a severe limitation in traditional numerical methods because we cannot directly use the standard FD or FV stencils, causing the adaptive creation of the stencil closer to coarse-to-fine grid interfaces a delicate task, as it can ruin the entire simulation due to numerical instabilities that can occur in these regions.

Losasso *et al.* (2004) introduced an innovative method combining the Particle Level Set Method and a first-order accurate pressure projection in L_∞ to simulate liquid and smoke on non-graded octrees. Although their method results in a symmetric positive definite (SPD) linear system, which can be efficiently solved by the Conjugate Gradient, the low order accuracy for the pressure can produce spurious velocity fields, resulting in instabilities near T-junctions (Figure 2). Later, Losasso *et al.* (2006) improved the previous method to achieve second-order accuracy for pressure projection while preserving the non-graded tree structure and the resulting SPD system.

Hong, House and Keyser (2009) combined Losasso's octree-based method of first-order of accuracy with Fluid Implicit Particle (FLIP) by adapting the cells and particle size within the simulation. This approach is similar to Ando, Thürey and Wojtan (2013)'s method, where a

Figure 6



Source: Elaborated by the author.

tetrahedral adapting mesh replaces the static grid. In the latter, the authors discretized the spatial representation of the fluid equations using Finite Volume Methods. As in extensive simulations, the number of particles becomes exceedingly large. The technique aims to decrease this quantity to minimize the simulation's processing time, maintaining quality. According to the authors, there is a considerable computational gain when using this approach. However, its application is not as direct, requiring adjustments in the narrow-band transition region with the fluid's interior. Furthermore, the simulation's grid is a regular MAC grid refined throughout the domain, which can still be a bottleneck for the simulation when solving pressure.

Ferstl, Westermann and Dick (2014) presented a multigrid solver devoted to fluid simulations on octree grids using the Finite Element Method (FEM). However, their method only deals with graded octrees, where the difference of depth between adjacent octree cells is constrained to one. As stated by Batty (2017), graded discretizations lead to an unwanted increase in the number of cells. In his approach, the solution is obtained by interpolating values over T-junctions through cells' diagonals. Thus balancing the octree is not necessary. However, the method makes the pressure system not symmetric anymore and introduces many non-trivial cases to be implemented, making it a difficult technique to implement. Another alternative to computing generalized FD stencils is the use of meshfree approximations (OLSHANSKII; TEREKHOV; VASSILEVSKI, 2013; SOUSA *et al.*, 2019). This class of methods requires solving a small least-squares problem at the T-junctions to determine the suitable stencil.

There are also adaptive methods that address only the particle sampling aspect in PIC/FLIP solvers. Ando, Thurey and Tsuruno (2012) proposed adaptive FLIP, which preserves thin sheets of fluid by inserting particles in poorly sampled regions. Recently, the narrow-band FLIP method (FERSTL *et al.*, 2016)(SATO *et al.*, 2018) was developed to alleviate the number

of particles in FLIP simulations using particles only near the liquid surface. However, they rely on a dense and uniform grid for the pressure projection.

Klingner *et al.* (2006) and Chentanez *et al.* (2007) highlighted how unstructured tetrahedral meshes are inherently adaptive by conforming to complex boundaries and allowing for differently-sized tetrahedra, enabling finer resolution where computational effort is needed. Since colliding objects and boundaries can be animated, the mesh has to be continuously regenerated. Batty, Xenos and Houston (2010) combined embedded boundary techniques with tetrahedral meshes to avoid the remeshing problem. Ando, Thürey and Wojtan (2013) proposed a second-order accurate liquid solver on adaptive tetrahedral meshes, which combines a variant of FEM with FLIP advection. However, their method drops to the first-order accuracy due to poorly shaped tetrahedra, which occur in coarse-to-fine resolution interfaces.

2.3 Summary

This chapter discussed many approaches to simulate numerical liquids adaptively. Our review focused on methods that adopted adaptive structures and methods based on hybrid models to achieve the final result. While adaptive grid methods proposed using octrees as part of their eulerian formulation, several do not show an easy way to solve the T-junction problems over the structure, causing issues on the final simulation. Our method especially proposes a straightforward and grid-independent way to compute the Pressure projection step on this kind of structure while maintaining the grid's adaptivity. Also, we propose the use of RBF interpolations over all the pipelines.

CHAPTER

3

FINITE DIFFERENCES USING RADIAL BASIS FUNCTIONS

When we talk about spatial discretization for solving partial differential equations, we commonly choose to use structured meshes (finite differences) or unstructured meshes (finite elements). However, these methods restrict the position of the points to specific locations to obtain the derivatives' values. An alternative to this problem was the emergence of pseudo-spectral methods, which use base functions to calculate the derivatives. Although they are very accurate, the computational cost for performing the calculations makes them unfeasible for numerical applications that require precision and short processing time. In this context, radial based functions have been used extensively in methods that have sparse points spread across the domain. The use of arbitrary points in the domain makes its implementation interesting for adaptive methods. It is unnecessary to maintain the configuration of points for the entire simulation length and may have a higher density of points close to regions of interest, such as borders.

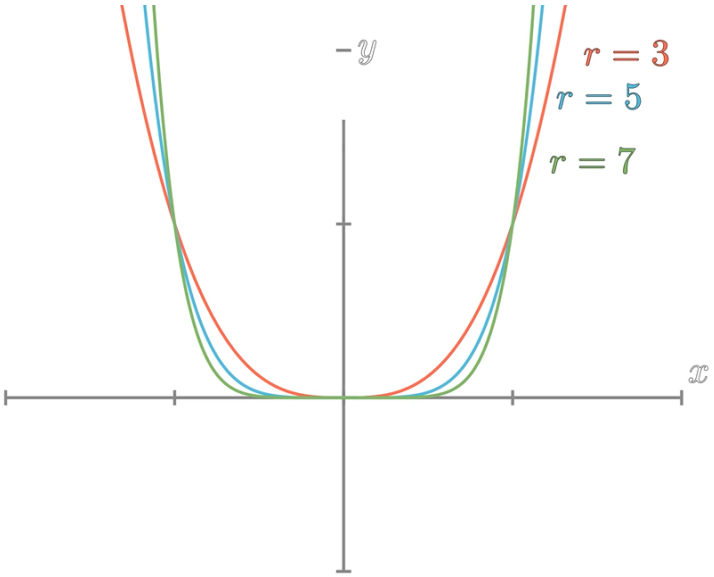
The use of radial basis functions for solving partial equations has been of great interest in the literature (FORNBERG; FLYER, 2015) and has shown promising results for simulating geophysical phenomena (TILLENIUS *et al.*, 2015). Besides, few studies focused on computer graphics using this method, such as simulations of free surfaces and turbulent fluids. However, the low numerical error obtained when using RBF Radial Basis Functions makes them of great interest in studying these applications. In Computer Graphics, RBFs are popular in geometric modeling applications, mainly in surface reconstruction from scattered *Hermite data* (surface points and its normals) (CARR *et al.*, 2001; TURK; O'BRIEN, 2002; OHTAKE; BELYAEV; SEIDEL, 2005; MACÊDO; GOIS; VELHO, 2011).

Since the RBF-FD formulas are derived from RBF interpolants, we review the basic concepts of RBF interpolation and then provide a brief introduction to RBF-FD.

Table 1 – Tipos de funções de base radial

Piecewise smooth functions	$\varphi(r)$
Polyharmonic splines	$r^m \quad m = 1,3,5,\cdots$
	$r^m \log(r) \quad m = 2,4,6,\cdots$
Compact support (Wendland)	$(1-\varepsilon)^m + p(\varepsilon r) \quad p$: Polynomial
Infinitely smooth functions	
Gaussian	$e^{-(\varepsilon r)^2}$
Multiquadric	$\sqrt{(1+(\varepsilon r)^2)}$
Inverse multiquadric	$\frac{1}{\sqrt{(1+(\varepsilon r)^2)}}$

Figure 7 – Polyharmonic RBF power variation, $r = 3,5,7$



3.1 Radial basis function (RBF)

An RBF is a radially symmetric function $\Phi_k : \Omega \subseteq \mathbb{R}^d \to \mathbb{R}$ which relies on the Euclidean distance between its center $\mathbf{x}_k$ and the evaluated point $\mathbf{x} = (x_1, \ldots, x_d)$, both in the domain Ω. Mathematically, an RBF can be represented by $\Phi_k(\mathbf{x}) = \phi(r)$ with $r = \|\mathbf{x} - \mathbf{x}_k\|$, where ϕ is a scalar function on $[0, \infty)$ and $\|\cdot\|$ denotes the Euclidean norm. There are many possible choices for $\phi(r)$, as can be seen in Table 1 (FASSHAUER, 2007). In particular, we use a *Polyharmonic Spline* (PHS) given by odd radial powers, due to its implementation simplicity:

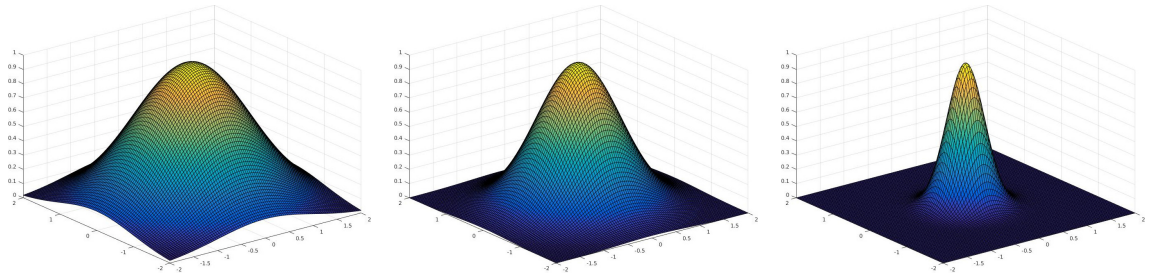
$$\phi(r) = r^m, \quad m = 1,3,5,\ldots$$

The power influence in the function can be analysed in the Figure 7.

Note that some functions have the parameter epsilon, which is called a *shape parameter*, responsible for controlling the RBF functions' radius of influence. Low values of epsilon, that is to say, large variance, cause the functions to take a flattened shape, while larger values lead to RBFs with peaks. The choice of the ideal value for the shape parameter is still a target study in

Figure 8 – *Shape parameter* variation on Gaussian RBF

(a) $\varepsilon = 0.5$ (b) $\varepsilon = 1.0$ (c) $\varepsilon = 5.0$



Source: Elaborated by the author.

the literature. Observe in Figure 8 the influence of the variation of shape parameter in a Gaussian RBF.

In a scenario where interpolations or approximations need more precision in some regions, the natural intuition is to reduce the spacing between points, increasing the resolution of the problem we want to solve. In classic finite differences, this approach can work well, from the intuition of derivatives. However, in some cases of RBF, where the shape parameter is present, this is not very effective, since a stagnation error (or saturated error) prevents increasing precision. In some cases, the error of the interpolation even increases as the spacing of the points decreases. For infinitely smooth RBFs (e.g., Gaussian and Multiquadric), for example, reducing the distance between points increases the singularity of matrices at some point. Consequently, the method demands increasing the shape parameter value to try to avoid singularities. This approach causes the final matrix to have a better condition number, but the generated system is nearly identical to the more refined resolution system.

In contrast to infinitely smooth RBFs, PHS's main advantage is that they do not require tuning a shape parameter, avoiding a laborious work and research for finding a suitable value (FLYER *et al.*, 2016). The polyharmonic functions are also suitable for applying this approach since they will significantly gain the space generated together with the polynomials. Moreover, this combination has information on the interpolation points' location, which is essential for interpolation to be non-singular, according to the theorem from Mairhuber-Curtis (MAIRHUBER, 1956).

3.2 RBF interpolation with polynomials

The introduction of RBFs as a multivariate data interpolator popularized its use in different applications (FORNBERG; FLYER, 2015), such as geoclimate forecast and scattered data interpolation. The problem consists of finding an interpolating function whose error for unknown points is as small as possible. In a mathematical formulation, given a set of distinct

nodes $\{\mathbf{x}_k\}_{k=1}^N$, scattered across the domain $\Omega \in \mathbb{R}^N$, and known function values $y_k = y(\mathbf{x}_k) \in \mathbb{R}$, we want to find an interpolant $\mathscr{S}_y : \Omega \to \mathbb{R}$ such that

$$\mathscr{S}_y(\mathbf{x}_i) = y_i\,, \quad \forall i = 1,\ldots,N\,. \tag{3.1}$$

The general form of an RBF interpolant is given by:

$$\mathscr{S}_y(\mathbf{x}) = \sum_{k=1}^{N} \alpha_k\, \phi(\|\mathbf{x}-\mathbf{x}_k\|) + \sum_{j=1}^{M} \beta_j\, P_j(\mathbf{x})\,, \tag{3.2}$$

where $\{P_1(\mathbf{x}),\ldots,P_M(\mathbf{x})\}$ is a basis for the $M = \binom{m+d}{d}$-dimensional space Π_m^d of all d-variate polynomials with degree less than or equal to m. To guarantee uniqueness of the coefficients α_k and β_j, we need to enforce additional constraints:

$$\sum_{k=1}^{N} \alpha_k\, P_j(\mathbf{x}_k) = 0\,, \quad \forall j = 1,\ldots,M\,. \tag{3.3}$$

The constraints (3.1) and (3.3) result in the following linear system of order $N+M$:

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix}, \tag{3.4}$$

where the entries of the square matrix $\mathbf{A}$ (of order N) are given by $\mathbf{A}_{ij} = \phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$, $\mathbf{P}$ is a $N \times M$ matrix with entries $\mathbf{P}_{ij} = P_j(\mathbf{x}_i)$, $\mathbf{O}$ is a square zero matrix of order M, $\boldsymbol{\alpha} = [\alpha_1,\ldots,\alpha_N]^\top$, $\boldsymbol{\beta} = [\beta_1,\ldots,\beta_M]^\top$, $\mathbf{y} = [y_1,\ldots,y_N]^\top$ and $\mathbf{0}$ is a zero vector of length M. Furthermore, the block matrix of the system (3.4) is SPD (or negative definite), thus ensuring that the system has a unique solution.

An essential remark on RBFs with polynomial augmentation is the guarantee of polynomial precision, i.e., if $y \in \Pi_m^d$ then the set of function values $\{y_k\}_{k=1}^N$ with $N \geq M$ is fitted by $\mathscr{S}_y = y$, except for errors on the order of machine accuracy.

3.3 Finite Differences using RBF

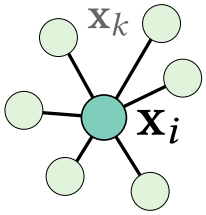
In this section, we derive RBF-FD formulas for approximating the set of differential equations that describe the behavior of liquids.

Let $\mathscr{L}$ be a linear differential operator (e.g., partial derivatives $\partial/\partial x_k$, Laplacian operator Δ) and the set $\mathfrak{X}_i$ be a stencil composed by the scattered nodes $\{\mathbf{x}_k\}_{k=1}^N \subseteq \Omega$, as showed in Figure 9a. For a given location $\mathbf{x}_i$ in the domain Ω (itself included in the stencil $\mathfrak{X}_i$ or not), we desire to approximate $\mathscr{L}y(\mathbf{x})$ evaluated at the center node $\mathbf{x}_i$ as a linear combination of the function values $\{y_k\}_{k=1}^N$ as follows:

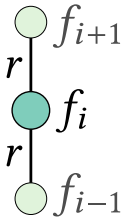
$$\mathscr{L}y(\mathbf{x}_i) = \sum_{k=1}^{N} \omega_k\, y_k\,. \tag{3.5}$$

Figure 9 – Representative schemes of RBF-FD samples

(a) Scattered scheme of points



(b) Comparing the RBF-FD with standard finite differences.



Source: Elaborated by the author.

The coefficients ω_k are known as *differentiation weights* and N is the *stencil size*.

Similar to the RBF interpolation, the weights ω_k are obtained by solving the following linear system:

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\gamma} \end{bmatrix} = \begin{bmatrix} \mathscr{L}\boldsymbol{\phi} \\ \mathscr{L}\mathbf{P} \end{bmatrix}, \tag{3.6}$$

with $\boldsymbol{\omega} = [\omega_1, \ldots, \omega_N]^\top$, $\boldsymbol{\gamma} \in \mathbb{R}^M$, $\mathscr{L}\mathbf{P} = [\mathscr{L}P_1(\mathbf{x}_i), \ldots, \mathscr{L}P_M(\mathbf{x}_i)]^\top$ and $\mathscr{L}\boldsymbol{\phi} = [\mathscr{L}\phi(\|\mathbf{x}_i - \mathbf{x}_1\|), \ldots, \mathscr{L}\phi(\|\mathbf{x}_i - \mathbf{x}_N\|)]^\top$. Since $\mathscr{L}$ operates on the right-hand side of Eqn. (3.6), the RBF-FD weights for the partial derivatives require the PHS derivatives. By the chain rule, we have:

$$\frac{\partial \phi}{\partial x_k}(r) = s x_k r^{s-2} \quad \text{and} \quad \frac{\partial^2 \phi}{\partial x_k^2}(r) = s r^{s-2} + s(s-2) x_k^2 r^{s-4},$$

After solving the linear system (3.6), the weights stored in $\boldsymbol{\gamma}$ are discarded. The approximation (3.5) is obtained by imposing that $\mathscr{L}y(\mathbf{x}_i)$ be exact for the interpolant $\mathscr{S}_y(\mathbf{x})$ with the constraints (3.1) and (3.3). Applying the operator $\mathscr{L}$ on Eqn. (3.2) and then evaluating at the point $\mathbf{x}_i$, we have

$$\begin{aligned}
\mathscr{L}\mathscr{S}_y(\mathbf{x}_i) &= \sum_{k=1}^{N} \alpha_k \mathscr{L}\phi(\|\mathbf{x}_i - \mathbf{x}_k\|) + \sum_{j=1}^{M} \beta_j \mathscr{L}P_j(\mathbf{x}_i) \\
&= \begin{bmatrix} \mathscr{L}\boldsymbol{\phi}^\top & \mathscr{L}\mathbf{P}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} \\
&= \begin{bmatrix} \mathscr{L}\boldsymbol{\phi}^\top & \mathscr{L}\mathbf{P}^\top \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{O} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{\omega}^\top & \boldsymbol{\gamma}^\top \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix} \\
&= \sum_{k=1}^{N} \omega_k y_k \,.
\end{aligned}$$

The transpose of $[\boldsymbol{\omega}^\top, \boldsymbol{\gamma}^\top]$ is the solution of the linear system (3.6). Notice that the weights stored in $\boldsymbol{\gamma}$ are ignored because their entries are multiplied by zero.

The choice of the PHS-polynomial basis is due to its many attractive properties, as reported by Flyer *et al.* (2016). Among these properties, we highlight:

- The approximations are shape parameter-free, and their interpolation matrix has an acceptable condition number;
- The accuracy of $\mathscr{S}_y$ and $\mathscr{L}y$ can be improved by increasing the polynomial basis and the PHS order;
- The convergence is dominated by the highest degree in Π_m^d.

3.3.1 Connection to standard FD

Regardless of the choice of the RBF, the standard FD method can be derived from RBF-FD when the stencil is parallel to coordinate axes, and its nodes are collinear, like in Figure 9b. In this case, polynomials in Π_1^1 are used to approximate partial derivatives in the direction x_k. For instance, given a scalar-valued function f and denoting $f(\mathbf{x}_i)$ by f_i, we want to compute $\partial f_i/\partial x_k \approx \omega_{i-1} f_{i-1} + \omega_{i+1} f_{i+1}$. The weights are the solution of the following linear system:

$$\begin{bmatrix} \phi(0) & \phi(2r) & 1 & (\mathbf{x}_{i-1})_k \\ \phi(2r) & \phi(0) & 1 & (\mathbf{x}_{i+1})_k \\ 1 & 1 & 0 & 0 \\ (\mathbf{x}_{i-1})_k & (\mathbf{x}_{i+1})_k & 0 & 0 \end{bmatrix} \begin{bmatrix} \omega_{i-1} \\ \omega_{i+1} \\ \gamma_{i-1} \\ \gamma_{i+1} \end{bmatrix} = \begin{bmatrix} \frac{\partial \phi}{\partial z}(r) \\ \frac{\partial \phi}{\partial z}(r) \\ 0 \\ 1 \end{bmatrix}$$

where $r = \|\mathbf{x}_{i+1} - \mathbf{x}_i\| = \|\mathbf{x}_{i-1} - \mathbf{x}_i\|$ and $(\cdot)_k$ is the k-th coordinate of a point. Thus, we have $\omega_{i+1} = 1/2r$ and $\omega_{i-1} = -1/2r$. Therefore,

$$\frac{\partial f_i}{\partial x_k} \approx \frac{f_{i+1} - f_{i-1}}{2r}. \tag{3.7}$$

3.3.2 Implementation

The weights can easily be computed in parallel architectures. To simplify the evaluation of $\mathscr{L}\mathbf{P}$ on the right-hand side of Eqn. (3.6), the stencil nodes of $\mathfrak{X}_i$ are translated such that the center $\mathbf{x}_i$ coincides with the origin in $\mathbb{R}^d$. Thus, the entries of $\mathscr{L}\mathbf{P}$ become all zero, except for the few entries equal to 1 or 2, depending on whether the derivative order of $\mathscr{L}$ is of the first or second order. However, switching an RBF center by the center node causes a change of sign in the first-order derivatives, because $\partial\phi/\partial x_k(\|-\mathbf{x}\|) = -\partial\phi/\partial x_k(\|\mathbf{x}\|)$.

CHAPTER

4

ADAPTIVE PIC SOLVER

In this chapter we describe each component of our fully adaptive PIC solver. Our primary objective is to solve the *Euler equations* for incompressible inviscid fluid flows, given by

$$\frac{D\mathbf{u}}{Dt} = -\frac{\nabla p}{\rho} + \mathbf{f} \quad \text{and} \quad \nabla \cdot \mathbf{u} = 0\,,$$

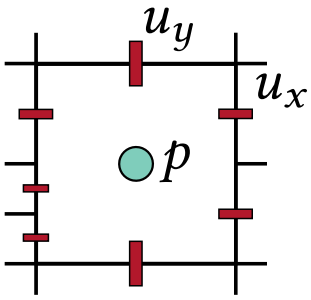
where $\mathbf{u}$ represents the fluid velocity; ρ the fluid density, which we assume to be constant and equal to 1; p is the inner fluid pressure; $\mathbf{f}$ is the accumulation of external forces to the fluid; and D/Dt denotes the material derivative.

PIC methods try to combine the benefits of both grid and meshfree discretizations: Lagrangian particles are used to advect all transported quantities, while the pressure projection scheme (STAM, 1999) is computed on an Eulerian grid. This scheme projects the velocities into a divergence-free space by solving the *pressure Poisson equation* (PPE): $\Delta p = \delta t\, \rho^{-1} \nabla \cdot \mathbf{u}^*$, where δt denotes the time-step and $\mathbf{u}^*$ an intermediate velocity after the advection. Then, the velocities are updated by $\mathbf{u} = \mathbf{u}^* - \delta t\, \rho^{-1} \nabla p$. In Sec. 4.2, we provide discretizations of these differential operators on adaptive grids.

4.1 Spatial discretization and adaptivity

Our spatial discretization is based on staggered quadtrees/octrees that store velocity components at cell faces (in this section, we use the term *faces* to refer to both faces, in 3D, and edges, in 2D) and pressure samples at cell centers, similar to the staggered grid introduced with the Marker-And-Cell (MAC) method (MCKEE *et al.*, 2008). This discretization has more degrees of freedom for velocity quantities than a usual collocated regular grid since they are stored into cell faces of an adaptive grid (where T-junctions may occur). Furthermore, in the MAC method, a grid cell is labeled *empty* if it contains no particles in its interior. Otherwise, it is labeled as a *fluid cell*.

Figure 10 – Staggered grid on a quadtree grid.



Source: Elaborated by the author.

Tree-based structures provide an efficient way to decompose the domain adaptively with a reduced number of resulting cells when compared to other adaptive structures, like tetrahedral meshes. The smaller the number of cells, the smaller the PPE system's size to be solved at each time-step of the simulation.

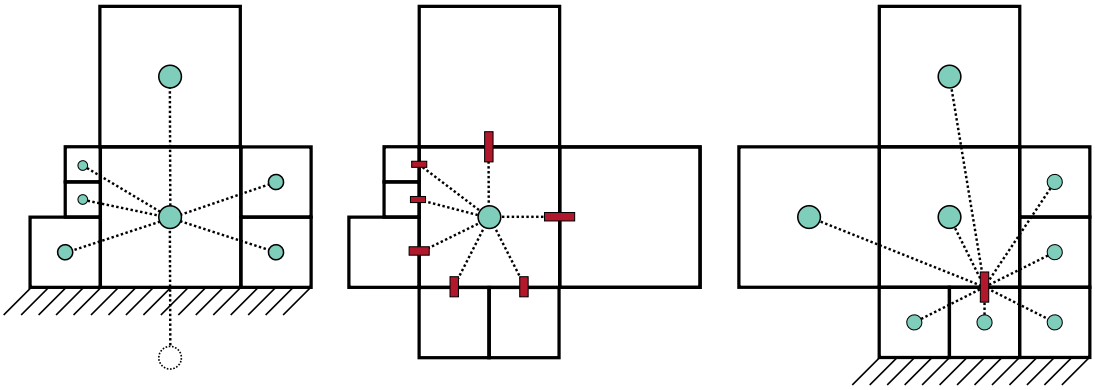
The adaptivity is lead by the liquid regions, which demand higher numerical accuracy and better representation of the visual details, such as the liquid surface and liquid-solid interfaces. For each time-step, our adaptive grid splits and/or merges cells when needed, keeping a locally-refined grid in a narrow-band around the liquid surface and liquid-solid interfaces, since higher resolutions capture sharper and smaller features of the free-surface and accurate solid-liquid interactions that may disappear when using coarser cells. Also, a region of finer cells is kept at the air (empty) cells near the surface, to avoid that particles change abruptly from finer to coarser cells.

Fluid cells far from the free-surface are coarser than *surface cells* (i.e., fluid cells whose 1-ring cell neighborhood contains at least one cell that is not a fluid cell), since no visual details are obtained from deep inside the liquid and the RBF interpolation does not dissipate velocity quantities at coarse cells.

4.2 Discretization of the differential operators

Before computing the RBF-FD weights of each differential operator by solving the system (3.6), we need to build the stencil layouts based on a MAC grid discretization. Given a fluid cell C^F, the stencil layout of the *Laplacian* approximation for pressure at the center of C^F is formed by the cell centers of its adjacent face cells in the k-ring cell neighborhood. We initially create the stencils taking $k = 1$, as shown in Figure 11. This strategy allows us to increase the stencil size, just by increasing k, improving the accuracy of the discretization (BAYONA *et al.*, 2017). Special treatment is needed when C^F is adjacent to a *solid cell* (i.e., a cell containing at least part of a wall or another obstacle); in this case, we create *ghost nodes* outside of the domain Ω containing liquid by reflecting the center of C^F across the boundary $\partial\Omega$ to enforce boundary

Figure 11 – RBF-FD stencil layouts from 1-ring neighborhood: Laplacian (*left*), divergence (*middle*), and gradient (*right*).



Source: Elaborated by the author.

conditions (Sec. 4.3). These weights are used to assemble a sparse Laplacian matrix associated with the resulting PPE system. The resulting Laplacian matrix from RBF-FD is not symmetric due to the weights derived from distinct arrangements of the stencils. This phenomenon may occur even when the topology of stencils are the same since the distances between neighboring cells may vary, causing such an asymmetry.

Regarding the stencil setup for the first-order partial derivatives, the stencil layout associated with *divergence* of the velocity is also computed at the center of C^F using the velocity components stored in adjacent faces of its k-ring neighborhood. The stencil of the pressure *gradient* components is centered on the faces of C^F; if the cells that share the same face have different resolutions, the remaining nodes are taken from the centers of the adjacent face cells in their k-ring neighborhood. Notice that if the cells and faces involved in the approximation of the differential operators have the same resolution (i.e., without T-junctions), we can directly apply the standard FD instead of RBF-FD.

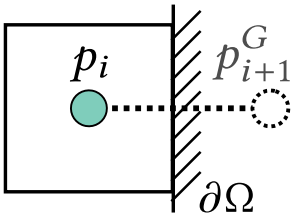
4.3 Boundary conditions

Similarly to the standard FD, to determine pressure values p^G at ghost nodes placed outside of the domain Ω, we need to impose the homogeneous Neumann boundary condition $\nabla p \cdot \mathbf{n} = 0, \forall p \in \partial\Omega$, where $\mathbf{n}$ is the unit normal vector pointing out of $\partial\Omega$. Discretizing Neumann boundary condition with RBF-FD, by Eqn. (3.7), we have:

$$0 = \nabla p \cdot \mathbf{n} = \frac{\partial p}{\partial x_i} \approx \frac{p^G_{i+1} - p_i}{\|\mathbf{x}^G_{i+1} - \mathbf{x}_i\|}.$$

Therefore, $p^G_{i+1} = p_i$. Hence, during the Laplacian matrix assembly, when a ghost node is present in the stencil, we add its weight to the center node weight, like in the standard FD scheme. Refer to Figure 12.

Figure 12 – Ghost point created for boundary computation



Source: Elaborated by the author.

For boundary conditions at solid walls, we impose the free-slip condition, i.e., velocity components normal to the walls are zero, while tangential velocities are equal to the fluid velocity. For the free-surface, we enforce Dirichlet boundary conditions, i.e., we set the pressures in air cells to zero. For implementation details, please see (KIM, 2016). For velocity boundary conditions at solid walls, we ensure the impermeability condition, i.e, $\mathbf{u} \cdot \mathbf{n} = 0$ and free slip condition, i.e., normal components of velocity in respect to walls are zero, while tangential velocities are equal to fluid velocity.

4.4 Particle reseeding

To keep the number of particles in our method low, we resample the particles when the number of particles in a cell is insufficient or when there are many more particles than necessary, regardless of the cell size. For each procedure, we use a user-defined threshold range to manage the number of particles inside a fluid cell C^F. If the number of particles of C^F is out of the threshold range, we delete its particles and create 2^d new sampled particles inside C^F using *Halton points* (FASSHAUER, 2007).

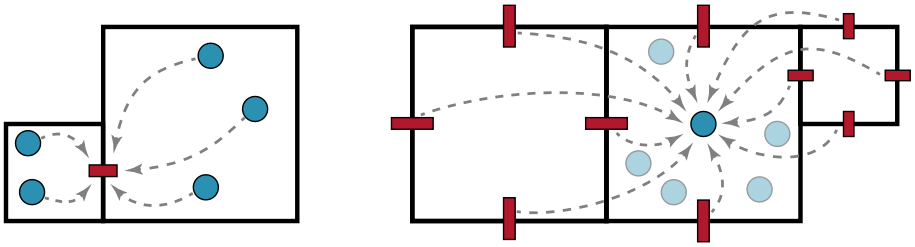
The reseeding is necessary to preserve fluid cells with particles, avoiding velocity transferences from particles that are far from interpolation centers. Our particle reseeding maintains low density of particles during the simulation without losing details on the surface or affecting the accuracy of our fluid solver. In our experiments, the number of particles varies in the threshold ranges $[3, 12]$ and $[5, 15]$ for 2D and 3D simulations, respectively.

4.5 Particle-grid transfers

Exchange of information between grid and particles is an essential step in PIC/FLIP advection. In our method, we transfer velocities between each structure using the RBF interpolation provided by Equation 3.2 and illustrated by Figure 13.

For transferring velocities from particles to a given cell face, we look for particles inside its adjacent cells. Thus, the velocity components of these particles are interpolated via RBF to the face center. Reciprocally, to obtain the velocity transfer from the grid to a given particle, we

Figure 13 – Velocity transfers between cell faces (*red*) and particles (*blue*): particle → grid (*left*) and grid → particle (*right*).



Source: Elaborated by the author.

take the velocity components of all faces of the cells incident to the cell which contains this particle. Then, we interpolate the velocities stored in the faces to the particles using RBF again. In particular, for the RBF interpolation, we use PHS of order $s = 5$ augmented with polynomials up to the 2nd degree, where the degree is governed by the number of particles or cell faces.

4.6 Our tree-based grid data structure

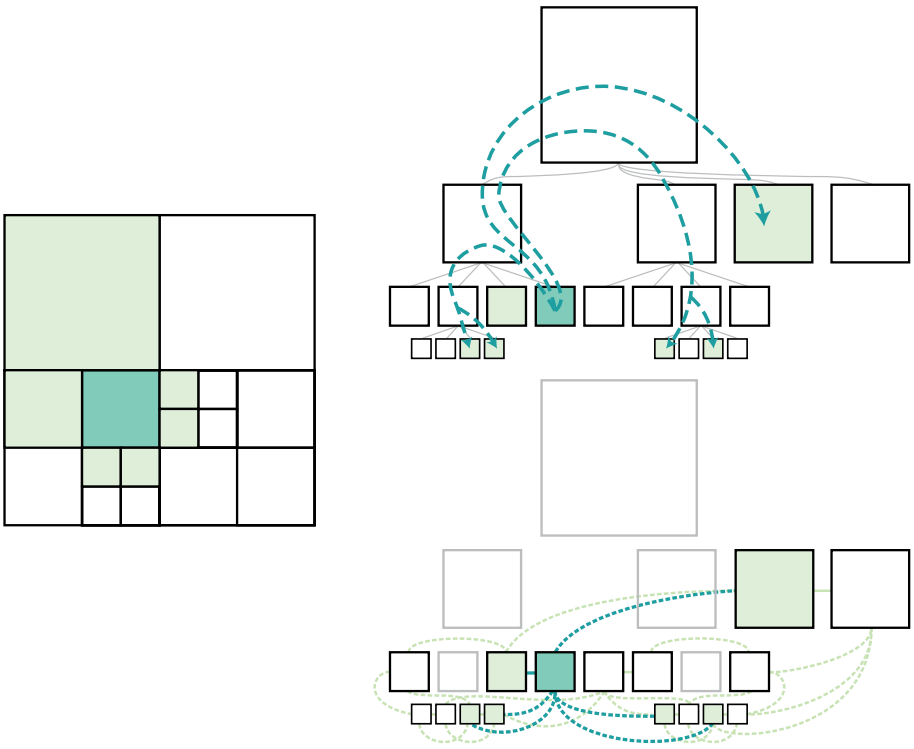
The implementation of a tree-based staggered grid as a classical octree, in 3D, or a quadtree, in 2D, is not trivial for many reasons:

- The data structure should deal with the tree faces which belong to the staggered grid since a (large) lower depth cell can have as neighbors multiple (small) higher-level cells in the adaptive grid;
- The computation of RBF-FD stencils resulting from different neighborhood configurations require the traversal of several levels in the tree (see Figure 14);
- As the simulation advances over time, the structure should be updated by splitting or merging leaf cells at each time-step. Otherwise, a new one should be rebuilt.

To address these issues, we consider the vertices and edges of the associated dual quadtree/octree, where each dual vertex corresponds to a single leaf cell, and each dual-edge corresponds to a unique primal tree face which is shared by two adjacent leaf cells. That set of dual edges provides the tree faces where the velocity components should be stored as well as the explicit topological representation required to construct the RBF-FD stencils.

In our framework, we introduce a data structure for adaptive staggered grids, called *Cellgraph*, which combines the geometry of the quadtree/octree and the topology of the corresponding dual-tree into a single graph structure. Each node of the graph is associated with a leaf cell, i.e., a dual vertex of the grid, and holds the axis-aligned bounding box (AABB) of the associated cell, while each edge of the graph is associated with a dual-edge. Such a structure

Figure 14 – A 2D stencil defined by a center cell and its direct neighbors. The stencil depicted in a quadtree with the center cell in dark green and its adjacent cells in light green (*left*). With a quadtree representation, the neighborhood calculation requires several tree traversals, drawn as dashed arrows (*top-right*). With a dual quadtree, the adjacency of neighboring cells is explicitly represented (*bottom-right*).



Source: Elaborated by the author.

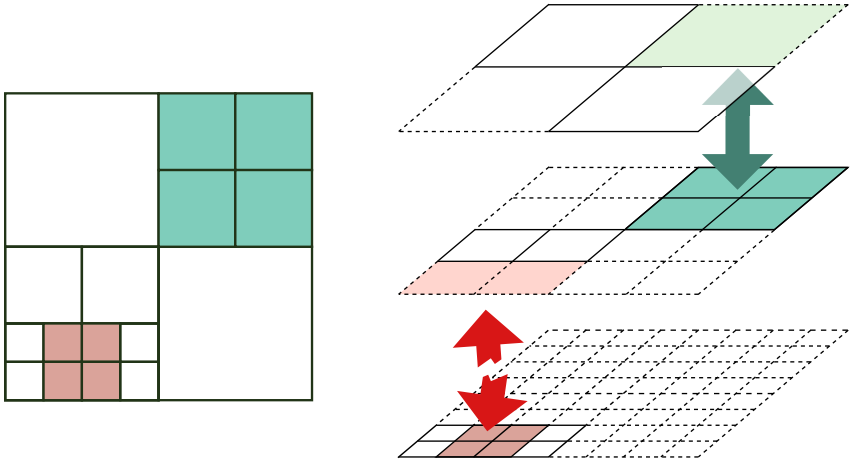
makes the computation of RBF-FD stencils straightforward, as adjacent leaf cells are identified explicitly by the graph edges, and larger stencils can be resolved with simple graph traversal procedures.

Different refinement levels can be achieved by splitting and/or merging nodes, allowing the graph to change its configuration from one simulation step to the next, without the need to construct an entirely new structure from scratch.

4.7 Cell refinement approach

The splitting of a leaf cell C of a quadtree/octree is reflected in the Cellgraph as follows: the node V of the graph associated with C is removed and then new nodes corresponding to the child cells of C, $\{C_i^\star\}$, $i = 1, \ldots, 2^d$, are added to the structure. The region given by the AABB of C is split following the quadtree/octree subdivision rules, and the resulting AABBs are properly assigned to the new graph nodes. All the edges adjacent to V are deleted, and new ones are created to represent, in the graph, the adjacency between each $C_i^\star$ and the 1-ring cell neighborhood of the parent cell C.

Figure 15 – The address code of a node can be used to check if a group of nodes (cells) can be merged together. On the left, a quadtree depicting two groups of cells, the green group can be merged, but not the red group. On the right, the two groups depicted in the resolution pyramid and their respective address codes on one level up.



Source: Elaborated by the author.

Merging a set of leaf cells $\{C_i^\star\}$ into a cell C is equivalent to removing the nodes associated with each $C_i^\star$ from the Cellgraph and then adding a new node V corresponding to the parent cell C. The AABB resulting from the union of the removed nodes is assigned to V. Similarly to splitting, all the edges adjacent to the removed nodes are deleted, and new ones are created to reflect the adjacency between C and its 1-ring cell neighborhood.

It is worth noting that the cells $\{C_i^\star\}$ can be merged if the cell C is their parent. Cellgraph checks if the cells associated with a group of nodes can be merged since each node stores its *level* and its *address code*. Two nodes are siblings and may have the same parent if they have the same level value and compatible address codes. The level of a node is the depth level of its associated cell in the quadtree/octree. The address code of a node represents the index of its associated cell at the same level as a *resolution pyramid* (i.e., a fully refined quadtree/octree). Two address codes are compatible if their indices are mapped to the same address code on a higher level of the pyramid (see Figure 15).

4.8 Surface tracking

The tracking of the free-surface controls the refinement level of our adaptive data structure. Before starting the simulation, we create a Cellgraph with only one node corresponding to the tree's root cell. The AABB of the initial node is set to cover the whole region of the computational domain, and the root cell is classified as a fluid cell. Next, we split each fluid cell until we reach the maximum depth of the tree, i.e., all fluid cells are fully refined. Then, we label the cells associated with the nodes of the resulting Cellgraph as empty, fluid, surface, or solid cells. Moreover, our setup scheme enforces that cells do not contain both particles and solids.

Finally, all surface nodes are added to a list.

In addition to the surface cell list, we define the cells belonging to a narrow-band around the liquid surface. While there is a cell C in the k-ring neighborhood of a surface cell C^S, if the level of C is greater than the level of C^S, then C is refined. Otherwise, C is considered to be in the narrow-band. All the cells with the same classification outside the narrow-band are merged up to the lowest level possible. In our experiments, we adopted $k = 2$.

At each time-step of the simulation, the surface cell list is updated to redefine the narrow-band. When updating the surface cell list at the current time-step, we detect new fluid cells and new air cells without the need to iterate over cells not belonging to the neighborhood of the surface cells defined at the previous time-step. One of the advantages of keeping the explicit list of surface cells is that it optimizes the task of cell material type classification. We can avoid checking the presence of particles in all cells based on the assumption that fluid cells become air cells and vice-versa only when changes happen in the list of surface nodes. This strategy also prevents the appearance of air cells inside the liquid or close to walls or obstacles.

CHAPTER

5

RESULTS

In the previous chapters, we introduced RBF concepts as a Finite Difference alternative for computing numerical derivatives and the dynamic structure used in conjunction in our simulation pipeline. Both approaches work adaptively to efficiently solve the Euler equations while using fewer data and storage space. Although the matrix's asymmetry may seem an issue of the method, we present results in this chapter that prove it wrong. Moreover, using RBF as an interpolator shows promising results on the unstructured octree grid, losing less information and interpolating data even on low sample regions, as opposed to the commonly used MLS method.

The next sections present the numerical studies on RBF interpolation on RBF-FD methods that support our hypothesis on using them on fluid simulation pipeline. Next, we deliver full 3D simulations using the proposed approach with high grid resolutions and details capturing. As shown, we compare the performances obtained with our adaptive method with standard methods used in the industry.

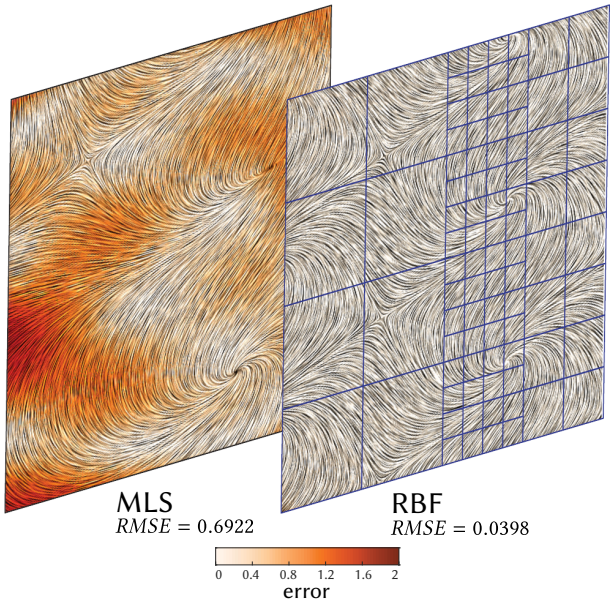
5.1 Numerical studies

The numerical studies refer to the RBF's isolated use as an interpolator and as a Finite Difference alternative. To support our hypothesis on using them on a liquid simulation pipeline, we imposed numerical problems with known solutions and checked the proposed method's performance. As we show in each subsection, RBF-FD presents satisfactory results compared to standard techniques used in the literature.

5.1.1 Particle-grid transfer

To demonstrate the accuracy of our transfer, as demonstrated on section 4.5, we perform a comparison against Moving Least Square (MLS) with a polynomial basis of Π_3^d as used by *Higher-Order Particle-in-Cell* (HOPIC) (EDWARDS; BRIDSON, 2012). Figure 16 shows RBF

Figure 16 – RBF and MLS approximations of a vector field sampled in a non-graded quadtree (blue).



Source: Elaborated by the author.

and MLS approximations of a vector field given by:

$$X(x,y) = (\cos(x+2y), \sin(x-2y)) \,, \quad (x,y) \in \Omega = [-2,2]^2 \,.$$

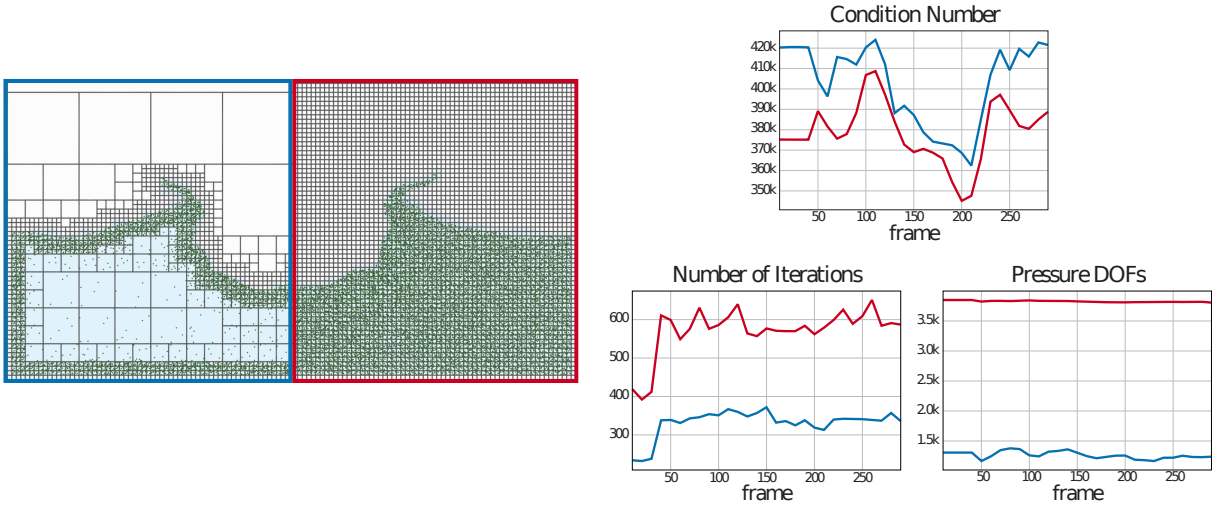
The approximated field is obtained sampling X at the midpoints of the edges of a quadtree discretization of Ω, splatting the sampled field onto a regular background grid of resolution 64^2. We can notice that RBF achieves higher accuracy regarding *root-mean-square error* (*RMSE*), preserving vector field singularities such as vortices and saddles even when using a polynomial space of a lower degree. This result motivated the RBF use as an interpolator since the fluid's interior has a coarser resolution and possibly may suffer from low sampling when using MLS.

5.1.2 Conditioning

When dealing with adaptive grids, a challenge is to guarantee that the resulting pressure system matrix will be symmetric positive-definite to have a real stable solution. Figure 17 shows a liquid drop simulation in a regular and adaptive grid discretization side by side. Our adaptive method decreases the computational efforts needed to solve the system, employing a non-graded grid with a difference of two resolution levels between adjacent cells. Our approach achieved similar results using 68% fewer fluid cells than the same simulation performed regularly grid with the same characteristic size.

To evaluate how the resulting PPE system is affected by the RBF interpolation, we compare the condition numbers of the PPE system matrices and the convergence of the BiCGSTAB (with relative tolerance set as the machine epsilon in double precision) for each simulation frame. The figure shows that although our non-graded discretization increases the condition number, it

Figure 17 – Liquid drop simulation (*leftmost*). Comparison between our non-graded quadtree (■) and a regular grid (■) with same characteristic size, $h = 2^{-7}$. From left to right, the condition number of the PPE matrix, the number of iterations of the linear system solver, and the number of pressure DOFs.



Source: Elaborated by the author.

does not affect the iterative linear solver's convergence. Meanwhile, the number of iterations of the linear system solver is less than the uniform case due to its reduced number of degrees of freedom (DOF).

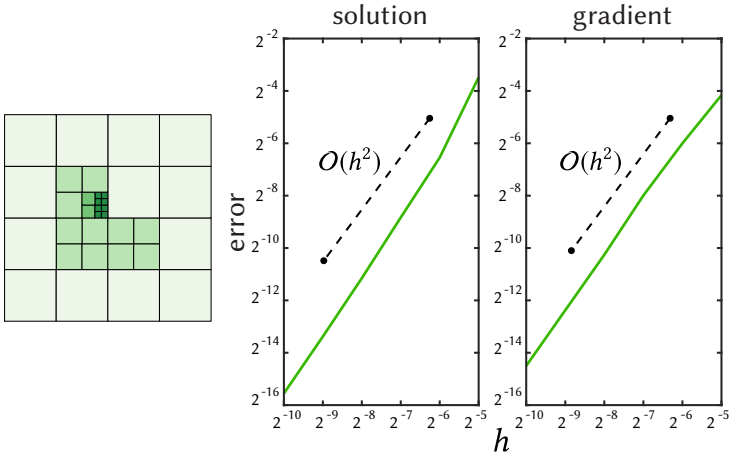
5.1.3 Convergence test

To numerically attest that our solver achieves a spatial accuracy of second-order in L_∞ norm, we consider 2D and 3D Poisson problems with homogeneous Neumann boundary condition in the grids depicted by Figure 18 and Figure 19. The exact solutions are $p(x,y) = \cos(x)\cos(y) - 1$ and $p(x,y,z) = \cos(x)\cos(y)\cos(z) - 1$ on $\Omega = [0,\pi]^d$, respectively. For the first iteration, we process the error obtained by the method and then recursively subdivide the grid while computing the L_∞ error for each refinement. The log-log plots (base 2) show second-order accuracy for the solution and its gradient using our approach. An important remark is that the resulting linear systems are singular. To avoid this problem, we need to impose a Dirichlet boundary condition at one center node of the domain to obtain a nonsingular system.

5.2 Liquid simulation

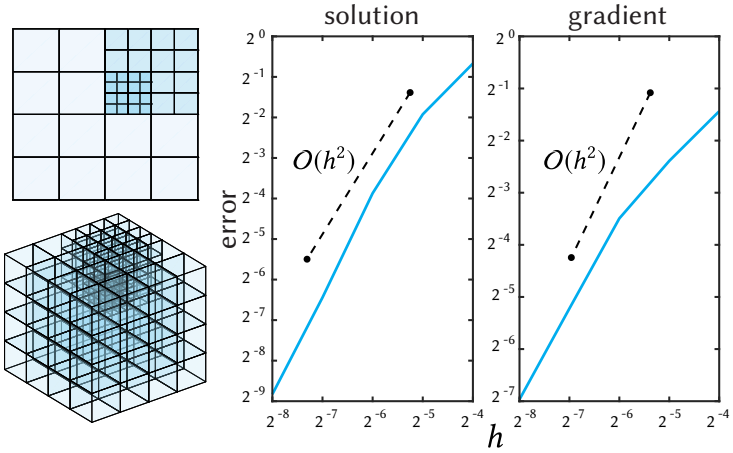
To demonstrate the effectiveness of our approach, we simulate incompressible inviscid liquids over different domains. In our experiments, we compute the RBF-FD stencils using PHS of order $s = 3$ and order $s = 5$ for the velocity transfers, both augmented with polynomials of Π_2^d. We use the `Eigen` library (GUENNEBAUD; JACOB *et al.*, 2010) to solve all linear systems required by our method since the library supports parallel implementations. Specifically,

Figure 18 – Initial non-graded quadtree for a Poisson problem (*left*). The log-log plots of the error in L_∞ norm varying with h (*right*). Dashed reference line indicates the second-order slope.



Source: Elaborated by the author.

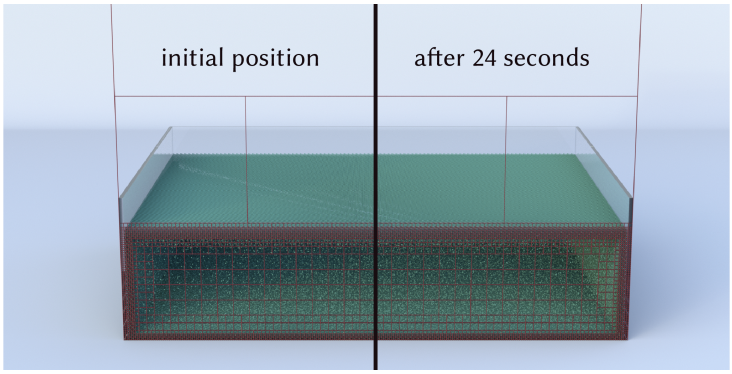
Figure 19 – Initial non-graded octree for a Poisson problem. A 3D view of the adaptive grid (*bottom-left*) and a cutting plane corresponding to $z = \pi/2$ (*top-left*). The log-log plots of the error in L_∞ norm varies with h (*right*). Dashed reference line indicates the second-order slope.



Source: Elaborated by the author.

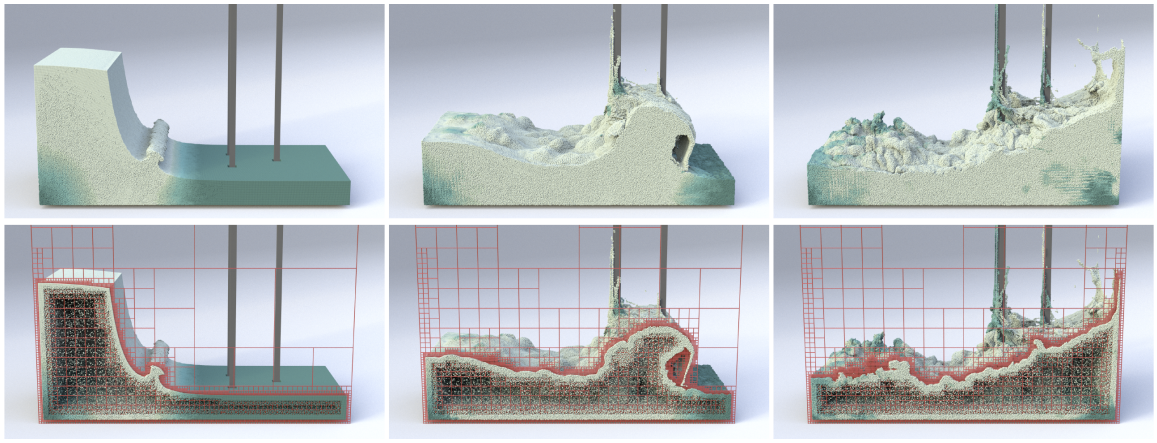
we use Householder QR factorization for the small RBF systems. For the large, sparse, and non-symmetric PPE system, we use Stabilized Biconjugate Gradient (BiCGSTAB) with Jacobi preconditioner (BARRETT *et al.*, 1994) in double precision and relative tolerance of 10^{-6}. Regarding the time integration, we use explicit Euler method with the time-step dictated by the CFL condition, when necessary. The rendered images and surface reconstruction were produced using SideFX Houdini. In our experiments, we employ up to five levels of refinement for fluid cells, although this is not a strict limitation of our method. Moreover, we denote the smallest *characteristic grid size* by h.

Figure 20 – Standing pool in an octree (red) with characteristic grid size $h = 2^{-8}$.



Source: Elaborated by the author.

Figure 21 – Simulation of a dam break with obstacles using an octree with a characteristic grid size of 2^{-8} and 4M PIC particles (*top*). Our method adapts the grid and particles around the free-surface, generating 68% fewer particles and 55% fewer fluid cells than regular PIC-based solvers. Besides, our adaptive pressure solver is at least twice faster than regular ones. A cutaway view shows our tree-based grid (red) and its underlying particle sampling (*bottom*). Lighter particle colors indicate fast velocities.



Source: Elaborated by the author.

5.2.1 Volume preservation

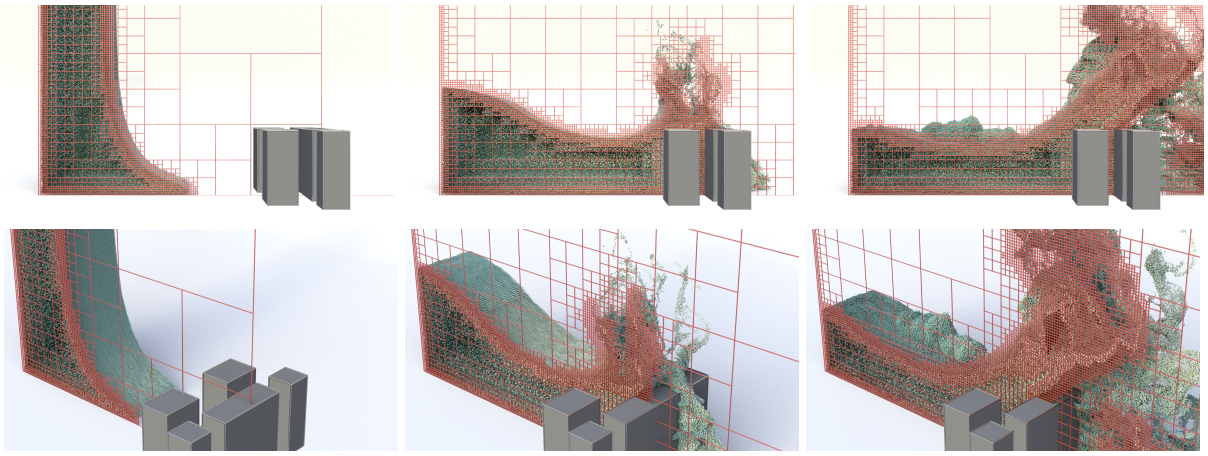
Figure 20 shows a hydrostatic standing pool simulation, which yields a linear pressure distribution. In this example, we imply increased resolution on the boundaries to provide an accurate pressure field, preventing error propagation, and consequently avoiding volume loss even after a long simulation period. The characteristic surface grid size used in the example is $h = 2^{-8}$ and $h = 2^{-6}$ inside the liquid, as show by in red in the figure.

5.2.2 Adaptivity

The Figure 21 shows a liquid splash from a dam break with obstacles to demonstrate our method's dynamic adaptation. According to the fluid behavior, the grid adapts, preserving details

Figure 22 – Simulation of a dam break with obstacles using an octree with a characteristic size of $10 \cdot 2^{-10}$. The first row shows a side view of the simulation with a cross-section of the adaptive grid. In the bottom row, the same timesteps are showed with a zoom on the cross-section.

(a) $t = 0.2s$ (b) $t = 0.5s$ (c) $t = 0.6s$



Source: Elaborated by the author.

on the surface with smaller cells and more particles while reducing their numbers inside. Overall, this approach reduces the total cell and particle count drastically (refer to Table 2). Another example of thin details being simulated can be seen in the Figure 22, with a grid characteristic size of $10 \cdot 2^{-10}$.

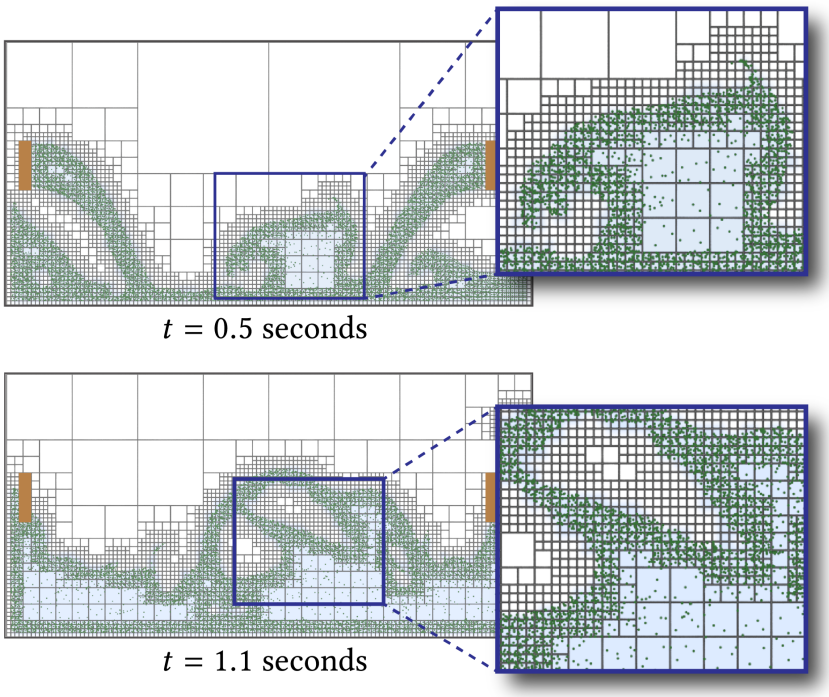
Analogously, the Figure 23 shows our method's dynamic adaptability in a 2D simulation. In this example, two sources are pouring a liquid into a container. As can be seen, our mesh-free approximation can efficiently handle non-graded grid discretizations, allowing for complex stencil configurations near T-junctions (zoomed section).

Figure 24 shows the impact of a drop on a liquid layer, the Cellgraph adapts the grid around the interface dynamically as the simulation progresses, tracking the topological changes of the free-surface gracefully. Our structure refines the fluid cells near the liquid surface and is coarse far away from it, without discarding any information of the previous time-step.

5.2.3 Comparisons against regular PIC-based solvers

Figure 26 shows the behavior of our adaptive RBF-based PIC compared to PIC, FLIP, and APIC (implementations provided by Kim (2016)). Our approach produces a smooth and well-behaved spreading of particles during a liquid splash around an S-shaped corridor's walls due to its accuracy and stability of velocity transfers between the grid and particles. As can be seen, our method preserves thin fluid sheets without particle clumping, even using 40% fewer particles than regular PIC-based solvers. Moreover, Figure 27 shows a plot of the kinetic energy over time. Our RBF-based PIC is more resilient to artificial dissipation without sacrificing stability, keeping the splashes "alive" throughout the simulation.

Figure 23 – Two sources (brown) pour a liquid in a container. Our RBF-based method can generate FD stencils in non-graded grids. The fluid cells are highlighted in light blue.



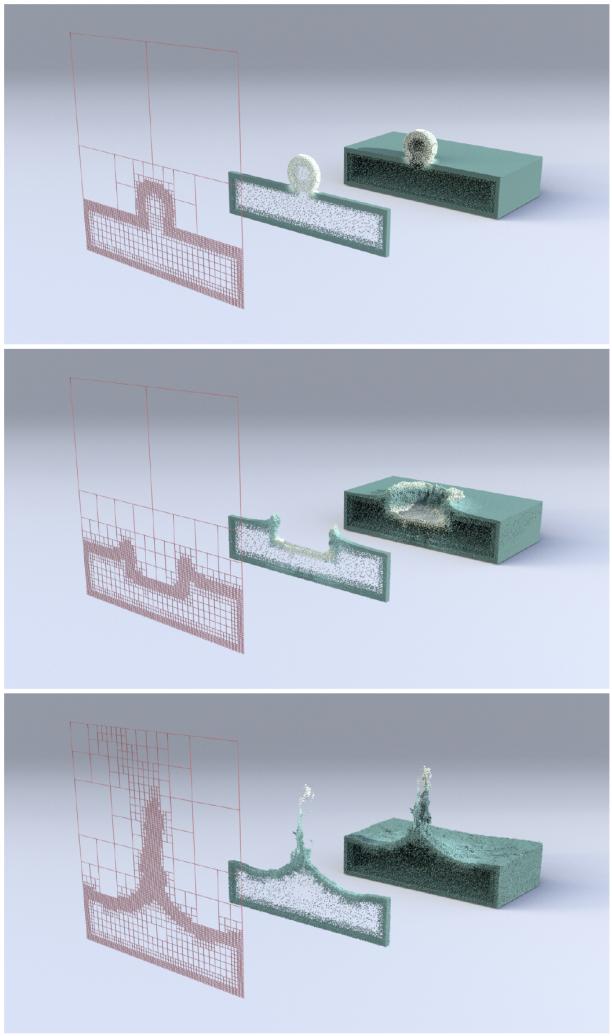
Source: Elaborated by the author.

Besides, the high-order accuracy presented by RBF interpolation and RBF-FD is attested by our experiments. Our method preserves vorticities over long periods, as can be seen in Figure 25. Due to the severe numerical diffusion of the bilinear/trilinear interpolations commonly present in regular PIC/FLIP simulations, the fine details of the flow vanish quickly. Although APIC produces comparable results for vorticity preservation, their technique is more complicated to implement and stores additional information per particle, such as the affine state matrix and velocity derivatives.

5.2.4 PPE linear system solver

We analyze the efficiency of our adaptive PPE solver using BiCGSTAB with a parallel Jacobi preconditioner in a dam break simulation with four water columns creating a huge splash, as illustrated by Figure 28. It is worth mentioning that there is a trade-off between the computational cost of building and applying the preconditioner and the convergence speed gain. Note that the PPE matrix size changes throughout the simulation, which requires the preconditioner to be re-computed at each iteration. Thus, we compare the PPE solver's performance when applied to the non-symmetric Laplacian matrix from our adaptive discretization and the SPD system arising from the regular PIC solvers. To eliminate external factors and guarantee a fair comparison, we apply a preconditioned Conjugate Gradient (PCG) for regular solvers, also implemented by `Eigen`, using the same tolerance of the BiCGSTAB and the same parallel preconditioner as well.

Figure 24 – Our Cellgraph dynamically controls the grid refinement/coarsening as well as the particle sampling in a narrow-band of the interface of a liquid drop without the need to build an entirely new grid from scratch.



Source: Elaborated by the author.

In this case, we observe that our solver is 2.6× faster than PCG with Jacobi. For more sophisticated preconditioners[1], we perform a comparison between our adaptive solver using BiCGSTAB preconditioned with ILU and the regular solver using PCG with incomplete Cholesky. For this experiment, we obtain a speed-up of 2.5× with 3× fewer iterations to converge.

5.2.5 Perfomance analysis

Table 2 presents the average timings and statistics per time-step run on a 16-core AMD Ryzen 9 3950X processor at 3.5GHz and 32GB RAM. The first column *Scene* indicates the experiment performed using our fluid solver. The columns *#Cells* and *#Particles* show the average number of fluid cells of the adaptive grid and the average number of particles, respectively. To

[1] `Eigen` does not provide a parallel implementation for these preconditioners.

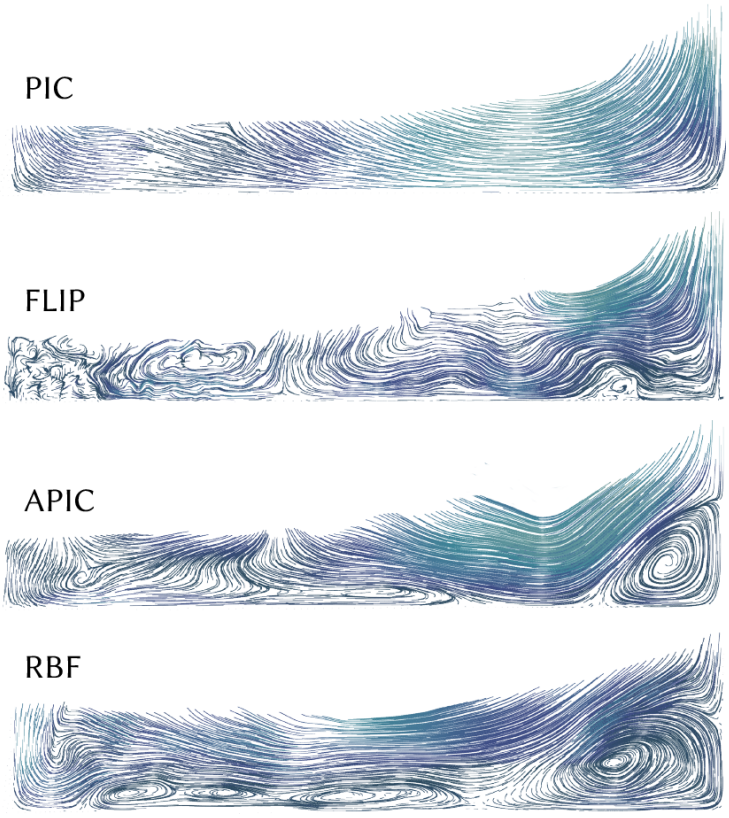
Table 2 – Average timings (in seconds) and statistics of our method per time-step.

Scene	h	#Cells	#Particles	Time				
				T_{grid}	T_{PPE}	T_{Lap}	T_{vel}	total
Figure 21	2^{-7}	169K (71%)	0.8M (58%)	3.54	0.43	0.10	0.70	6.18
Figure 21	2^{-8}	745K (45%)	4.0M (32%)	27.97	6.57	0.60	5.88	49.09
Figure 24	2^{-7}	240K (44%)	1.4M (37%)	7.36	1.25	0.25	1.52	12.97
Figure 28	2^{-7}	201K (73%)	1.0M (56%)	5.82	0.46	0.12	0.92	9.10

Source: Elaborated by the author.

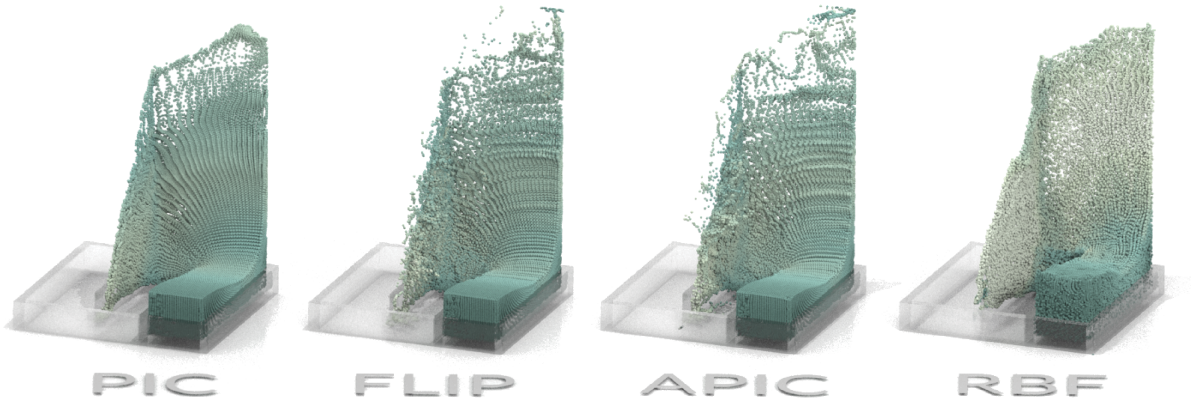
show the efficiency of our method, we include in parenthesis the percentage ratio of the number of fluid cells and particles in our method to the number of fluid cells and particles in a regular PIC solver with the same h. In the last columns, we present the average computational time consumed by adaptation of the grid (T_{grid}), solving the PPE linear system (T_{PPE}), assembling the Laplacian matrix (T_{Lap}), velocity transfers between particles and grid (T_{vel}), and the total time. All stages of our code, except the grid update, have been parallelized using OpenMP. This explains why this update is the current bottleneck of our method.

Figure 25 – Streamline visualization of the velocity field of a dam break simulation at $t = 2.2$ seconds, with characteristic grid size of $h = 2^{-6}$ (lighter colors indicate slower velocities). Note that our method preserves more vorticity than regular PIC/FLIP and similar to APIC.



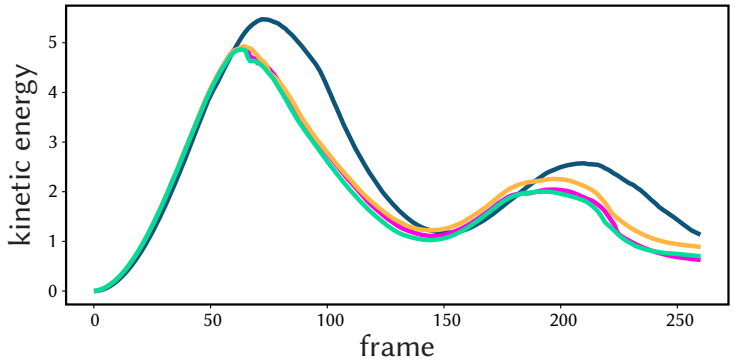
Source: Elaborated by the author.

Figure 26 – Liquid flowing around an S-shaped corridor in an octree with characteristic grid size $h = 2^{-6}$. Our RBF-based method preserves fluid sheets in comparison with regular PIC solvers. Also, our method produces stable and energetic splashes, even when using fewer particles.



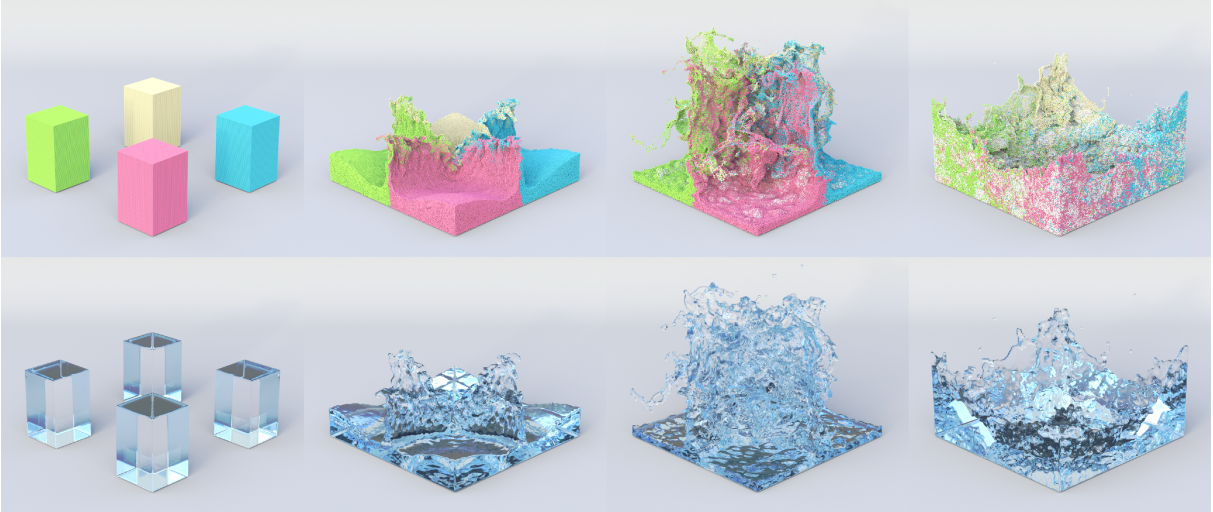
Source: Elaborated by the author.

Figure 27 – The plot of the particles' kinetic energy after velocity transfers from the grid for the simulation depicted by Figure 26: PIC (■), FLIP (■), APIC (■), and RBF (■). Our RBF-based PIC preserves more energy than regular solvers.



Source: Elaborated by the author.

Figure 28 – A quadruple dam break in an octree with characteristic grid size $h = 2^{-7}$ and 1M particles (*top*) and its surface reconstruction (*bottom*). The RBF-FD discretization reaches a speed-up of 2.6× in the PPE linear system solver. The white particles are created during particle reseeding.



Source: Elaborated by the author.

CHAPTER

6

FINAL CONSIDERATIONS

In this chapter, we highlight the final considerations of this Thesis. We showcase the paper that was accepted in a critical Journal to the Computer Graphics community. Later in the chapter, we present some current limitations in our method and potential improvements to the simulation pipeline and future projects.

6.1 Community collaboration

As we have shown in the previous chapters, our method has a solid theoretical and practical foundation, demonstrated with numerical values and qualitative figures. As a result, a paper was published at the SIGGRAPH Asia conference to recognize the method's promising results. This conference is crucial to the Computer Graphics community, with A1 CAPES stratum, the highest possible evaluation grade. This paper emphasizes the good research quality of the Visual and Geometric Processing Group (VGPG), as we are the first authors within the group to publish at ACM SIGGRAPH conference.

6.2 Limitations

As expected, the method's main limitation is the excessive and necessary calculation of the weights at each iteration of time. The number of points and the adaptability generated in the proposed algorithm will cause the neighborhood of the points to change from one instant to the next, being necessary to recalculate the RBF-FD weights. The repetitive weight computation depends exclusively on the distance between the points used, which changes as the structure adapts to the simulation's needs. We use parallel computation to solve this problem since calculating the weights between each set of points is independent. The same solution is applied for interpolating values from particles to cells and vice versa.

Parallel computation can not be applied directly to the structure modification steps. Adapting cells' size is more complicated due to dependence between neighborhood and siblings. Thus, we have to tackle the problem in a more specific way in the future. Also, the way we adapt the cells, more resolution cells just over the surface, is not optimal and could be improved. We can observe the consequences of this approach when introducing a liquid injector into the simulation: particle with high velocity are introduced, and when in contact with the fluid body, they may produce regions with information that will be lost during the reseeding step.

6.2.1 Stencil size

Recent research has shown promising applications of the RBF-FD technique in the numerical solution of PDEs (FLYER; BARNETT; WICKER, 2016; BAYONA *et al.*, 2017). In those applications, the stencil size is vital for the accuracy of the solution. For the best accuracy, it is recommended the stencil size should be approximately twice the number of polynomial terms. In our framework, we can increase the stencil size, expand the k-ring neighborhood, or complement it by using the k-nearest neighbors of the center node. In our simulations, the stencil size can reach up to 48 nodes. High-order approximations with minimal meshfree stencils (SEIBOLD, 2008) are yet a topic we have to explore further since the polynomial degree, the size, and the layout of the stencil can impact the final result of the simulation.

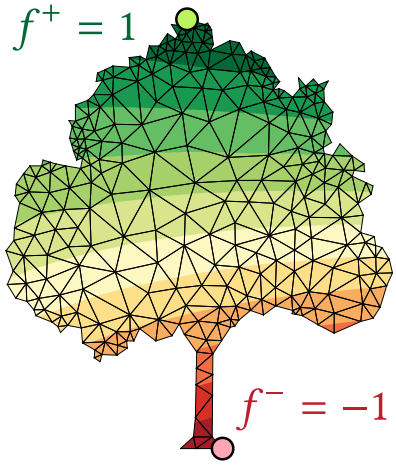
6.2.2 Levels of resolution

Due to the RBF interpolation and RBF-FD scheme we have used, the proposed method can deal with non-graded grids whose difference of the resolution levels between adjacent cells may be higher than two, although in our experiments we allow a level difference of up to two. While this is already an improvement regarding other adaptive approaches found in the literature, allowing more than two resolution levels between cells has to be further studied since it requires more stencil nodes to compute the differential weights and maintaining a reasonable number of particles over coarser cells.

6.2.3 Complex domains

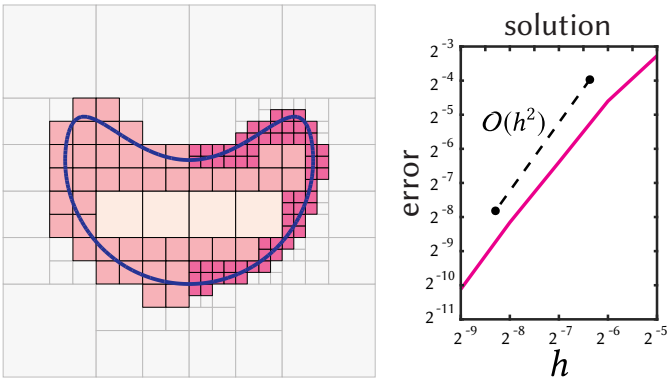
The differential operators derived from RBF-FD can be trivially adapted to complex domains modeled by unstructured meshes, as showed on Figure 29. For instance, given a simplicial mesh, we can solve the Laplace equation $\Delta f = 0$ with Dirichlet boundary conditions using RBF-FD (see inset). In this experiment, we take the 2-ring vertex neighborhood to build the RBF-FD stencils, while the boundary conditions $f^- = -1$ and $f^+ = 1$ are imposed on the bottommost and topmost vertices, respectively. However, for fluid flow simulations, the Neumann boundary condition on arbitrary domains is not a trivial task for RBF-FD and is a limitation of our method. As future investigation, we intend to study the possibility of enforcing the Neumann

Figure 29 – RBF-FD applien on a complex domain.



Source: Elaborated by the author.

Figure 30 – On the left, initial non-graded quadtree for a Poisson problem with Dirichlet boundary condition at the level-set (blue), the gray cells are discarded. On the right, the log-log plot of the error in L_∞ norm varying with h. Dashed reference line indicate ideal second-order slope.



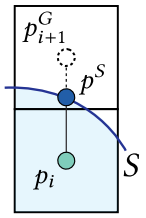
Source: Elaborated by the author.

boundary condition as an additional interpolation constraint using Hermite RBFs (MACÊDO; GOIS; VELHO, 2011).

6.2.4 RBFs near domain boundaries

Our velocity transfer step relies on an RBF-based interpolation scheme known to present suitable results over smooth continuous functions, as assumed for the Euler equations. Near the solid walls and obstacles, the velocity field can display some oscillatory behavior (Runge's phenomenon) because the RBF nodes become highly one-sided. To avoid these unwanted oscillations, during the velocity transfer from the grid cells to the particles, we clamp the interpolated particle velocities in the range of the velocities stored in the cell edges/faces. A more detailed discussion about RBF interpolations near boundaries can be found in (BAYONA; FLYER; FORNBERG, 2019).

Figure 31 – Accuracy near the free-surface can be improved assigning a ghost point over the surface, discarding the air cell center.



Source: Elaborated by the author.

6.2.5 Boundary condition for free-surface

For the free-surface flow, the exact Dirichlet pressure boundary condition, $p = 0$, needs to be ensured at the free-surface. Similar to (ENRIGHT *et al.*, 2003), we can improve the order accuracy of our discretization by using level-sets, as can be seen on Figure 31. Let S be the zero level-set, the cell centers inside S contain pressure DOFs, the cell centers outside S and belonging to a stencil cut by S are assigned ghost values p^G, and the remaining centers are discarded. For each stencil edge $(\mathbf{x}_i, \mathbf{x}_{i+1})$ cutted by S, we compute the point $\mathbf{x}^S \in S$ where the interface intersects the edge. Then, the Dirichlet boundary condition $p^S = 0$ is applied at $\mathbf{x}^S$. To assess the accuracy of this strategy, we solve a Poisson equation with an analytic pressure field given by $p(x,y) = (y - x^2 + 1)^4 + (x^2 + y^2) - 1$ and the level-set $S = p^{-1}(0)$ in the domain $\Omega = [-1.5, 1.5] \times [-1.75, 1.25]$. Fig. 30 depicts a discretization of Ω provided by a quadtree where adjacent cells can differ in depth by up to 2 levels. The log-log plot shows a second-order accurate discretization of the Dirichlet pressure boundary condition at S using our approach. Regarding level-sets, to avoid the usage of an additional high-resolution grid to compute Eulerian level-sets (AANJANEYA *et al.*, 2017), we intend to take advantage of the particles to track the free-surface by using the boundary particles to define the level-set (SANDIM *et al.*, 2019). We leave the inclusion of this boundary treatment to our framework as future work.

6.2.6 Lookup table

Concerning the performance, the computation of the weights of generalized stencils is a compute-intensive task in our simulations. We have to calculate their differential operators at each time-step for each cell and its neighbors in the staggered grid. An alternative approach would be to build a geometric dictionary storing all different stencil configurations and their respective weights as they appear during the simulation. Such a scheme would avoid computing the weights of already mapped stencils, thus increasing the overall performance of the method. Moreover, recent studies have shown promising solutions for solving RBF-FD stencils in parallel by using point indexing (ELLIOTT *et al.*, 2019), which would be another alternative to be investigated for our framework.

6.3 Conclusion

We have introduced a novel approach for liquid simulations which combines an RBF-FD pressure projection solver with the PIC method. Our RBF-FD solver can compute the pressure field and its gradients from generalized stencils on staggered cells of graded/non-graded tree-based grids. As far as we know, this is the first application of RBF-FD in Computer Graphics. Besides, we implement an adaptive grid representation, called Cellgraph, capable of updating itself over time without discarding its previous state. The results obtained with our framework, both in 2D and 3D, are visually convincing, preserving details and energy even in the presence of coarser cells and T-junctions, due to the accurate interpolation scheme and discrete differentiation we adopted.

BIBLIOGRAPHY

AANJANEYA, M.; GAO, M.; LIU, H.; BATTY, C.; SIFAKIS, E. Power diagrams and sparse paged grids for high resolution adaptive liquids. **ACM Transactions on Graphics**, v. 36, n. 4, p. 1–12, 2017. ISSN 07300301. Citation on page 62.

ADAMS, B.; WICKE, M. Meshless Approximation Methods and Applications in Physics Based Modeling and Animation. 2009. Citation on page 27.

ANDO, R.; THUREY, N.; TSURUNO, R. Preserving fluid sheets with adaptively sampled anisotropic particles. **IEEE Trans. Vis. Comput. Graph.**, v. 18, n. 8, p. 1202–1214, 2012. Citation on page 30.

ANDO, R.; THÜREY, N.; WOJTAN, C. Highly Adaptive Liquid Simulations on Tetrahedral Meshes. **ACM Trans. Graph.**, ACM, New York, NY, USA, v. 32, n. 4, p. 103:1—–103:10, 2013. ISSN 0730-0301. Citations on pages 29 and 31.

AVRAMOUSSIS, N.; WARNER, M. Distributing Liquids using OpenVDB. In: **ACM SIGGRAPH 2015 Talks on - SIGGRAPH '15**. New York, New York, USA: ACM Press, 2015. p. 2792544. ISBN 9781450336369. Citation on page 28.

BARRETT, R.; BERRY, M.; CHAN, T. F.; DEMMEL, J.; DONATO, J.; DONGARRA, J.; EIJKHOUT, V.; POZO, R.; ROMINE, C.; VORST, H. van der. **Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods**. [S.l.]: SIAM, 1994. Citation on page 50.

BATTY, C. A cell-centred finite volume method for the Poisson problem on non-graded quadtrees with second order accurate gradients. **Journal of Computational Physics**, Elsevier Inc., v. 331, p. 49–72, 2017. ISSN 0021-9991. Citation on page 30.

BATTY, C.; XENOS, S.; HOUSTON, B. Tetrahedral embedded boundary methods for accurate and flexible adaptive fluids. **Comput. Graph. Forum**, Wiley, v. 29, n. 2, p. 695–704, 2010. Citation on page 31.

BAYONA, V.; FLYER, N.; FORNBERG, B. On the role of polynomials in RBF-FD approximations: Iii. behavior near domain boundaries. **J. Comput. Phys.**, v. 380, p. 378 – 399, 2019. Citation on page 61.

BAYONA, V.; FLYER, N.; FORNBERG, B.; BARNETT, G. A. On the role of polynomials in RBF-FD approximations: II. Numerical solution of elliptic PDEs. **Journal of Computational Physics**, v. 332, p. 257–273, 2017. ISSN 10902716. Citations on pages 40 and 60.

BRIDSON, R. **Fluid Simulation for Computer Graphics**. 1st. ed. [S.l.: s.n.], 2007. 1–246 p. ISSN 1098-6596. ISBN 9781568813264. Citation on page 28.

BUDSBERG, J.; LOSURE, M.; MUSETH, K.; BAER, M. Liquids in The Croods. **DigiPro**, 2013. Citation on page 28.

CARR, J. C.; BEATSON, R. K.; CHERRIE, J. B.; MITCHELL, T. J.; FRIGHT, W. R.; MCCALLUM, B. C.; EVANS, T. R. Reconstruction and representation of 3D objects with radial basis functions. In: **Proc. of SIGGRAPH '01**. [S.l.]: ACM, 2001. p. 67–76. Citation on page 33.

CHENTANEZ, N.; FELDMAN, B. E.; LABELLE, F.; O'BRIEN, J. F.; SHEWCHUK, J. R. Liquid simulation on lattice-based tetrahedral meshes. In: **SCA '07**. [S.l.: s.n.], 2007. p. 219–228. Citation on page 31.

DA, F.; HAHN, D.; BATTY, C.; WOJTAN, C.; GRINSPUN, E. Surface-only liquids. **ACM Trans. on Graphics (SIGGRAPH 2016)**, 2016. Citations on pages 21 and 27.

EDWARDS, E.; BRIDSON, R. A high-order accurate particle-in-cell method. **Int. J. Numer. Meth. Eng.**, v. 90, n. 9, p. 1073–1088, 2012. Citation on page 47.

ELLIOTT, S.; KUMAR, R. R. P.; FLYER, N.; TA, T.; LOFT, R. Implementation of a scalable, performance portable shallow water equation solver using radial basis function-generated finite difference methods. **Int. J. High Perform. Comput. Appl.**, v. 33, n. 4, p. 619–631, 2019. Citation on page 62.

ENRIGHT, D.; FEDKIW, R.; FERZIGER, J.; MITCHELL, I. A Hybrid Particle Level Set Method for Improved Interface Capturing. **Journal of Computational Physics**, v. 183, n. 1, p. 83–116, 2002. ISSN 0021-9991. Available: <http://www.sciencedirect.com/science/article/pii/S0021999102971664>. Citation on page 28.

ENRIGHT, D.; MARSCHNER, S.; FEDKIW, R. Animation and rendering of complex water surfaces. In: **Proceedings of the 29th annual conference on Computer graphics and interactive techniques**. [S.l.: s.n.], 2002. p. 736–744. Citation on page 21.

ENRIGHT, D.; NGUYEN, D.; GIBOU, F.; FEDKIW, R. Using the particle level set method and a second order accurate pressure boundary condition for free surface flows. In: **Fluids Engineering Division Summer Meeting**. [S.l.: s.n.], 2003. v. 36975, p. 337–342. Citations on pages 21 and 62.

FASSHAUER, G. E. **Meshfree Approximation Methods with Matlab**. [S.l.]: World Scientific Publishing Co Inc, 2007. ISBN 9789812706348. Citations on pages 34 and 42.

FERSTL, F.; ANDO, R.; WOJTAN, C.; WESTERMANN, R.; THUEREY, N. Narrow band FLIP for liquid simulations. **Computer Graphics Forum**, v. 35, n. 2, p. 225–232, 2016. ISSN 14678659. Citation on page 30.

FERSTL, F.; WESTERMANN, R.; DICK, C. Large-scale liquid simulation on adaptive hexahedral grids. **IEEE Trans. Vis. Comput. Graph.**, v. 20, n. 10, p. 1405–1417, 2014. Citation on page 30.

FLYER, N.; BARNETT, G. A.; WICKER, L. J. Enhancing finite differences with radial basis functions: Experiments on the Navier-Stokes equations. **Journal of Computational Physics**, v. 316, p. 39–62, jul 2016. ISSN 10902716. Citation on page 60.

FLYER, N.; FORNBERG, B.; BAYONA, V.; BARNETT, G. A. On the role of polynomials in RB-FD approximations: I. Interpolation and accuracy. **Journal of Computational Physics**, v. 321, p. 21–38, 2016. ISSN 00219991. Citations on pages 35 and 38.

FORNBERG, B.; FLYER, N. Solving PDEs with radial basis functions. **Acta Numerica**, v. 24, n. April, p. 215–258, may 2015. ISSN 0962-4929. Citations on pages 33 and 35.

FOSTER, N.; METAXAS, D. Realistic Animation of Liquids. **Graphical Models and Image Processing**, v. 58, n. 5, p. 471–483, 1996. ISSN 10773169. Citation on page 21.

GUENNEBAUD, G.; JACOB, B.; OTHERS. **Eigen v3**. 2010. Http://eigen.tuxfamily.org. Citation on page 49.

HARLOW, F. H. **The particle-in-cell method for numerical solution of problems in fluid dynamics**. [S.l.], 1962. Citation on page 28.

HARLOW, F. H.; WELCH, J. E. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. **The physics of fluids**, AIP, v. 8, n. 12, p. 2182–2189, 1965. ISSN 00319171. Citation on page 26.

HONG, J.-M.; KIM, C.-H. Animation of Bubbles in Liquid. **Computer Graphics Forum**, Blackwell Publishing, Inc, v. 22, n. 3, p. 253–262, sep 2003. ISSN 0167-7055. Citation on page 26.

HONG, W.; HOUSE, D. H.; KEYSER, J. **An Adaptive Sampling Approach to Incompressible Particle-Based Fluid**. Phd Thesis (PhD Thesis) — Texas A & M University, 2009. Citation on page 29.

HUANG, L.; MICHELS, D. L. Surface-only ferrofluids. **ACM Transaction on Graphics**, ACM, New York, NY, USA, v. 39, n. 6, 12 2020. Citation on page 27.

IHMSEN, M.; ORTHMANN, J.; SOLENTHALER, B.; KOLB, A.; TESCHNER, M. SPH Fluids in Computer Graphics. In: **Eurographics 2014 - State of the Art Reports**. [S.l.: s.n.], 2014. Citation on page 28.

KIM, D. **Fluid Engine Development**. [S.l.]: CRC Press, 2016. 300 p. ISBN 9781498719926. Citations on pages 42 and 52.

KLINGNER, B. M.; FELDMAN, B. E.; CHENTANEZ, N.; O'BRIEN, J. F. Fluid animation with dynamic meshes. In: **ACM Trans. Graph.** [S.l.: s.n.], 2006. v. 25, n. 3, p. 820–825. Citation on page 31.

LOSASSO, F.; FEDKIW, R.; GIBOU, F.; FEDKIW, R. Simulating Water and Smoke with an Octree Data Structure. In: **ACM SIGGRAPH 2004 Papers**. New York, NY, USA: ACM, 2004. (SIGGRAPH '04), p. 457–462. Citations on pages 22 and 29.

LOSASSO, F.; SHINAR, T.; SELLE, A.; FEDKIW, R.; LOSASSO, F.; SHINAR, T.; SELLE, A.; FEDKIW, R. Multiple interacting liquids. In: **ACM SIGGRAPH 2006 Papers on - SIGGRAPH '06**. New York, New York, USA: ACM Press, 2006. v. 25, n. 3, p. 812. ISBN 1595933646. ISSN 0730-0301. Citations on pages 22, 26, and 29.

MACÊDO, I.; GOIS, J. P.; VELHO, L. Hermite radial basis functions implicits. **Comput. Graph. Forum**, v. 30, n. 1, p. 27–42, 2011. Citations on pages 33 and 61.

MAIRHUBER, J. C. On Haar's Theorem Concerning Chebychev Approximation Problems Having Unique Solutions. **Proceedings of the American Mathematical Society**, American Mathematical Society, v. 7, n. 4, p. 609, aug 1956. ISSN 00029939. Citation on page 35.

MANTEAUX, P. L.; WOJTAN, C.; NARAIN, R.; REDON, S.; FAURE, F.; CANI, M. P. Adaptive physically based models in computer graphics. **Comput. Graph. Forum**, v. 36, n. 6, p. 312–337, 2017. Citation on page 28.

MCKEE, S.; TOMÉ, M. F.; FERREIRA, V. G.; CUMINATO, J. A.; CASTELO, A.; SOUSA, F. S.; MANGIAVACCHI, N. The MAC method. **Computers & Fluids**, v. 37, n. 8, p. 907–930, 2008. ISSN 0045-7930. Citations on pages 26 and 39.

MONAGHAN, J. J. Smoothed particle hydrodynamics. **Annual review of astronomy and astrophysics**, Annual Reviews 4139 El Camino Way, PO Box 10139, Palo Alto, CA 94303-0139, USA, v. 30, n. 1, p. 543–574, 1992. Citation on page 27.

MÜLLER, M.; CHARYPAR, D.; GROSS, M. Particle-based fluid simulation for interactive applications. In: EUROGRAPHICS ASSOCIATION. **Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation**. [S.l.], 2003. p. 154–159. Citation on page 27.

NAKANISHI, R.; NASCIMENTO, F.; CAMPOS, R.; PAGLIOSA, P.; PAIVA, A. Rbf liquids: An adaptive pic solver using rbf-fd. **ACM Trans. Graph.**, Association for Computing Machinery, New York, NY, USA, v. 39, n. 6, Nov. 2020. ISSN 0730-0301. Available: <https://doi.org/10.1145/3414685.3417794>. Citation on page 24.

OHTAKE, Y.; BELYAEV, A.; SEIDEL, H. 3D scattered data interpolation and approximation with multilevel compactly supported RBFs. **Graph. Models**, v. 67, n. 3, p. 150–165, 2005. Citation on page 33.

OLSHANSKII, M. A.; TEREKHOV, K. M.; VASSILEVSKI, Y. V. An octree-based solver for the incompressible Navier-Stokes equations with enhanced stability and low dissipation. **Computers and Fluids**, Elsevier, v. 84, p. 231–246, 2013. Citation on page 30.

OLSSON, E.; KREISS, G. A conservative level set method for two phase flow. **Journal of Computational Physics**, v. 210, n. 1, p. 225–246, 2005. ISSN 0021-9991. Citation on page 27.

OLSSON, E.; KREISS, G.; ZAHEDI, S. A conservative level set method for two phase flow II. **Journal of Computational Physics**, v. 225, n. 1, p. 785–807, 2007. ISSN 0021-9991. Citation on page 27.

PENG, D.; MERRIMAN, B.; OSHER, S.; ZHAO, H.; KANG, M. A PDE-Based Fast Local Level Set Method. **Journal of Computational Physics**, v. 155, n. 2, p. 410–438, 1999. ISSN 0021-9991. Citation on page 27.

PIJL, S. Van der; SEGAL, A.; VUIK, C.; WESSELING, P. A mass-conserving level-set method for modelling of multi-phase flows. **International journal for numerical methods in fluids**, Wiley Online Library, v. 47, n. 4, p. 339–361, 2005. Citation on page 27.

SANDIM, M.; OE, N.; CEDRIM, D.; PAGLIOSA, P.; PAIVA, A. Boundary particle resampling for surface reconstruction in liquid animation. **Computers & Graphics**, v. 84, p. 55 – 65, 2019. Citation on page 62.

SATO, T.; WOJTAN, C.; THUEREY, N.; IGARASHI, T.; ANDO, R. Extended narrow band FLIP for liquid simulations. **Comput. Graph. Forum**, Wiley, v. 37, n. 2, p. 169–177, 2018. Citation on page 30.

SEIBOLD, B. Minimal positive stencils in meshfree finite difference methods for the Poisson equation. **Comput. Methods Appl. Mech. Eng.**, v. 198, n. 3–4, p. 592–601, 2008. Citation on page 60.

SOUSA, F. S.; LAGES, C. F.; ANSONI, J. L.; CASTELO, A.; SIMAO, A. A finite difference method with meshless interpolation for incompressible flows in non-graded tree-based grids. **J. Comput. Phys.**, v. 396, p. 848–866, 2019. Citation on page 30.

STAM, J. Stable Fluids. 1965. Citation on page 26.

____. Stable fluids. In: **Proceedings of the 26th annual conference on Computer graphics and interactive techniques**. [S.l.: s.n.], 1999. p. 121–128. Citations on pages 21, 28, and 39.

STOMAKHIN, A.; SCHROEDER, C.; CHAI, L.; TERAN, J.; SELLE, A. A material point method for snow simulation. **ACM Trans. Graph.**, Association for Computing Machinery, New York, NY, USA, v. 32, n. 4, Jul. 2013. ISSN 0730-0301. Citation on page 21.

STOMAKHIN, A.; SELLE, A. Fluxed animated boundary method. **ACM Transactions on Graphics**, v. 36, n. 4, p. 1–8, 2017. ISSN 07300301. Citations on pages 21 and 22.

SUSSMAN, M.; FATEMI, E. An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. **SIAM Journal on scientific computing**, SIAM, v. 20, n. 4, p. 1165–1191, 1999. Citation on page 27.

TILLENIUS, M.; LARSSON, E.; LEHTO, E.; FLYER, N. A scalable RBF–FD method for atmospheric flow. **Journal of Computational Physics**, v. 298, p. 406–422, 2015. Citation on page 33.

TURK, G.; O'BRIEN, J. F. Modelling with implicit surfaces that interpolate. **ACM Trans. Graph.**, v. 21, n. 4, p. 855–873, 2002. Citation on page 33.

ZHU, Y.; BRIDSON, R. Animating sand as a fluid. **ACM SIGGRAPH 2005 Papers on - SIGGRAPH '05**, p. 965, 2005. ISSN 07300301. Citation on page 28.


ICMC
USP
SÃO CARLOS