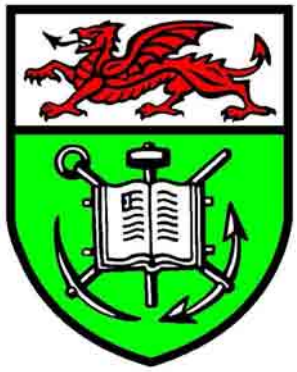


UNIVERSITY OF WALES SWANSEA

SCHOOL OF ENGINEERING

Dynamic refinement and boundary contact forces in Smoothed Particle Hydrodynamics with applications in fluid flow problems.

By

Jonathan Feldman

MMath (University of Bath)

Thesis submitted to the University of Wales Swansea
in candidature for the degree of Doctor of Philosophy

March 2006

Declaration and Statements

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed_____ (Candidate)

Date_____

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Where correction services have been used, the extent and nature of the correction is clearly marked in a footnote(s).

Other sources are acknowledged by footnotes giving explicit references.
A bibliography is appended.

Signed_____ (Candidate)

Date_____

Statement 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed_____ (Candidate)

Date_____

Abstract

Smoothed Particle Hydrodynamics (SPH) is a relatively new, simple and effective numerical method that can be used to solve a variety of difficult problems in computational mechanics. It is a fully Lagrangian meshless method ideal for solving large deformation problems such as complex free surface fluid flows.

This research was carried out with the support of BAE Systems and falls into two distinct areas. Firstly to investigate new methods for treating fixed boundaries and secondly to investigate refinement algorithms which allow for both sparsely and densely populated regions of particles within the same computational domain.

Much work has been done in the modelling of particle-boundary interactions in SPH since the governing equations do not naturally incorporate essential boundary conditions. In this research a new technique for calculating boundary contact forces is developed. The forces are obtained from a variational principle and as such conserve both the linear and angular momentum of the system. The boundaries are explicitly defined using this new approach and so the need for additional boundary particles is removed.

In the past most SPH derivations have been based on a uniform distribution of particles of equal mass. This leads to large simulations with many particles and long run times. In other mesh based schemes it has become common place to use mesh adaptivity to improve numerical results and reduce computation times. With a corresponding refinement strategy SPH can gain these same advantages.

In this research a refinement strategy based upon particle splitting is developed. Candidate particles are split into several 'daughter' particles according to a given refinement pattern centred about the original particle position. Through the solution of a non-linear minimisation problem the optimal mass distribution for the daughter particles is obtained so as to reduce the errors introduced into the underlying density field. This procedure necessarily conserves the mass of the system. The unique daughter particle velocity configuration that conserves the linear and angular momentum of the system is also identified.

The conclusion of the research was the successful implementation of these improvements into the existing SPH framework. As a result the capability and flexibility of the code is greatly increased and the computational expense needed for running large simulations has been reduced.

Acknowledgments

I would like to express my profound thanks to Prof. Javier Bonet, my supervisor, for his guidance, support and encouragement throughout all stages of this research.

I would also like to thank Dr. Robert Banim and Chris Constaninou, my industrial supervisors, for their expertise and support during the time I spent working with them. The financial support I received from BAE Systems is gratefully acknowledged.

I would like to thank all the secretaries and staff of the School of Engineering at the University of Wales Swansea for creating a very friendly and supportive research environment in which to work. A special thanks should go to Diane Cook for solving all my computer problems and generally 'fixing everything'.

A special thanks to all my friends who have helped and supported me throughout my time at university and especially here in Swansea. Barrie Cooper, Matthew Dorey, Peter Dyson, Antonio Gil, Ray Hickey, Richard Jefferson-Loveday, Dzmitry Korachkin, Sam Rolland, Paul Saunders, Javier Sila, Clare Wood, 'Kim and Alex' and 'Ben and Rod'.

Last but not least I dedicate this thesis to my parents, my brother and Marina for their constant love and support, without which the completion of this thesis would not have been possible.

Dedicated to my loving family,
and friends.

Contents

Nomenclature

The symbols most frequently used in the text are given below.

Any other notation introduced will be defined when required.

- Scalars and scalar functions are written in regular italic typeface (f, m, ρ)
- Vector and tensor quantities are written in bold typeface $(\mathbf{v}, \mathbf{M}, \boldsymbol{\mathcal{T}})$
- Subscripts $a, b, \ldots$ denote evaluation at a given particle $(m_a, \rho_a, \mathbf{x}_a)$
- Indices $i, j, k, \ldots$ denote components of the cartesian coordinate system (x^i, v^j, σ^{ij})

The Einstein summation convention is adopted for repeated cartesian indices.

Operations:

Gradient:	$\nabla f = f^{,i}$
Divergence:	$\nabla \cdot \mathbf{v} = v^{i,i}$
Scalar product of vectors:	$\mathbf{u} \cdot \mathbf{v} = u^i v^i$
Norm of vector:	$\|\mathbf{v}\| = (\mathbf{v} \cdot \mathbf{v})^{\frac{1}{2}}$
Trace of tensor:	$\mathrm{tr}\,(\boldsymbol{\sigma}) = \sigma^{ii}$
Double contraction of tensors:	$\mathbf{u} : \mathbf{v} = u^{ij} v^{ij}$
Tensor product:	$\mathbf{u} \otimes \mathbf{v} = a^i b^j$
Hessian:	$\boldsymbol{\mathcal{H}} f = \nabla (\nabla f)$

SPH operations:

$\langle f(\mathbf{x}) \rangle$	Reproducing kernel approximation of scalar function $f(x)$
$f_h(\mathbf{x})$	Summation approximation of $\langle f(x) \rangle$

Scalar quantities:

h	Smoothing length
V	Volume
m	Mass
ρ	Density (and ρ_0 is the material density)
P	Pressure
μ	Viscosity
$\dot{\varepsilon}$	Von Mises equivalent strain rate
e	Internal energy
c	Speed of sound
P_0	Artificial isothermal bulk modulus
t	Time (and Δt is the timestep)
$\delta(\mathbf{x} - \mathbf{x}_b)$	Dirac delta function based at $\mathbf{x}_b$
$w_b(\mathbf{x}, h_b)$	Kernel function based at $\mathbf{x}_b$ $(w_b(\mathbf{x}, h_b) = w(\mathbf{x} - \mathbf{x}_b, h_b))$
$N_b(\mathbf{x})$	SPH shape function $(N_b(\mathbf{x}) = V_b w_b(\mathbf{x}, h_b))$
$\hat{w}_b(\mathbf{x}, h_b)$	Corrected kernel function (constant or linear)
$\alpha(\mathbf{x})$	Constant kernel correction term

Scalar quantities (cont.):

$E\langle f, \mathbf{x}\rangle$	Error introduced by the reproducing kernel approximation of f at $\mathbf{x}$
$E_h(f, \mathbf{x})$	Error introduced by the summation approximation of f at $\mathbf{x}$
ψ_{visc}	Viscous potential per unit volume
π	Internal energy per unit mass
$K(\mathbf{v})$	Total kinetic energy of the system
$\Pi_{\text{ext}}(\mathbf{x})$	Total external energy of the system
$\Pi_{\text{int}}(\mathbf{x})$	Total internal energy of the system
$\Pi_{\text{visc}}(\mathbf{x})$	Total viscous dissipation of the system
$\gamma(\mathbf{x}_b, h_b)$	Gamma function based at $\mathbf{x}_b$ with smoothing length h_b
(ε, α)	Refinement parameter (separation parameter ε, smoothing ratio α)
$\mathcal{E}[\varepsilon, \alpha](\boldsymbol{\lambda})$	Density refinement error for given mass distribution $\boldsymbol{\lambda}$
$\mathcal{E}^*[\varepsilon, \alpha]$	Minimum density refinement error for refinement pattern (ε, α)

Vector and tensor quantities:

$\mathbf{x}$	Position vector
$\mathbf{v}$	Velocity vector
$\mathbf{a}$	Acceleration vector
$\mathbf{n}$	Normal vector
$\mathbf{I}$	Identity tensor
$\boldsymbol{\sigma}$	Stress tensor (and $\boldsymbol{\sigma}'$ deviatoric stress tensor)
$\mathbf{d}$	Rate of deformation tensor ($\mathbf{d} = \frac{1}{2}\left(\nabla\mathbf{v} + \nabla\mathbf{v}^T\right)$)
$\mathbf{d}'$	Deviatoric rate of deformation tensor ($\mathbf{d}' = \mathbf{d} - \frac{1}{3}(\nabla\cdot\mathbf{v})\mathbf{I}$)
$\mathbf{F}$	External force
$\mathbf{T}$	Internal force
$\mathbf{T}^P$	Pressure force
$\mathbf{T}^{\text{dev}}$	Deviatoric component of internal force
$\mathbf{T}^B$	Boundary contact force
$\mathbf{T}_{ab}$	Interaction force between particle a and particle b
$\boldsymbol{\beta}(\mathbf{x})$	Linear kernel correction term
$\mathbf{W}(r)$	Vector function satisfying $\nabla\cdot\mathbf{W} = w$
$\nabla w_b(\mathbf{x}, h_b)$	Gradient of the kernel function based at $\mathbf{x}_b$ with smoothing length h_b
$\nabla N_b(\mathbf{x})$	Gradient SPH shape function ($N_b(\mathbf{x}) = V_b\nabla w_b(\mathbf{x}, h_b)$)
$\tilde{\nabla} w_b(\mathbf{x}_a, h_b)$	Corrected gradient of kernel function (constant or linear)
$\tilde{\nabla}\hat{w}_b(\mathbf{x}_a, h_b)$	Corrected gradient of constant corrected kernel function
$\boldsymbol{\xi}_a$	Constant gradient correction term evaluated at $\mathbf{x}_a$
$\mathbf{L}_a$	Linear/mixed gradient correction term evaluated at $\mathbf{x}_a$
$\tilde{\boldsymbol{\mathcal{H}}} w_b(\mathbf{x}_a, h_b)$	Corrected kernel Hessian
$\mathbf{B}_a$	Matrix correction term for the kernel Hessian evaluated at $\mathbf{x}_a$
$\boldsymbol{\mathcal{A}}_a$	Tensor correction term for the kernel Hessian evaluated at $\mathbf{x}_a$
$\tilde{\tilde{\nabla}}\hat{w}_b(\mathbf{x}_a, h_b)$	Corrected and stabilised gradient of the kernel function
$\tilde{\tilde{\nabla}} f_h(\mathbf{x}_a)$	Corrected and stabilised gradient of scalar function $f(\mathbf{x})$

Chapter 1

Introduction

Free surface fluid flows arise in a wide variety of circumstances and represent a challenging class of problems that scientists and engineers are seeking to better understand. The sloshing of fuel tanks, the wake surrounding a ship, the impact of solid objects into water, the flow over hydrofoils and the flow through turbines and propellers are all examples of large scale complex free surface fluid flows.

With the sea level predicted to rise by as much as 50cm over the next 100 years[1] the risk of flooding is increasing all over the world and the numerical simulation of free surface fluid flows will be essential to help predict flood damage and to aid the design of effective flood defense structures. Methods to efficiently and accurately model the behaviour of these complex systems is of great economic, environmental and human interest.

Grid based numerical approaches such as the finite difference method and finite element method have fast developed into extremely powerful and flexible tools for the solution of many important problems in the field of computational continuum mechanics. However, these methods require expensive re-meshing algorithms for problems which exhibit large deformations and typically free surfaces are difficult to follow with any degree of accuracy.

This thesis presents a 'meshless' numerical technique particularly suited to the simulation of free surface fluid flows with several advantages over traditional finite element approaches.

[1] wikipedia.org: `http://en.wikipedia.org/wiki/Global_Warming`

Case Study: The Asian tsunami

On the 26th of December 2004 at 7:58am local time an earthquake in the middle of the Indian ocean measuring 9.15 on the Richter scale occurred along a 1200km stretch of the subduction zone between the Australian and Eurasian tectonic plates[2].

The earthquake (the second largest ever recorded) and the subsequent tsunami killed more than 200,000 people who inhabited the coastline of 13 neighbouring countries including Indonesia, Sri Lanka, South India and Thailand; making it one of the most deadly disasters in modern history.

Over a period of several minutes the plates slipped roughly 15m causing the seabed to rise by as much as a few metres, displacing an estimated 30km^3 of water. In order to regain its equilibrium this disturbed mass of water collapsed under the influence of gravity resulting in the destructive tsunami waves that claimed so many lives. The area affected was greatly increased because of the large region of subduction which caused the waves to spread outwards along the entire length of the faultline.

Deceptively, the power of tsunamis are not evident in deep regions of ocean where the waves are largely harmless and often pass by unnoticed. For the Asian tsunami the maximum recorded height of the waves in the Indian ocean was only 2ft. It was only as the waves approached the shallower water of the coastlines that they became a significant threat.

As the Asian tsunami reached the shallower water the waves slowed from upwards of 500kmh^{-1} to 1000kmh^{-1} down to tens of kilometres per hour; forming large destructive waves that were capable of traveling as far as 2km inland. In the region of Aceh in Indonesia it has been proposed that the waves reached heights of up to 24m, rising to 30m further inland. The Tsunami Society[3] estimate that the total energy of the resulting waves was approximately 2×10^{16} joules equivalent to 5 megatons of TNT. The destructive power of the tsunami wave at Aceh can clearly be seen in Figure 1.1.

[2] NewScientist.com: `http://www.newscientist.com/popuparticle.ns?id=in51`
[3] The Tsunami Society: `http://www.sthjournal.org/soc.htm`

January 10, 2003 December 29, 2004

Figure 1.1: Aceh, Sumatra, Indonesia

Tsunami waves hit coastlines with tremendous force generated by the increased weight and pressure of the ocean behind them, surging inland with enough energy to destroy almost anything in its path. The resulting debris including any vehicles, ships or boulders in its path are carried by the currents further increasing the wave's destructive force and eroding coastal areas down to the bedrock.

The human, economic and environmental costs of such a disaster are impossible to measure. Exact figures are still unknown but it is likely that more than 200,000 people lost their lives in the floods (a third of whom were women and children) and in total over 500,000 people were injured in the disaster.

The affected areas will feel the economic impact for many years to come. Millions lost their homes, livelihoods and access to food and clean water. Local infrastructures were seriously damaged with water supplies and farm land contaminated by salt water for the foreseeable future.

The Asian tsunami hit the poorest people in the region the hardest where the local economies are largely driven by tourism, farming and the fishing industry. Much of the fishing community was left at a standstill after the floods due to the loss of fishing boats and equipment. In Sri Lanka alone the fishing industry accounted for the employment of over 250,000 people.

Tsunamis can have far reaching consequences and should be a global concern. The affect of the Asian tsunami was felt as far away as Struisbaai in South Africa some 8,500km away from the epicentre, where sixteen hours later it caused high tides of 1.5m.

Unfortunately, it is impossible to accurately predict when or where future tsunamis will occur. A great deal of effort is being put into the development of early warning systems and coastal defences in vulnerable regions. However, tsunami detection in deep water is a difficult task due to the low amplitude of the waves in relation to the surrounding ocean. Such systems require many interconnected sensors to be effective at great cost. Consequently, the initial earthquake remains the best means for the detection of imminent tsunamis but in poor areas even issuing timely tsunami warnings can be a difficult task due to the lack of sufficient communication infrastructures.

Case Study: Flooding in the UK

Closer to home it is speculated that the Bristol channel floods of the 30th of January 1607 may have been caused by a tsunami originating off the Irish coast. The floods claimed an estimated 2,000 lives and caused widespread damage to farmland, livestock and villages along the channel including the town of Cardiff in Wales.

The floods had long been considered a result of a combination of high tides and other meteorological factors. However, eyewitness accounts from the time tell of *'huge and mighty hills of water'* advancing at speeds *'faster than a greyhound can run'* that only receded ten days later[4]. This has lead scientists to investigate alternative causes.

The possibility that an earthquake off the coast of the United Kingdom and Ireland may have triggered a tsunami is not as unlikely as it may seem. In fact on the 8th of February 1980 in that region an earthquake measuring 4.5 on the Richter scale[5] was recorded confirming the fault remains active.

[4] Burnham-On-Sea website: `http://www.burnham-on-sea.com/1607-flood.shtml`
[5] BBC News: `http://www.news.bbc.co.uk/2/hi/uk_news/wales/4397679.stm`

The Thames barrier has been forced to close 90 times to prevent serious flooding since it was completed in 1983, averaging four times a year[6]. However, the authorities anticipate this to increase to up to 30 times per year by 2030 due to the predicted rise sea levels and increased number of violent storms caused by global warming[7].

The Thames estuary is home to over one million people, 500,000 properties, 38 underground stations and the city airport[8]. A large scale flood across this region would cause millions of pounds worth of damage to infrastructure, residences and businesses; not to mention the potential for loss of life. This is clearly seen in Figure 1.2 which shows the flood plain around the city of London.

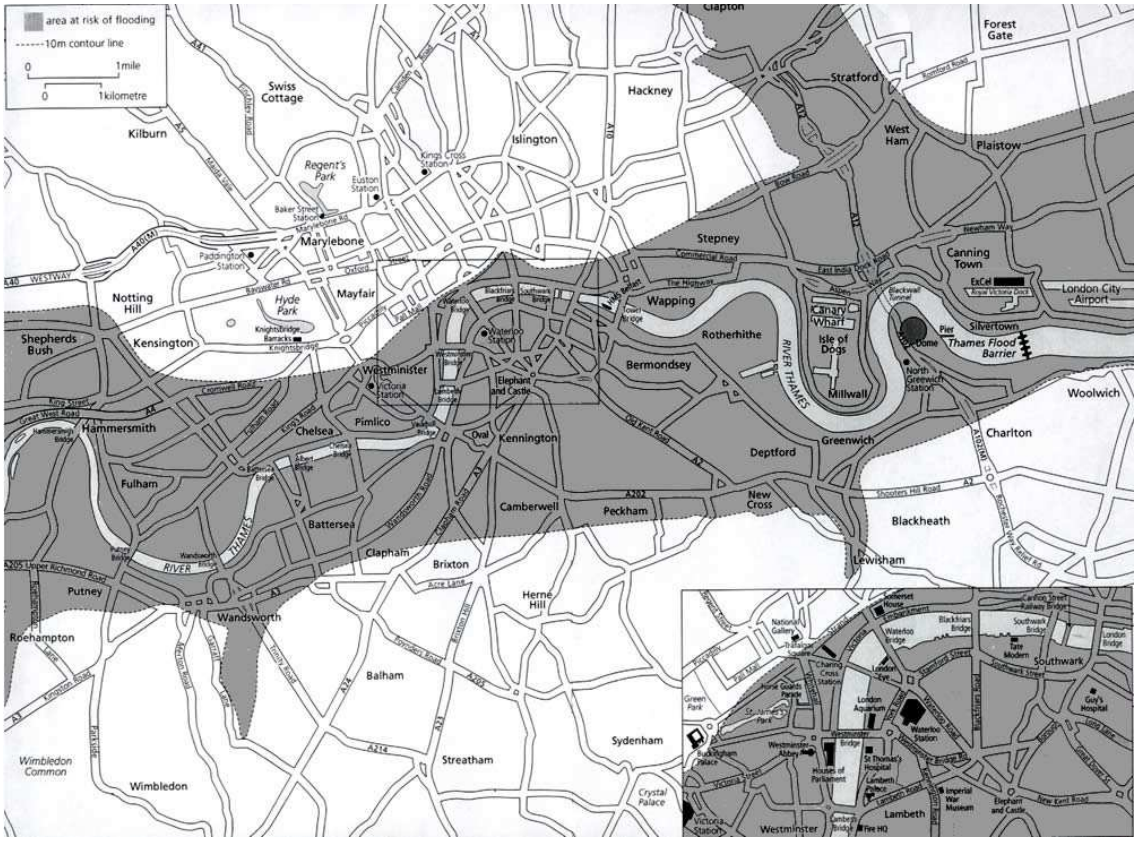


Figure 1.2: City of London flood plain

[6] The Environment Agency: `http://www.environment-agency.gov.uk`
[7] NBC News: `http://msnbc.msn.com/id/6241449`
[8] The Thames Estuary Partnership: `http://www.thamesweb.com`

Changing weather patterns and rising sea levels caused by global warming mean that the risk of flooding is increasing all over the world. It remains the job of flood defences to help reduce the potential damage and danger caused by flood waters.

In Japan extensive walls up to 4.5m high have been erected to protect populated coastal regions and in other parts of the world channels and flood gates have been built in an attempt to redirect potential flood waters (as shown in Figure 1.3).

Figure 1.3: Tsunami wall at Tsu-Shi, Japan and the Thames barrier, London

The use of computer simulations is playing an increasing role in helping to predict and assess the impact of future flood scenarios. Numerical modelling is also essential in the design and testing of flood defences where, due to the scale and complexity of the engineering problems involved, building full scale models is impossible and laboratory experiments prohibitively expensive. However, the situations illustrated above are examples of large, complex fluid flow problems which have traditionally been difficult to model using well established numerical techniques.

This thesis presents the 'meshless' numerical method known as Smoothed Particle Hydrodynamics (SPH) which is particularly suited to the solution and modelling of large scale, complex free surface fluid flows such as those discussed in this introduction. In addition SPH can be easily be extended to simulate the related problems of rock and debris flows with only simple modifications to the constitutive model.

The remainder of this chapter proceeds as follows:

- The next section presents a brief introduction to meshless methods. Although short it should provide the reader with an understanding of the historical context and motivation behind the development of meshless numerical methods.
- A detailed description of the development of the state of art Smoothed Particle Hydrodynamic method is given. This includes numerous references to literature.
- The aims and scope of the thesis are presented along with an up to date literature review of the relevant topics in SPH.
- The layout of the remainder of the thesis is given and the contents of each chapter is described.

1.1 Meshless methods

In recent years traditional grid based numerical methods such as the finite difference method and the finite element method have fast become the standard tools for the solution of a wide range of engineering problems in the areas of computational fluid and solid mechanics.

In approaches such as these the spatial domain over which the governing partial differential equations are to be solved is discretized using a set of interconnected nodes defined by an underlying grid or mesh. For a given mesh, the governing equations are approximated by a set of algebraic equations which can then be assembled and solved.

However, grid based numerical methods are not without their limitations:

- Mesh generation is an essential but time consuming process. The creation of regular meshes for complex or irregular geometries is both an intellectually and mathematically challenging task in its own right.
- In Eulerian formulations it is very difficult and computationally expensive to accurately resolve free surfaces and track moving interfaces and boundaries as the problem evolves over the fixed mesh.

- In Lagrangian formulations large deformations often lead to excessive element distortion. This can severely reduce the accuracy of the method and reduce the maximum stable timestep. Regular re-meshing can help to circumvent these problems but often introduces additional diffusion to the simulation and as a result material can no longer be accurately tracked.

Meshless numerical methods are those which attempt to solve these long standing problems of traditional mesh based approaches by providing a framework in which general partial differential equations can be solved without the need for any underlying regular mesh or nodal connectivity.

With no explicit mesh to generate the initial preprocessing time is reduced and the need for subsequent re-meshing is totally eliminated since mesh entanglement no longer occurs. Without a mesh the motion of the nodes are no longer constrained so material and free surfaces can be accurately tracked. Consequently, meshless methods can easily simulate problems involving large deformation and fragmentation.

In grid based schemes mesh adaptivity has been used to improve numerical results and reduce computation times. In this respect meshless methods show a great deal of promise. Adaptivity is easily implemented since nodes can be added or removed at will without the implications of mesh regeneration.

Over the last twenty-five years research into the next generation of meshless numerical methods has gained considerable momentum and several different approaches have been developed. A number of good review papers and books are currently available which cover many of the recent developments in this field [10, 47, 75, 76, 127].

Two notable early meshless techniques are known as the Marker And Cell (MAC) and Particle In Cell (PIC) methods which were developed in 1960's by Harlow [50–52] at the Los Alamos laboratory, California.

Although these formulations still use a Eulerian mesh and the finite difference method for the solution of the Navier-Stokes equations they were the first to incorporate a set of Lagrangian 'marker' particles which move with the fluid. The main disadvantage of both these methods is the continual mapping of variables between the particles and the Eulerian mesh which introduces large amounts of dissipation.

The MAC method was originally developed to model confined, viscous, incompressible fluid flows [51,52]. In later applications to low viscosity flows the MAC method was found to be unstable and poorly capture free surfaces [25]. This was attributed to both the simplified implementation of the boundary conditions and the method used to extrapolate the particle velocities from the Eulerian mesh. Later variations of the MAC method [2, 25] improved its accuracy and extended the boundary conditions to include curved and moving boundaries [124].

Similarly, the PIC method models the fluid as a set of Lagrangian particles moving through a fixed grid of cells [23,50]. In the original PIC formulation each particle is defined only by its position and mass. All other cell properties are calculated and updated by the transition of particles moving from one cell to another. It was found later that numerical dissipation could be reduced if the particles were assigned all fluid properties such as momentum and energy. With these improvements the PIC method has been successfully adapted to solve a variety of solid mechanics problems [119–121].

Smoothed Particle Hydrodynamics (SPH) is the earliest of the truly meshless methods. SPH is a fully Lagrangian particle method based upon a smoothing interpolation technique known as the reproducing kernel approximation. By introducing kernel functions with compact supports this approach allows any function to be approximated by a weighted average of its values over a finite set of disordered points.

In this way the continuum is described by a large set of Lagrangian points, known as particles, which move with the material. Kernel functions are centred at each particle which in conjunction with the discretized governing equations determine the particle interactions and the motion of the continuum.

First published in 1977 by Lucy [86] and Gingold [95] the SPH method was initially used to model large scale astrophysical problems such as the formation of binary star formations and galaxies in the early universe. Since then SPH has established itself as a flexible method for the solution of both fluid and solid mechanics problems and is particularly suited to the solution of large deformation problems and complex free surface flows.

Since this thesis is concerned with the SPH method an in depth overview of its development is given in Section 1.2.

Another formulation derived using the reproducing kernel approximation is known as the Reproducing Kernel Particle Method (RKPM) [77, 81]. It was proposed by Liu in an effort to enforce consistency and improve accuracy of meshless methods in the vicinity of boundaries. The difference between the RKPM and SPH comes from the addition of a boundary correction terms introduced to the kernel. Liu later modified this method to incorporate a moving least square interpolation technique to generate the kernel corrections [82, 115]. The resulting formulation is n^{th}-order consistent and became known as the Moving Least Square Reproducing Kernel Particle Method (MLSRKPM).

The moving least squares approximation is also used in the Element-Free Galerkin Method (EFGM) [11] developed by Belytschko. Based upon the Diffuse Element method (DEM) [65, 104] the EFGM uses moving least square interpolants to construct the trial and test functions of the Galerkin weak form and improves the previous boundary implementation with the use of Lagrange multipliers to enforce the boundary conditions.

Both the DEM and EFGM require a background mesh in order to evaluate integral terms that appear in the final system of equations and as such are not considered to be truly meshless methods. The EFGM has been applied successfully to a wide range of problems including heat transfer, elasticity and fracture [60, 85].

For completeness several other meshless methods should be mentioned in this summary. These are the h-p cloud method [40, 74], the natural element method [118], the moving particle semi-implicit method [63], the meshless local boundary integral equation method [4], the meshless local Petrov-Galerkin method [4] and most recently the meshless finite element method [58].

As these methods become more established the amount of research into the development and improvement of meshless methods increases. With the attention of numerous researchers it is hard to predict how this field will develop but it seems that the future of meshless numerical methods is assured.

1.2 The development of Smoothed Particle Hydrodynamics

Since the initial development of SPH by Lucy, Gingold and Monaghan [86, 95] in 1977 many initial weaknesses have been identified and resolved.

Currently there is a great deal of research taking place into SPH. The longest running groups being those of Monaghan [88–100] at Monash University and Cleary [28–33] at CSIRO, both based in Australia.

For a number of years the Civil and Computational Engineering Centre at the University of Wales Swansea has been investigating reproducing kernel based particle methods. The papers of Bonet [13–21], Kulasegaram [66–69] et al [44, 84, 111] cover a wide range of topics including a variational derivation of the SPH equations, the development of kernel consistency corrections (CSPH), a shallow water formulation, variable smoothing length simulations and a new boundary implementation.

In 2003 the first book devoted to SPH was written by Liu [76] and SPH also features in the first book to deal exclusively with meshless methods, entitled 'Mesh Free Methods' [75] also written by Liu. In addition to these books several good SPH review papers exist in the literature [33, 108, 125].

Two early reviews by Monaghan [89, 92] provide a good background to the early development of SPH. These papers cover the derivation of the classical SPH governing equations and details the first improvements made to the method. These include the formulation utilising smoothed particle velocities (known as XSPH) which help to keep particles equally spaced [90] and artificial viscosity terms that were used to help resolve shocks in the absence of a physical viscosity [97]. These papers also contain references to many of the earlier SPH applications to astrophysical problems.

Soon after Libersky extended the SPH formulation to include the full stress tensor in order to simulate problems with material strength [72, 73, 98]. The introduction of more sophisticated material models highlighted one of the main weaknesses of SPH known as tensile instability which can result in unphysical clustering of particles and numerical fracture.

Tensile instability was first studied in detail by Swegle [122]. It was found that this numerical instability was caused by the kernel interpolation procedure. From a Von Neumann stability analysis the stability criterion is found to be $w''\boldsymbol{\sigma} > 0$ where w'' is the second derivative of the kernel function and $\boldsymbol{\sigma}$ is the stress (negative in compression) [5]. Consequently, this criterion may not be satisfied by materials in either tension or compression. However, in most cases the particle separation is such that the instability only manifests itself in regions of tension.

Various methods to prevent or reduce the effects of tensile instability have been proposed. Monaghan showed that tensile instability can be removed with the introduction of artificial stresses [93]. Dyka and Randles introduced the concept of separate stress points in SPH [41, 109]. In this approach the particle stresses are not evaluated at particles rather at surrounding stress points, similar to quadrature points used in the finite element method. This approach has been successful modified for two dimensional problems by Vignjevic [126]. Hicks proved that tensile instabilities could not be removed by artificial viscosities and developed the conservative smoothing method [55–57] which adds stabilizing dissipation terms that can be applied to SPH simulations as well as other numerical methods. Most recently, Bonet has proved that tensile instability is a property of the continuum mechanics equation for elastic fluids [15] and not necessarily a defect of the SPH formulation. This source of instability is then shown to be eliminated by using a total Lagrangian formulation where all derivatives of the kernel functions are taken with respect to a fixed reference configuration [12].

Zero-energy modes are another weakness found to be present in the SPH method. Not unique to particle methods, these spurious modes are generated by nodal underintegration caused from evaluating the function derivatives at the same point at which the function values are sought. If not identified such modes can grow and eventually dominate the solution. Spurious modes have been addressed by introducing stabilization potentials [13, 14, 16] and by computing derivatives at neighbouring stress points [126] rather than at the particles themselves, where the derivative of the kernels will always be zero.

Belytschko [9] showed that the traditional SPH method lacked even zeroth order consistency. Improvements in the consistency and accuracy of the standard SPH

equations have come via the introduction of kernel correction methods which enforce the discrete consistency conditions. The normalised kernel SPH method [59,109] and the corrected SPH method (CSPH) [13, 16, 18] are two such methods which enforce first order consistency. Several other meshless methods have been developed in the process, such as the element-free Galerkin method [11], reproducing kernel particle method [81] and moving least-square particle hydrodynamics method [38, 39] which can enforce consistency upto any order. It has been proved by Kulasegaram [67] that the kernel correction methods are identical to those of the RKPM and MLSRKPM.

The convergence of meshless methods is still not very well understood. Early results from Moussa have proven the convergence of the SPH method for scalar, non-linear, conservation laws [102,103]. These constitute the first rigourous convergence results for meshless methods.

Formulations based upon combining particle based methods, such as SPH, with traditional mesh based approaches have the potential to utilise the best properties of both methods. Mixed formulations for coupling SPH to the finite element method have already been proposed by several authors [8,43,64] as a way to enforce essential boundary conditions and for modelling fluid-structure interactions.

In the paper by Fernádez-Méndez [44] SPH particles are introduced locally into regions where previously mesh degradation and element distortion had prevented the finite element method from converging. The coupling of SPH with the discrete element method has been used to model particulate flows where a viscous fluid contains solid particles [32,107]. Such formulations show great promise for the biological and environmental sciences where blood clots in arteries, or landslides and lava flows could be more accurately modelled.

Over the last 25 years SPH has developed into a simple and reliable meshless method which is capable of modelling complex physics and has been applied to a remarkable variety of problems, across many different disciplines. These problems include free surface flows [91], multiphase flows [110], viscoelastic flows [42], gas dynamics and explosions [78,79], fragmentation and penetration [48,71], heat conduction [28], material strength [73] and casting [29, 30].

1.3 Scope of the thesis

This thesis presents several developments and improvements to the smoothed particle hydrodynamics (SPH) method. The scope of the work can be broadly divided across four main objectives:

- To present a detailed study of the state of the art SPH method for the simulation of incompressible free surface fluid flows.
- To investigate new methods for treating fixed boundaries in SPH that avoid the need for uniformly spaced boundary particles.
- To develop a variable resolution formulation of the SPH method and investigate various particle refinement algorithms. The main aim being to implement a general particle splitting routine that both improves the accuracy and reduces the computation expense required for large scale SPH simulations.
- To combine these developments into a single, flexible, variable resolution SPH code incorporating the new dynamic particle refinement algorithms.

The remainder of this section constitutes a survey of the existing SPH research concerned with the topics of incompressible free surface flow simulations and the current adaptivity implementations in SPH.

1.3.1 Incompressible free surface flow simulations in SPH

Monaghan [91] was the first to apply the SPH method to the simulation of incompressible free surface flows. Rather than working directly with the incompressibility constraint he noted that in reality fluids such as water are compressible, but have a sound speed that is considerably faster than the speed of the bulk flow. In this way an incompressible flow can be simulated by a fluid which is more compressible than a real fluid. To ensure the relative density fluctuations are only of the order of 1% the Mach number of the flow must be sufficiently small. It should be noted that by reducing the Mach number, the speed of sound of the fluid will increase and the stable timestep can be significantly restricted by the Courant condition. However, since this method is explicit and comparative methods often require several iterations to converge this is not a great penalty to pay. Monaghan successfully

modelled the breaking dam experiment of Martin [87], the formation of a bore and the propagation of waves up a sloped incline in two dimensions. These problems have since become the standard benchmarks for free surface simulations using SPH.

This artificial compressibility formulation of SPH is adopted and used as the basis for the work contained in this thesis. A detailed description and several applications of this formulation can be found in subsequent chapters.

With the same approach Morris [101] used SPH to simulate low Reynolds number incompressible flows. Morris introduces an extra dynamic pressure term to the hydrostatic pressure in order to more accurately evaluate the pressure gradients of the flows. He also suggests an alternative expression for artificial viscosity. The method accurately simulated Poiseuille flows, Couette flows and the flow around a cylinder with results in good agreement with the analytical solutions and comparative finite element solutions. Sigalotti [117] and Takeda [123] later used the same formulation to model the related 3D Hagen-Poiseuille flow in a capillary tube of circular cross-section.

The fluid-structure interaction problem consisting of a rigid box sinking vertically under its own weight into a tank of water has also been investigated by Monaghan [99]. The motion of the box and the resulting solitary wave generated by the displaced fluid are simulated using the SPH method. Particles were used to describe both the fixed boundary of the tank and the moving boundary of the box, as well as the fluid. The acceleration of the box was determined by the total force acting on the box from the surrounding fluid particles. The fluid cavity formed in the vicinity of the box as it drops and the height of the resulting wave were in satisfactory agreement with the experimental data.

Problems involving interfaces between fluids of varying densities have also been simulated using SPH. Monaghan [94] modelled the flow of a fluid of one density under the influence of gravity into a stratified tank consisting of two layers of fluid of different densities. The simulations exhibited larger variations in density than the experiments but the numerical results were still encouraging. The thickness and velocity of the head of the gravity current were accurately predicted and the ampli-

tude of the waves generated were in good agreement with the experimental data.

This approach was later refined by Colagrossi [34] to more accurately resolve interfaces between two fluids. The resulting SPH formulation remains stable even when the density ratio between the fluids is small. This allows simulations of air-fluid flows with interface breaking and air-entrapment. It has been successfully used to model bubbles of fluid rising through another fluid with a density ratio of only 0.001 and a two-phase collapsing dam problem which incorporates the surrounding air and accurately models the air entrapment as the wave breaks.

Shao and Lo [83,114] have implemented a predictor-corrector fractional step method to enforce incompressibility in SPH simulations which is based on the SPH Projection method of Cummins [37]. The first step is an explicit integration in time which generates intermediate particle positions and velocities without enforcing incompressibility. The second, corrective, step is then applied to adjust the particle densities back to the initial constant values prior to the prediction step. The compromise with this approach is that the pressure is no longer an explicit thermodynamic variable, rather it is obtained through the solution of a pressure Poisson equation which needs to be solved using a preconditioned conjugate gradient method. This approach has successfully modelled the breaking dam problem, mud flows, the Rayleigh-Taylor instability, and solitary waves breaking against a vertical wall and running up a plane slope.

1.3.2 Adaptivity and variable resolution SPH simulations

As previously mentioned one of the most promising features of meshless methods, such as SPH, is the relative ease with which adaptivity and variable particle resolutions can be introduced into simulations.

Several different formulations have been proposed which implement adaptivity of one form or another into SPH simulations. These fall broadly into two categories:

- Adaptive smoothing length methods – whereby the smoothing length h and the shape of the support of the kernel functions w can dynamically adapt according to the relative motion of the particles.
- Particle refinement methods – whereby particles can be removed, added or relocated in regions that satisfying given refinement criteria or error measures.

Adaptive smoothing length methods

In SPH the particle smoothing length h determines the resolution of simulations and controls the number of neighbouring particles which contribute to the evaluation of the material properties at any given point (see Figure 1.4).

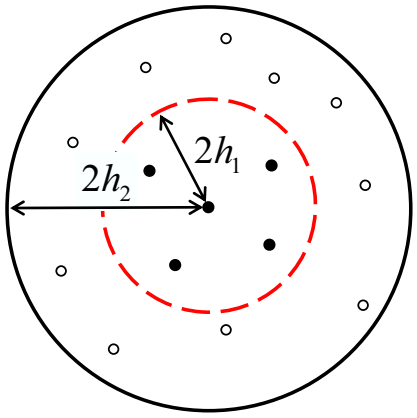


Figure 1.4: Spherical kernel with varying smoothing lengths

Early implementations of SPH used spherical kernel functions with a constant, global smoothing length. For large deformation problems it was soon noticed that in regions of expansion particles would end up with too few neighbour particles; while in regions of contraction particles would end up with an excessive number of neighbour particles. This lead to unbalanced and unstable simulations.

In general, particle smoothing lengths can be a function of space, time and the relative motion of neighbouring particles. By adjusting the smoothing length intelligently the accuracy and efficiency of the SPH method can be improved.

A review of the early applications using adaptive smoothing lengths can be found in the paper of Monaghan [92]. In the initial investigations, rather than being a constant global value, the particle smoothing lengths varied in time but remained constant in space. The value of the smoothing length was adjusted in proportion to the inverse of the average density of the simulation, $h \propto \bar{\rho}^{-1/d}$ where d is the dimension of the simulation. This resulted in a global smoothing length which grew and contracted with the average density of the simulation.

Soon after, formulations where the particle smoothing lengths were evolved as a function of both space and time were developed. In the simplest case smoothing lengths were adjusted to ensure that the number of neighbours remained roughly constant for each particle throughout the duration of the simulation. When each particle has its own individual smoothing length momentum is no longer conserved since the interactions between particle pairs are not necessarily symmetric. Therefore, conservation of momentum was enforced by generating a symmetric kernel for each pair of particles by using the average of the two individual kernel functions $w_{ab} = \frac{1}{2}(w_a + w_b)$.

When the smoothing length is a function of space the spatial derivative of the kernel function should include a term coming from the spatial derivative of the smoothing length [1, 92]. This introduces additional terms to the governing equations but these have often been neglected since, in many cases, they have a negligible effect on simulations. However, without these terms the equations of motion are no longer conservative. By defining a functional form for the smoothing length, the spatial derivative can be calculated explicitly and the terms included in the formulation. It has been shown the inclusion of these terms has no detrimental effect to the standard SPH method but can significantly improve energy conservation in certain situations [105].

Bonet has recently introduced variable smoothing lengths into a variational formulation of SPH [21]. The addition of variable smoothing lengths introduces extra terms into the governing equations and results in a non-linear expression for the

SPH equation for density. This equation for the density is solved by a Newton-Raphson iteration procedure at each timestep. Despite this being computationally more expensive than non-iterative techniques the new formulation was found to be more accurate and required fewer particles than a standard SPH formulation. This method has been successfully applied to simulations using the Lagrangian shallow-water equations to model the collapse of a circular dam and flows over various terrains [112].

In many problems the motion of particles can generate non-uniform distributions where there are a greater number of particles in one direction than there are in another. The occurrence of these anisotropic particle distributions has lead to the development of several approaches which deform the support of the kernel functions in order to adapt to the local particle distribution and the motion of particles (see Figure 1.5).

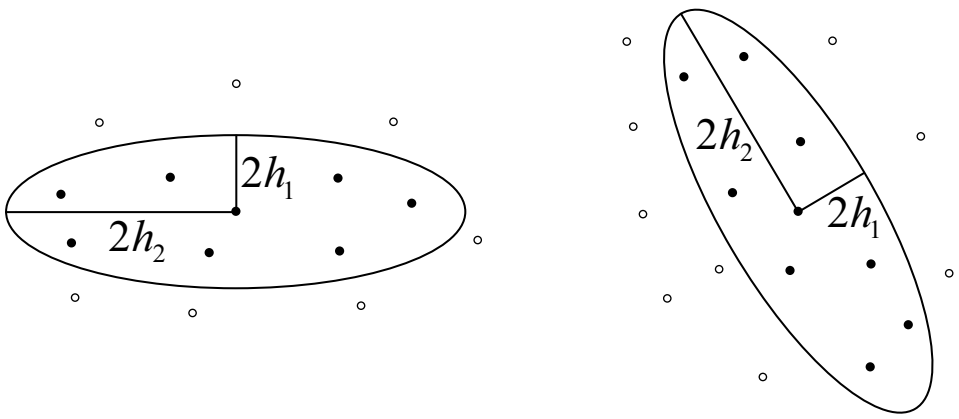


Figure 1.5: Anisotropic kernels with neighbour particles

In the thesis of Schick [113] and the papers of Shapiro and Owen [106, 116] the scalar-valued smoothing lengths h that characterise symmetric kernel functions are replaced by an anisotropic smoothing tensor $\boldsymbol{\mathcal{H}}$ that dynamically adjusts the shape of the kernel function according to the motion and distribution of the surrounding particles. These formulations have been successfully tested using shock tube problems in one and two dimensions [113]. More recently anisotropic kernels have been used in impact and material strength simulations [76] and in astrophysics to help model large scale gravitational collapse problems [106, 116].

Particle refinement methods

An alternative approach to adaptivity in meshless methods is by particle refinement algorithms. In the absence of any nodal connectivity particles are free to be added or removed without any of the implications of re-meshing. Throughout a simulation particles can be dynamically added in regions where higher accuracy is required or removed in regions of less interest.

Dynamic particle refinement is incorporated into the framework of meshless methods in two stages:

- The first stage is the derivation of suitable refinement criteria with which to identify candidate particles (or regions) for refinement.
- The second stage is the procedure by which the particles or nodes are added into the simulation, ensuring that in the process the basic properties of the simulation are conserved.

Particle refinement has been successfully introduced into the Element-Free-Galerkin method (EFGM) [53] and the Reproducing Kernel Particle method (RKPM) [80]. In both these examples regions of high gradients are identified and used to specify the areas at which nodes should undergo refinement. However, their refinement criteria are quite different. Häussler-Combe [53] uses an a posteriori error estimate based upon the interpolation error of the EFGM, whereas Liu [80] uses the theory of wavelets to decompose the RKPM solution and identify regions for refinement.

In SPH particle refinement methods have been considered to be the more complicated of the two refinement approaches. Consequently, only a comparatively small amount of research exists in the literature.

Once a SPH particle has been identified as a candidate for refinement there are several factors which need to be taken into account while devising a general procedure for particle refinement:

- The distribution of the new particles needs to be chosen and any necessary particle properties need to be assigned (eg– velocity, temperature).
- The mass of the original particle needs to be distributed between the new particles in such a way that conservation of mass is ensured.

- The addition of new particles will change the local density and velocity fields in the region surrounding the original particle. Any such change should be minimised by the refinement procedure.
- The smoothing lengths of the new particles should be assigned (reduced) to correspond to the new particle distribution.
- Regions where fine and coarse distributions of particles interact will be a consequence of the refinement process. Any refinement procedure needs to cope with such regions.
- Where possible the global properties of kinetic energy and the linear and angular momentum of the system should be conserved by the refinement procedure.

Lastiwka [70] has developed a simple strategy for adaptively inserting and removing SPH particles in one dimension. Using a refinement criterion based on the velocity gradient, particles are added in regions where the velocity gradient is high and removed were the velocity gradients are low. The new particle positions are then iteratively adjusted to ensure an even particle spacing. Corrected kernels are used to interpolate the necessary particle properties at the new locations and the particle smoothing lengths and masses are adjusted according to the updated particle distribution.

This refinement algorithm was applied to the Riemann shock tube problem in one dimension which showed some improvement with adaptivity over the standard SPH method using a comparable number of particles. However, this approach has only been implemented in one dimension and the refined distribution was found to be unstable when applied without the addition of kernel consistency corrections.

Kitsionas [61] has also applied a particle splitting algorithm in SPH, this time in three dimensions, to solve problems in astrophysics concerned with the self-gravitating collapse of a region of gas. The refinement criterion used is based on satisfying a physical requirement of the problem known as the 'Jeans Condition' which ensures that the resolution of the particle distribution is sufficient to capture the known physics of the problem.

When refined, particles are replaced by thirteen child particles each positioned on the nodes of a hexagonal lattice centred on the parent particle. The smoothing

length of the child particles are fixed in proportion to parent particle and the mass of the parent particle is equally distributed between each of the child particles. All that remains is to calculate the optimal particle separation for the child particles and Kitsionas [62] obtained values for this parameter in one of two ways.

The first approach was to use the particle separation which minimised the local difference in density resulting from the refinement of a single particle. While the second approach was to study the affect particle refinement had on a large collection of particles. This was achieved by taking a large number of particles of very uniform density and simultaneously refining them. The optimal particle separation was then obtained as the one which corresponded to the refined distribution which took the least amount of time to resettle back to a uniform density. The results obtained from both these methods were inconclusive. It was found that the particle separation obtained from the second set of experiments helped lessen the global effect of the particle refinement and as such these parameters were used in the subsequent simulations.

Conclusion

While the above examples have made some progress towards understanding particle refinement in SPH it is the author's opinion that current research does not satisfactorily address all of the necessary considerations that have been discussed at the start of this section. In particular there has been no quantitative study into the errors introduced due to the refinement of particles and consequently, there has been no reliable way to assess the relative performance of a given refinement algorithm.

It is the aim of this thesis is provide a rigourous framework for the analysis of general particle refinement strategies and to answer the important question of whether it is possible to derive a fully conservative refinement algorithm for SPH simulations.

1.4 Layout of the thesis

The remainder of the thesis is divided into the following chapters:

Chapter 2: SPH for fluid dynamics problems

This first chapter presents the fundamentals of the SPH method as applied to fluid dynamics. Properties of the integral and summation approximations of a function are derived and the concept of consistency is introduced in relation to the discrete SPH equations. The various traditional forms of the discrete SPH equations for Newtonian fluids are then derived. To complete the chapter the required equation of state, timestepping schemes and nearest neighbour search algorithms are given.

Chapter 3: Corrected SPH and stabilization

The concept of kernel and gradient correction is introduced in Chapter 2. Constant and linear consistency of the discrete SPH equations is enforced with the addition of correction terms to both the kernel and its gradient. Hessian stabilization is presented as a method to add higher order terms into the expression for the gradient of functions and will be used later in Chapter 7. Finally, it is shown that with kernel corrections the SPH method conserves both linear and angular momentum without the restriction of uniform particle smoothing lengths.

Chapter 4: Variational formulation of SPH

In order to implement particle refinement into the SPH framework the underlying formulation must be able to cope with non-uniform particle masses and smoothing lengths. In this chapter such a formulation is derived from variational principles. A boundary contact force term is introduced in the process which will be developed and implemented Chapter 5. The resulting expressions for the internal forces are found to take the same form as those derived in Chapter 2. Finally, this variational formulation is shown to conserve both the linear and angular momentum of the system.

Chapter 5: Boundary methods

In the past various different approaches have been used to implement boundary conditions into the SPH formulation. In the first section of this chapter four commonly used approaches are described and discussed: the bounce back method, image particles, penalty methods and Lennard-Jones potentials. In the remaining sections a novel method for exactly calculating the variational boundary contact force derived in Chapter 4 is presented. Several examples utilising this new contact force are given and the accuracy of this new approach is verified.

Chapter 6: Adaptivity

The general principles of adaptivity in SPH are introduced in this chapter forming the basis of a variable resolution SPH formulation. A simple refinement strategy based upon particle splitting is developed and the concepts of density and velocity refinement errors are defined. The density refinement error is then minimised with the appropriate choice for the refined particle masses via the solution of a model problem. This solution is shown to be independent of the initial unrefined particle mass and smoothing length. Conservation properties of the refinement process in SPH simulations are discussed and it is proved that there is only one fully conservative velocity configuration that the refined particles can take.

Chapter 7: Refinement simulations

In this chapter all the essential ingredients are brought together and incorporated into a single, flexible variable resolution SPH code including dynamic particle refinement. Four two dimensional fluid flows are then used to validate the refinement procedure and the new boundary contact force implementation. The accuracy of the refinement procedure is first investigated using the Couette and Poiseuille flows for which analytic solutions are available. The second set of simulations consist of two more complex flows; the first example models the flow separation through a funnel while the second models an emptying tank with two small outlets on its side.

Chapter 8: Conclusions and future research

To conclude, the implementation of dynamic refinement into the existing SPH framework is assessed and summarised, including recommendations for future research.

Chapter 2

SPH for fluid dynamics problems

2.1 Introduction to SPH

Developed over twenty years ago, Smoothed Particle Hydrodynamics (SPH) is one of the most established meshless methods. SPH is a simple and robust numerical method which has been used to solve a remarkable variety of problems in the field of computational fluid dynamics.

This chapter provides an introduction to the fundamentals of the SPH method and describes the procedure for the discretization and solution of general partial differential equations using the SPH formulation.

The reproducing kernel approximation of a function, from which the discrete SPH approximations are derived, is introduced as the basis of the SPH method. Important properties of the kernel functions are presented. In particular, the kernel consistency conditions which ensure the consistency of the integral approximations are emphasised.

By directly applying the integral and summation approximations, the traditional discrete SPH forms for the continuity, momentum and energy equations for Newtonian fluids are derived.

The chapter concludes by presenting the required equation of state, timestepping schemes and nearest neighbour search algorithms necessary for the simulation of incompressible fluid flows using SPH.

2.2 Integral approximation

Smoothed Particle Hydrodynamics (SPH) is based on the simple integral identity

$$f(\mathbf{x}) = \int_{\Omega} f(\mathbf{x}')\delta(\mathbf{x}-\mathbf{x}')\,d\mathbf{x}'$$
$$\text{where } \delta(\mathbf{x}-\mathbf{x}') = \begin{cases} \infty & \mathbf{x}=\mathbf{x}' \\ 0 & \mathbf{x}\neq\mathbf{x}' \end{cases} \text{ (Dirac delta function)}. \tag{2.1}$$

By approximating the delta function by a suitable *kernel function* the *reproducing kernel approximation* of an arbitrary function f is obtained as

$$\langle f(\mathbf{x})\rangle = \int_{\Omega} f(\mathbf{x}')\,w(\mathbf{x}-\mathbf{x}',h)\,d\mathbf{x}' \tag{2.2}$$

where $\langle\cdot\rangle$ denotes the reproduced function approximation, $w(\mathbf{x}-\mathbf{x}',h)$ is the kernel function and h is the *smoothing length* (or *dilation parameter*) that defines the domain of influence of the kernel (see Figure 2.1). Kernel functions play a vital role in the SPH method and so will be discussed in greater detail in Section 2.4.

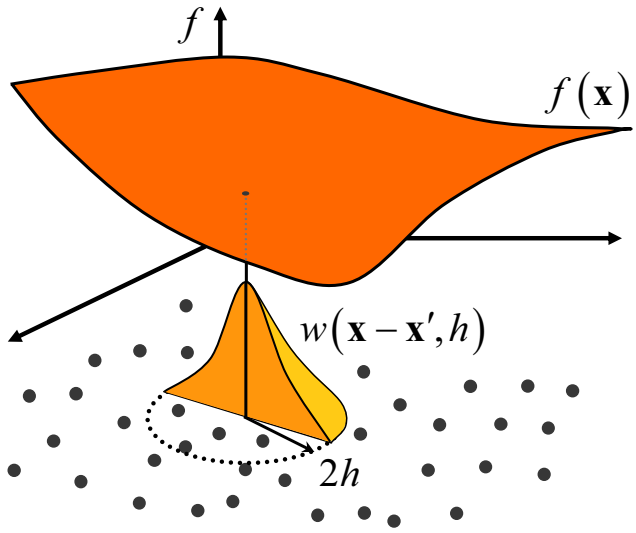


Figure 2.1: Reproducing kernel approximation of f over the whole domain

In general $\langle f(\mathbf{x})\rangle \neq f(\mathbf{x})$ since an error term will be introduced due to the substitution of the kernel function w in place of the Dirac delta function.

$$\langle f(\mathbf{x})\rangle = f(\mathbf{x}) + E\langle f,\mathbf{x}\rangle \tag{2.3}$$

where $E\langle f,\mathbf{x}\rangle$ is the error introduced in the approximation of f at the point $\mathbf{x}$.

It will be shown that if the kernel function is even, normalised, and has compact support then the reproducing kernel approximation is second order with respect to h [45, 92]. That is, for some small bounded constant e_k,

$$|E\langle f, \mathbf{x}\rangle| \leqslant e_k h^2. \tag{2.4}$$

Integral approximation of the gradient of a function

By applying the definition of the reproducing kernel approximation to the gradient of scalar function f and by invoking Gauss's theorem yields

$$\langle \nabla f(\mathbf{x})\rangle = \int_\Omega \nabla_{\mathbf{x}'} f(\mathbf{x}')\, w(\mathbf{x}-\mathbf{x}', h)\, d\mathbf{x}' \tag{2.5}$$

$$= \int_{\partial\Omega} f(\mathbf{x}')\, w(\mathbf{x}-\mathbf{x}', h)\, \mathbf{n}\, dS - \int_\Omega f(\mathbf{x}')\, \nabla_{\mathbf{x}'} w(\mathbf{x}-\mathbf{x}', h)\, d\mathbf{x}'. \tag{2.6}$$

Assuming that the kernel w has a compact support then the above integrals are taken only over the region where $w \neq 0$. If $\mathbf{x}$ is sufficiently far away from any boundaries so that the support of w is entirely contained in Ω the contribution from the surface integral is zero and

$$\langle \nabla f(\mathbf{x})\rangle = -\int_\Omega f(\mathbf{x}')\, \nabla_{\mathbf{x}'} w(\mathbf{x}-\mathbf{x}', h)\, d\mathbf{x}'. \tag{2.7}$$

It can be seen that the integral approximation has transferred the gradient operation from the function onto the kernel. In addition, if w is an even function $w(\mathbf{x}-\mathbf{x}', h) = w(\mathbf{x}'-\mathbf{x}, h)$ then $\nabla_{\mathbf{x}'} w(\mathbf{x}-\mathbf{x}', h) = -\nabla_{\mathbf{x}} w(\mathbf{x}-\mathbf{x}', h)$ and the expression for the integral approximation for the gradient can be simplified to

$$\langle \nabla f(\mathbf{x})\rangle = \nabla_{\mathbf{x}} \left(\int_\Omega f(\mathbf{x}')\, w(\mathbf{x}-\mathbf{x}', h)\, d\mathbf{x}' \right) = \nabla \langle f(\mathbf{x})\rangle. \tag{2.8}$$

Integral approximation of the divergence of a function

The reproducing kernel approximation of the divergence of a vector function $\mathbf{F}$ can be derived by applying the Divergence theorem to the vector identity

$\nabla \cdot (a\mathbf{F}) = a(\nabla \cdot \mathbf{F}) + \mathbf{F} \cdot \nabla a$ to give

$$\begin{aligned} \langle \nabla \cdot \mathbf{F}(\mathbf{x}) \rangle &= \int_\Omega [\nabla_{\mathbf{x}'} \cdot \mathbf{F}(\mathbf{x}')] \, w(\mathbf{x} - \mathbf{x}', h) \, d\mathbf{x}' \qquad (2.9) \\ &= \int_{\partial\Omega} \mathbf{F}(\mathbf{x}') \, w(\mathbf{x} - \mathbf{x}', h) \cdot \mathbf{n} \, dS - \int_\Omega \mathbf{F}(\mathbf{x}') \cdot [\nabla_{\mathbf{x}'} w(\mathbf{x} - \mathbf{x}', h)] \, d\mathbf{x}' \\ &= -\int_\Omega \mathbf{F}(\mathbf{x}') \cdot [\nabla_{\mathbf{x}'} w(\mathbf{x} - \mathbf{x}', h)] \, d\mathbf{x}'. \qquad (2.10) \end{aligned}$$

As before the expression for the integral approximation for the divergence can be simplified if the kernel function is even to give

$$\langle \nabla \cdot \mathbf{F}(\mathbf{x}) \rangle = \nabla_{\mathbf{x}} \cdot \int_\Omega \mathbf{F}(\mathbf{x}') \, w(\mathbf{x} - \mathbf{x}', h) \, d\mathbf{x}' = \nabla \cdot \langle \mathbf{F}(\mathbf{x}) \rangle . \qquad (2.11)$$

Integration approximation of a product of functions

Finally, using equation (2.3), the product of reproducing kernel approximations $\langle f \rangle \langle g \rangle$ is given by

$$\begin{aligned} \langle f \rangle \langle g \rangle &= (f + E\langle f \rangle)(g + E\langle g \rangle) \\ &= fg + gE\langle f \rangle + fE\langle g \rangle + E\langle f \rangle E\langle g \rangle \\ &= \langle fg \rangle - E\langle fg \rangle + gE\langle f \rangle + fE\langle g \rangle + E\langle f \rangle E\langle g \rangle . \qquad (2.12) \end{aligned}$$

In other words to within the order of accuracy of the method the product of reproducing kernel approximations is equal to the reproducing kernel approximation of the product.

$$\langle fg \rangle = \langle f \rangle \langle g \rangle . \qquad (2.13)$$

In summary, the reproducing kernel approximation operator $\langle \cdot \rangle$ is linear (and commutative) and as such it satisfies the following properties

$$\begin{aligned} \langle f_1 + f_2 \rangle &= \langle f_1 \rangle + \langle f_2 \rangle , \qquad (2.14a) \\ \langle f_1 f_2 \rangle &= \langle f_1 \rangle \langle f_2 \rangle , \qquad (2.14b) \\ \langle c f_1 \rangle &= c \langle f_1 \rangle , \qquad (2.14c) \\ \langle \nabla f \rangle &= \nabla \langle f \rangle , \qquad (2.14d) \\ \langle \nabla \cdot \mathbf{F} \rangle &= \nabla \cdot \langle \mathbf{F} \rangle . \qquad (2.14e) \end{aligned}$$

2.3 Summation approximation

In order to develop a practical numerical scheme equation (2.2) is discretized to give

$$\begin{aligned} \langle f(\mathbf{x})\rangle \approx f_h(\mathbf{x}) &= \sum_{b\in M_{\mathbf{x}}} V_b f(\mathbf{x}_b)\, w(\mathbf{x}-\mathbf{x}_b, h_b) \\ &= \sum_{b\in M_{\mathbf{x}}} V_b f(\mathbf{x}_b)\, w_b(\mathbf{x}, h_b) \end{aligned} \tag{2.15}$$

where V_b is a *volume* associated to the point b and $w_b(\mathbf{x}, h_b) := w(\mathbf{x}-\mathbf{x}_b, h_b)$ is the kernel based at point b with corresponding smoothing length h_b.

Here $M_{\mathbf{x}}$ is the set of neighbouring points that contribute to the summation. Choosing a kernel with a compact support means that $M_{\mathbf{x}}$ will be finite and the summation will be over a small number of neighbouring points only, as shown in Figure 2.2.

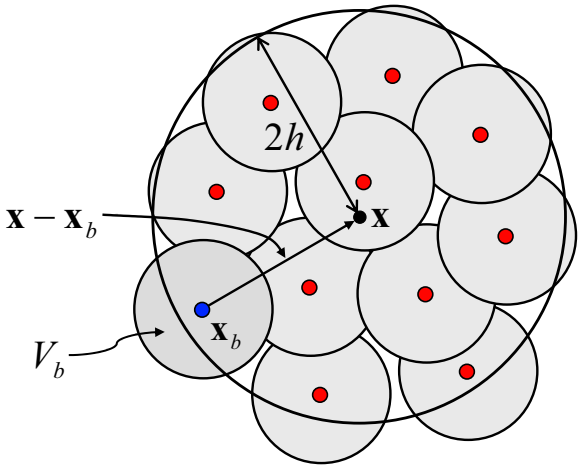


Figure 2.2: Summation approximation of f at point $\mathbf{x}$

In particular the interpolation of the density $\rho(\mathbf{x})$ of a continuum is given by

$$\rho_h(\mathbf{x}) = \sum_{b\in M_{\mathbf{x}}} m_b w_b(\mathbf{x}, h_b), \qquad m_b = V_b \rho(\mathbf{x}_b). \tag{2.16}$$

An expression for the volume derived from Monte-Carlo theory [95] is given by

$$V_a^{-1} = \sum_{b\in M_a} w(\mathbf{x}_a - \mathbf{x}_b, h_b). \tag{2.17}$$

Due to the Lagrangian nature of SPH these interpolation points can be interpreted as discrete particles moving with the continuum in question. The material response can therefore be visualised by tracking these moving interpolation points.

With this interpretation a given point (or particle) has an associated *mass density* $\rho(\mathbf{x}_a)$ and the volume V_a associated with particle a can be expressed as

$$V_a = \frac{m_a}{\rho(\mathbf{x}_a)} \,. \tag{2.18}$$

Writing (2.15) in terms of shape functions gives

$$f_h(\mathbf{x}) = \sum_{b \in M_{\mathbf{x}}} f(\mathbf{x}_b) N_b(\mathbf{x}), \qquad N_b(\mathbf{x}) = V_b w_b(\mathbf{x}, h_b) \,. \tag{2.19}$$

The gradient of f_h is now obtained from the point values of the function in terms of the gradient of the SPH shape functions

$$\nabla f_h(\mathbf{x}) = \sum_{b \in M_{\mathbf{x}}} f(\mathbf{x}_b) \nabla N_b(\mathbf{x}), \qquad \nabla N_b(\mathbf{x}) = V_b \nabla w_b(\mathbf{x}, h_b) \,. \tag{2.20}$$

Unlike their finite element counterparts SPH shape functions do not possess the interpolation property given by $N_b(\mathbf{x}_a) = \delta_{ab}$. Consequently, SPH approximations do not exactly interpolate the solution at particle points $f_h(\mathbf{x}_b) \neq f(\mathbf{x}_b)$ and Dirichlet type boundary conditions are not naturally incorporated in SPH formulations (see Figure 2.3).

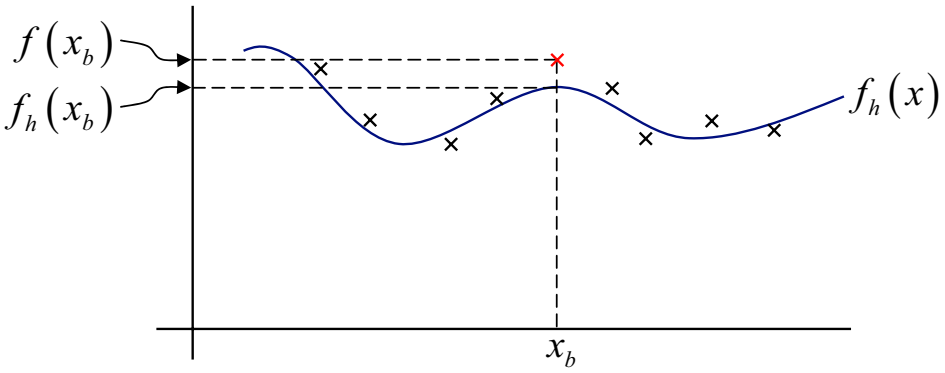


Figure 2.3: SPH interpolation of a function f

In general $f_h(\mathbf{x}) \neq \langle f(\mathbf{x}) \rangle$ since another error term has been introduced by the pointwise approximation of the integration in equation (2.15)

$$f_h(\mathbf{x}) = \langle f(\mathbf{x}) \rangle + E_h(f, \mathbf{x}) \tag{2.21}$$

where $E_h(f, \mathbf{x})$ is the error introduced in the approximation of $\langle f \rangle$ at the point $\mathbf{x}$.

When convenient the summation notation will be simplified from

$$\sum_{b \in M_{\mathbf{x}}} (\cdots) \quad \text{to} \quad \sum_{b} (\cdots) \tag{2.22}$$

with the understanding that the summation over b is over all contributing interpolation points in the vicinity of $\mathbf{x}$.

2.4 Kernel functions

In the previous sections some results depended on the assumption that the kernel w was an even function with a compact support. Theoretically there is no restriction to the choice of kernel function used in SPH. However, in normal practice there is a minimum set of requirements that need to be met [46].

A wide variety of kernel functions have been used in SPH. The most common kernels being spline or Gaussian based functions. The choice of kernel function can be likened to the choice of discretization in finite difference methods and the smoothing length can be interpreted as varying the element size in finite element methods.

- **Compact support**

All kernel functions in this thesis are assumed to have compact supports defined by

$$w(r, h) = 0 \quad \text{when} \quad r = \|\mathbf{x} - \mathbf{x}'\| \geqslant Kh \tag{2.23}$$

where K is a constant. It is common to take $K = 2$ and the smoothing length $h = \alpha \times \eta$ where $\alpha \approx 1.2 - 2.0$ and η is the average initial particle separation. The importance of the smoothing length will be discussed further in Section 2.4.3.

Consequently, due to the compact support of the kernel function the whole domain Ω in equation (2.2) can be replaced by the support of the kernel function based at the point $\mathbf{x}$ denoted by $B(\mathbf{x}, Kh)$ (typically, a ball of radius Kh centred about the point $\mathbf{x}$)

$$\langle f(\mathbf{x}) \rangle = \int_{B(\mathbf{x}, Kh)} f(\mathbf{x}')\, w(\mathbf{x} - \mathbf{x}', h)\, d\mathbf{x}'. \tag{2.24}$$

This has the effect of reducing the integral approximation from a *global* approximation to a *local* approximation.

Gaussian based kernels do not have a compact support so in theory all particles contribute to the summation approximation of a function. In practice Gaussian kernels are truncated since they satisfy $w(r) \to 0$ as $r \to \infty$ and contributions from particles sufficiently far away can be ignored without consequence.

- **Even**

An even kernel ensures that all equally spaced particles with identical smoothing lengths will interact symmetrically. This property has already been used to simplify the integral approximations and where possible it will be used to simplify the discrete governing equations under the assumption of uniform smoothing lengths.

- **Positive and monotonically decreasing**

The kernel function should be a strictly positive-valued and monotonically decreasing function. Positivity of the kernel function ensures that the summation approximation of a function is formed from an average of positively weighted point values which results in a physically meaningful numerical method. While a monotonically decreasing kernel function ensures that the strength of interaction between particle pairs decreases as the particle separation increases.

- **Delta function limit**

To ensure the reproducing kernel approximation approaches the desired function as the smoothing length is reduced

$$\lim_{h \to 0} \langle f(\mathbf{x}) \rangle = f(\mathbf{x}) \tag{2.25}$$

it is essential for the kernel function to approach the Dirac delta function as the smoothing length is reduced (see Figure 2.4)

$$\lim_{h \to 0} w(\mathbf{x} - \mathbf{x}', h) = \delta(\mathbf{x} - \mathbf{x}') . \tag{2.26}$$

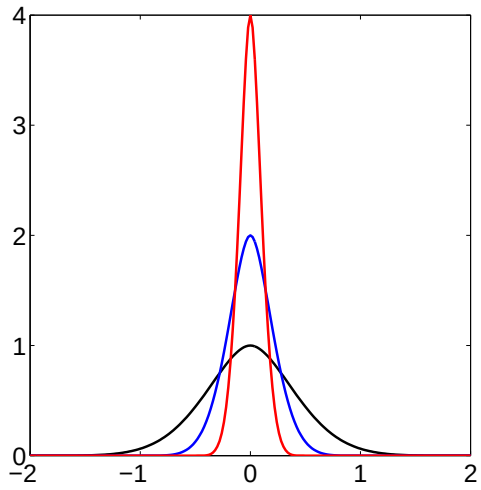


Figure 2.4: 1D Quintic Spline Kernel for $h = 1.0,\ 0.5,\ 0.25$

- **Consistency**

In order for the reproducing kernel approximation to exactly approximate constant functions the following zero order consistency condition must hold

$$\langle 1 \rangle = \int_{\Omega} 1 \cdot w\left(\mathbf{x} - \mathbf{x}', h\right) d\mathbf{x}' = 1. \tag{2.27}$$

If the integral of the kernel over the domain is normalised such that it is equal to 1 then the kernel is said to satisfy *zero order consistency*

$$\int_{\Omega} w\left(\mathbf{x} - \mathbf{x}', h\right) d\mathbf{x}' = 1. \tag{2.28}$$

Higher order consistency can be satisfied with a careful choice of kernel function. For example in the 1-D case a Taylor series expansion about x yields

$$f\left(x'\right) = f\left(x\right) + \left(x' - x\right) f'\left(x\right) + \frac{1}{2}\left(x' - x\right)^2 f''\left(x\right) + \cdots . \tag{2.29}$$

Substituting the above expression into equation (2.2) gives

$$\langle f\left(x\right) \rangle = f\left(x\right) \int_{\Omega} w\left(x - x'\right) dx' - f'\left(x\right) \int_{\Omega} \left(x - x'\right) w\left(x - x'\right) dx' + \cdots . \tag{2.30}$$

If the following *consistency conditions* are satisfied

$$\int_{\Omega} w\left(x\right) dx = 1, \tag{2.31a}$$

$$\int_{\Omega} x^j w\left(x\right) dx = 0 \qquad \text{for} \quad 0 < j \leqslant k \tag{2.31b}$$

then the reproducing kernel approximation is said to be of *order* k.

In this case $\langle f(x)\rangle = f(x)$ for any polynomial function of degree less than or equal to k and the reproducing kernel approximation satisfies

$$\langle f(x)\rangle = f(x) + \frac{1}{k+1} f^{(k+1)}(x)\, \mathcal{O}\left(h^{k+1}\right). \tag{2.32}$$

When the point integration is applied to the reproducing kernel approximation as in equation (2.15) any consistency conditions the kernel may have possessed will no longer be satisfied exactly by the discrete summation approximations. In Chapter 3 simple and effective methods for improving the accuracy and consistency of the discrete SPH equations are presented.

2.4.1 Evaluating the gradient of the kernel function

In general kernel functions are written in the form

$$w(r,h) = \frac{\alpha_d}{h^d} f(r), \qquad r = \|\mathbf{x} - \mathbf{x}'\| \in [0, Kh] \tag{2.33}$$

where α_d scales the kernel function to enforce the zero order consistency condition and d is the number of dimensions.

$$w_b(\mathbf{x},h) := w(\mathbf{x} - \mathbf{x}_b, h) = w_b(r) \quad \text{where} \quad r^2 = (\mathbf{x} - \mathbf{x}_b)\cdot(\mathbf{x} - \mathbf{x}_b). \tag{2.34}$$

The gradient of the kernel function in terms of r is calculated from

$$\nabla w_b(r) = \frac{dw_b}{dr} \nabla r \tag{2.35}$$

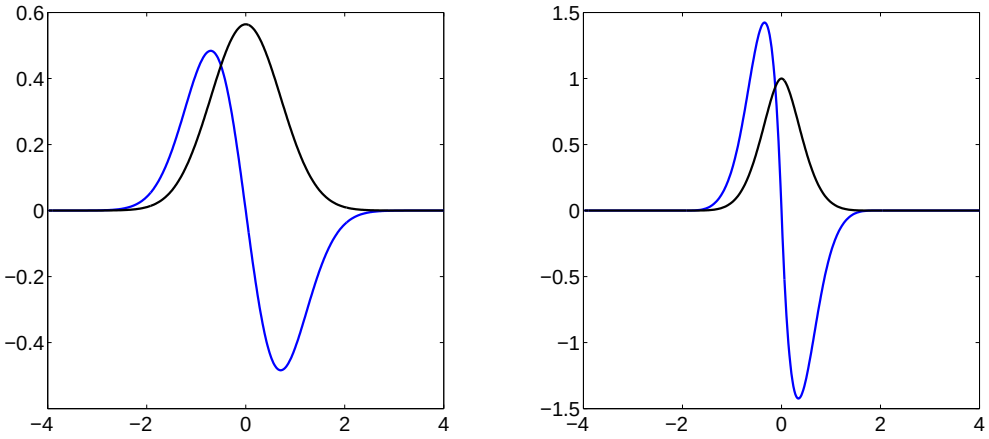
where ∇r is obtained from the identity $\nabla(r^2) = 2r\nabla r = 2(\mathbf{x} - \mathbf{x}_b)$.

The final expression for the gradient is obtained as

$$\nabla w_b(r) = \frac{1}{r}\frac{dw}{dr}(\mathbf{x} - \mathbf{x}_b). \tag{2.36}$$

2.4.2 Example kernel functions

The Gaussian and quintic kernel functions with their corresponding normalising coefficients α_d for one, two and three dimensions are given below. Graphs of these kernels and their derivatives are plotted in Figure 2.5.



Gaussian kernel Quintic kernel

Figure 2.5: 1D kernel functions with $h = 1$ (derivatives shown in blue.)

Gaussian kernel function

The Gaussian has been a popular choice for kernel function since it is infinity differentiable and the derivatives share the same exponential form. However, it doesn't have a compact support so in practice the kernel domain is truncated.

$$\begin{aligned} w(r,h) &= \alpha_d \, e^{-\left(\frac{r}{h}\right)^2} && r \geqslant 0, \\ \frac{dw}{dr}(r,h) &= \alpha_d \left(-\frac{2r}{h^2} e^{-\left(\frac{r}{h}\right)^2} \right) && r \geqslant 0, \end{aligned} \tag{2.37}$$

$$\text{where} \quad \alpha_{1D} = \frac{1}{\sqrt{\pi} h} \quad , \quad \alpha_{2D} = \frac{1}{h^2 \pi} \quad , \quad \alpha_{3D} = \frac{1}{h^3 \pi^{\frac{3}{2}}} \, .$$

Quintic kernel function

The quintic kernel function is a fifth order spline based kernel function and is used in all of the simulations presented in the later chapters of this thesis.

$$w(r,h) = \alpha_d \begin{cases} \left(2 - \frac{r}{h}\right)^5 - 16\left(1 - \frac{r}{h}\right)^5 & 0 \leqslant r \leqslant h \\ \left(2 - \frac{r}{h}\right)^5 & h < r \leqslant 2h \\ 0 & r > 2h \end{cases} ,$$

$$\frac{dw}{dr}(r,h) = \alpha_d \begin{cases} -\frac{5}{h}\left(2 - \frac{r}{h}\right)^4 + \frac{80}{h}\left(1 - \frac{r}{h}\right)^4 & 0 \leqslant r \leqslant h \\ -\frac{5}{h}\left(2 - \frac{r}{h}\right)^4 & h < r \leqslant 2h \\ 0 & r > 2h \end{cases} , \tag{2.38}$$

$$\text{where} \quad \alpha_{1D} = \frac{1}{16h} \quad , \quad \alpha_{2D} = \frac{3}{16h^2 \pi} \quad , \quad \alpha_{3D} = \frac{7}{40h^3 \pi} \, .$$

2.4.3 Smoothing length

The ability to have smoothing lengths that vary in both space and time is one of the attractive properties of SPH. The kernel functions in SPH provides a means to transform the point mass description of the continuum into a continuous representation. The smoothing length h governs the support of the kernel function and consequently the amount of smoothing that is applied to the SPH particles.

It is common to choose the initial smoothing length to be proportional to the mean inter-particle distance $h \propto \bar{\eta}$. In two and three dimensions this is given by

$$h = \alpha\bar{\eta} \quad \text{where} \quad \bar{\eta} = \sqrt{\frac{A}{N}} \text{ in 2D} \quad \text{or} \quad \bar{\eta} = \sqrt[3]{\frac{V}{N}} \text{ in 3D} \tag{2.39}$$

where α is a constant typically $\alpha \approx 1.2 - 2.0$, N is the total number of particles in the simulation and A (or V) is the initial area (or volume) of the problem domain.

In problems where a large amount of compression or expansion occur the number of neighbours of a given particle may change as the problem evolves. In such cases it is necessary for a particles smoothing length to evolve as required. A particle under compression may have an excessively large number of neighbours making the method computational inefficient, therefore a reduced smoothing length would be desirable. While a particle with too few neighbours would benefit from an increased smoothing length in order to keep the solution to within the desired accuracy.

A simple way to evolve the smoothing length of a particle in space and time is to adjust h_a according to its current density.

$$h_a = h_0 \left(\frac{\rho_0}{\rho_a}\right)^{\frac{1}{d}} \tag{2.40}$$

where h_0 is the initial smoothing length of the particle, ρ_0 is the material density of the particle, ρ_a is the current density of particle a, and d is the dimension of the problem.

An inconsistency arises here since the particle density ρ_a is itself a function of the smoothing length and is therefore highly non-linear. A solution to this problem is to calculate the density and then update the smoothing length, repeating this until the values converge [111]. This may take several iterations per particle to occur and so can be inefficient to implement.

Generally, a small smoothing length relative to the particle spacing results in irregular oscillatory interpolation, while a large smoothing length results in excessive smoothing of the interpolation. This can be seen in Figure 2.6 which shows the resulting interpolated density for several different values for the smoothing length. The theoretical continuous density distribution should be constant $\rho = 1$.

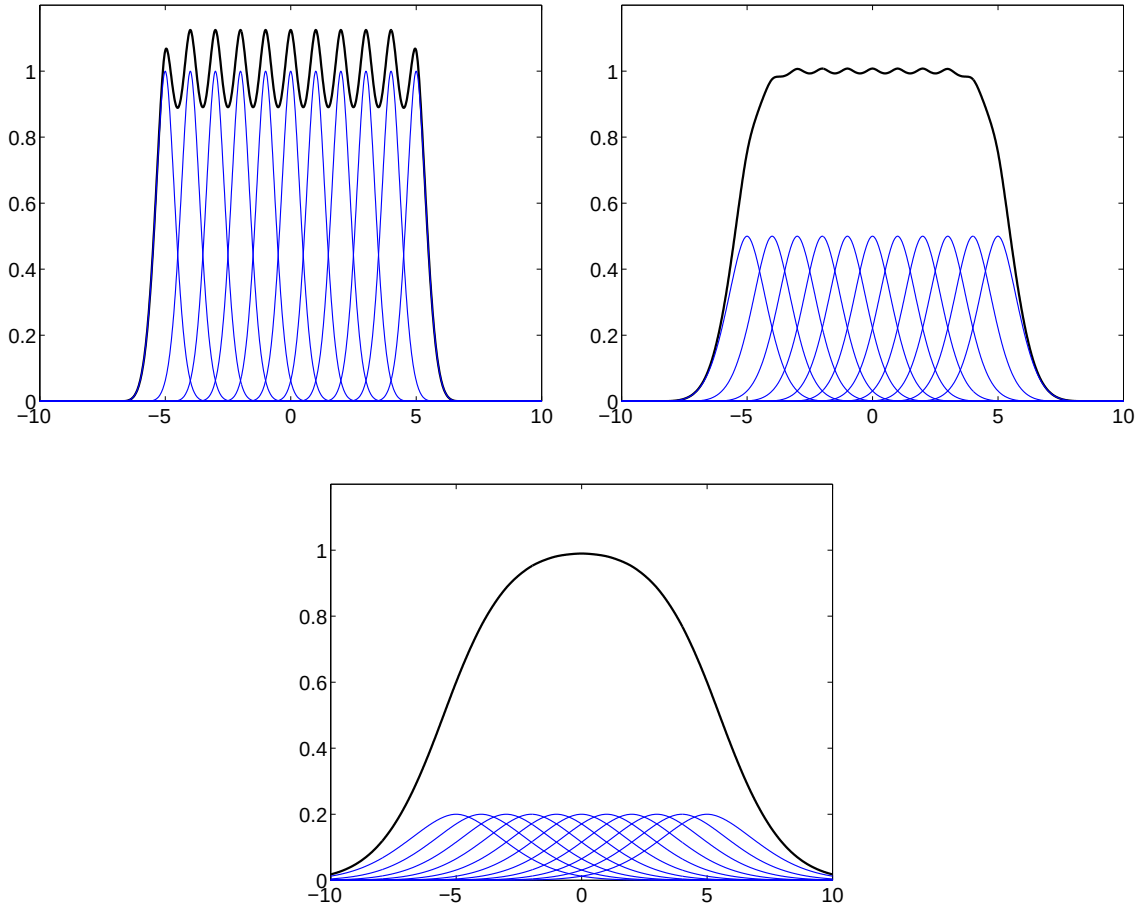


Figure 2.6: Density profiles for different values of smoothing length ($h = 1.0, 2.0, 5.0$)

2.5 Discrete SPH equations for fluids

Discrete SPH equations are not unique and different forms can derived using a variety of different approaches. In this section the three governing equations of fluid dynamics are discretized by directly applying the integral and summation approximations as defined in this chapter.

In Chapter 4 it will be shown that the same discrete equations can be derived by following a corresponding variational approach.

2.5.1 Continuity equation

The density can be calculated in two ways. The first and most simple method is to directly apply the summation approximation to the density field. While the second method calculates the derivative of the density using the continuity equation.

Direct density evaluation

Applying the summation approximation directly to the density field yields

$$\begin{aligned} \langle \rho(\mathbf{x}_a) \rangle &\approx \sum_b V_b \rho_b w_b(\mathbf{x}_a, h_b), \\ \rho_a &\approx \sum_b m_b w_b(\mathbf{x}_a, h_b) \end{aligned} \tag{2.41}$$

where $w_b(\mathbf{x}_a, h_b) = w(\mathbf{x}_a - \mathbf{x}_b, h_b)$ and h_b is the smoothing length associated with particle b.

From now on the $\approx$ and $\langle \cdot \rangle$ symbols will be dropped and replaced by $=$ with the understanding that the discrete SPH equations only approximate the governing partial differential equations.

Continuity method

Two different forms for the rate of change of density can be derived from the continuity equation which in continuum form is given by

$$\frac{D\rho}{Dt} = -\rho \, \nabla \cdot \mathbf{v}.$$

When the divergence of velocity term in the continuity equation is discretized using the SPH approximation $\dot{\rho}_a \approx -\rho_a \langle \nabla \cdot \mathbf{v}_a \rangle$ the first discrete SPH equation for $\dot{\rho}$ is obtained as

$$\dot{\rho}_a = -\rho_a \sum_b V_b \mathbf{v}_b \cdot \nabla w_b (\mathbf{x}_a, h_b). \tag{2.42}$$

An additional velocity difference can be introduced into the above equation for $\dot{\rho}$ by noting that to within the order of accuracy of the approximation the following summation vanishes

$$\langle \nabla 1 \rangle = \int_\Omega 1 \times \nabla w (\mathbf{x} - \mathbf{x}', h)) d\mathbf{x}' \approx \sum_b V_b \nabla w_b (\mathbf{x}, h_b) \approx 0. \tag{2.43}$$

By adding this zero term, $\rho_a \mathbf{v}_a \cdot \sum_b V_b \nabla w_b (\mathbf{x}_a, h_b)$ to equation (2.42) gives a second form for $\dot{\rho}$

$$\dot{\rho}_a = \rho_a \sum_b V_b (\mathbf{v}_a - \mathbf{v}_b) \cdot \nabla w_b (\mathbf{x}_a, h_b). \tag{2.44}$$

It will be shown in Chapter 3 that if the kernel function w has been corrected then equation (2.43) will be exactly zero and these two discretizations of the continuity equation are identical.

The second derivation uses the identity $\dot{\rho} = -\rho \, \nabla \cdot \mathbf{v} = -(\nabla \cdot (\rho \mathbf{v}) - \mathbf{v} \cdot (\nabla \rho))$ which results in the SPH approximation $\dot{\rho}_a \approx \mathbf{v}_a \cdot \langle \nabla \rho_a \rangle - \langle \nabla \cdot (\rho_a \mathbf{v}_a) \rangle$ for the continuity equation and the following discrete form is obtained

$$\begin{aligned} \dot{\rho}_a &= \mathbf{v}_a \cdot \sum_b V_b \rho_b \nabla w_b (\mathbf{x}_a, h_b) - \sum_b V_b (\rho_b \mathbf{v}_b) \cdot \nabla w_b (\mathbf{x}_a, h_b) \\ &= \sum_b m_b (\mathbf{v}_a - \mathbf{v}_b) \cdot \nabla w_b (\mathbf{x}_a, h_b). \end{aligned} \tag{2.45}$$

The smoothing property of the continuity equation

As particles approach each other their relative velocities and $\nabla w_b (\mathbf{x}_a, h_b)$ will be negative and as such will add positive contributions to $\dot{\rho}_a$. If in total $\dot{\rho}_a > 0$ the particle's density will increase. This will lead to an increase in pressure through the equation of state which in turn pushes the particles apart.

It is this interplay between velocity and density/pressure that ensures an approximately uniform density field and that particles remain on average equally spaced.

Equivalence of continuity and direct density formulations

It was noted by Vila [127] that the two density discretizations given by equation (2.41) and equation (2.45) are equivalent. Writing both as functions of the current time t gives

$$\rho_a(t) = \sum_b m_b w(r_{ab}(t)), \tag{2.46}$$

$$\dot{\rho}_a(t) = \sum_b m_b (\mathbf{v}_a(t) - \mathbf{v}_b(t)) \cdot \nabla w(r_{ab}(t)) \tag{2.47}$$

where $r_{ab}(t) \in [0, Kh]$ and $r_{ab}^2(t) = (\mathbf{x}_a(t) - \mathbf{x}_b(t)) \cdot (\mathbf{x}_a(t) - \mathbf{x}_b(t))$.

Differentiating equation (2.46) with respect to time gives

$$\dot{\rho}_a(t) = \frac{d}{dt}(\rho_a) = \sum_b m_b \frac{d}{dt}[w(r_{ab}(t))] = \sum_b m_b \frac{dr_{ab}}{dt}\frac{dw}{dr_{ab}}. \tag{2.48}$$

This can be shown to be equal to equation (2.47) by substituting

$$\frac{dr_{ab}}{dt} = \frac{(\mathbf{x}_a(t) - \mathbf{x}_b(t)) \cdot (\mathbf{v}_a(t) - \mathbf{v}_b(t))}{r_{ab}} \tag{2.49}$$

and

$$\nabla w(r_{ab}) = \frac{1}{r_{ab}}\frac{dw}{dr_{ab}}(\mathbf{x}_a - \mathbf{x}_b) \tag{2.50}$$

into equation (2.48).

More precisely, it has be shown that for any constant $K \in \mathbb{R}$

$$\begin{aligned} F &= \sum_b m_b w(r_{ab}(t)) + K \\ f &= -\sum_b m_b (\mathbf{v}_b(t) - \mathbf{v}_a(t)) \cdot \nabla w(r_{ab}(t)) \end{aligned} \qquad \text{where } \dot{F} = f. \tag{2.51}$$

By the fundamental theorem of calculus

$$\int_{t_0}^{t} f(s)\,ds = F(t) - F(t_0) = \sum_b m_b w(r_{ab}(t)) - \sum_b m_b w(r_{ab}(t_0)). \tag{2.52}$$

Therefore,

$$\rho_a(t) - \rho_a(t_0) = \sum_b m_b w(r_{ab}(t)) - \rho_a^0. \tag{2.53}$$

Thus, the two formulations will coincide when the initial density distribution is chosen to be

$$\rho_a(t_0) = \sum_b m_b w(r_{ab}(t_0)) \qquad \text{for each } a. \tag{2.54}$$

In practice due to the explicit time integration of the ordinary differential equations one should expect to obtain different numerical results from each formulation.

2.5.2 Momentum equation

This section will derive the discrete SPH forms of the momentum equation which in continuum form is given by

$$\frac{D\mathbf{v}}{Dt} = \frac{1}{\rho}\,\nabla\cdot\boldsymbol{\sigma}. \tag{2.55}$$

When the momentum equation is directly approximated by $\dot{\mathbf{v}}_a \approx \frac{1}{\rho_a}\langle\nabla\cdot\boldsymbol{\sigma}_a\rangle$ the following discrete SPH equation is obtained

$$\dot{\mathbf{v}}_a = \frac{1}{\rho_a}\sum_b V_b\boldsymbol{\sigma}_b\nabla w_b\left(\mathbf{x}_a, h_b\right). \tag{2.56}$$

Using equation (2.43) the equation can be symmetrized by adding the zero term $\frac{\boldsymbol{\sigma}_a}{\rho_a}\sum_b V_b\nabla w_b\left(\mathbf{x}_a, h_b\right)$ to give

$$\dot{\mathbf{v}}_a = \frac{1}{\rho_a}\sum_b V_b\left(\boldsymbol{\sigma}_a + \boldsymbol{\sigma}_b\right)\nabla w_b\left(\mathbf{x}_a, h_b\right). \tag{2.57}$$

Another form for the discrete momentum equation can be derived using the vector identity

$$\dot{\mathbf{v}} = \frac{1}{\rho}\,\nabla\cdot\boldsymbol{\sigma} = \nabla\cdot\left(\frac{1}{\rho}\,\boldsymbol{\sigma}\right) + \frac{1}{\rho^2}\,\boldsymbol{\sigma}\left(\nabla\rho\right). \tag{2.58}$$

Applying the SPH approximations to the gradients terms gives

$$\dot{\mathbf{v}}_a = \left\langle\nabla\cdot\left(\frac{1}{\rho_a}\,\boldsymbol{\sigma}_a\right)\right\rangle + \frac{1}{\rho_a^2}\,\boldsymbol{\sigma}_a\left\langle\nabla\rho_a\right\rangle \tag{2.59}$$

and the final form of the momentum equation is obtained as

$$\begin{aligned}
\dot{\mathbf{v}}_a &= \sum_b V_b\frac{\boldsymbol{\sigma}_b}{\rho_b}\nabla w_b\left(\mathbf{x}_a, h_b\right) + \frac{\boldsymbol{\sigma}_a}{\rho_a^2}\sum_b V_b\rho_b\nabla w_b\left(\mathbf{x}_a, h_b\right)\\
&= \sum_b m_b\frac{\boldsymbol{\sigma}_b}{\rho_b^2}\nabla w_b\left(\mathbf{x}_a, h_b\right) + \sum_b m_b\frac{\boldsymbol{\sigma}_a}{\rho_a^2}\nabla w_b\left(\mathbf{x}_a, h_b\right)\\
\dot{\mathbf{v}}_a &= \sum_b m_b\left(\frac{\boldsymbol{\sigma}_a}{\rho_a^2} + \frac{\boldsymbol{\sigma}_b}{\rho_b^2}\right)\nabla w_b\left(\mathbf{x}_a, h_b\right).
\end{aligned} \tag{2.60}$$

2.5.3 Discrete stress tensor for Newtonian fluids

In order to apply the SPH momentum equations the particle stresses also need to be formulated in the discrete SPH framework. In this section the discrete SPH equation for the stress of a Newtonian fluid is derived.

In this case the stress tensor can be decomposed into

$$\boldsymbol{\sigma} = -P\boldsymbol{I} + \boldsymbol{\sigma}' \tag{2.61}$$

where P is the *isotropic pressure* and $\boldsymbol{\sigma}'$ is the *deviatoric stress*.

For Newtonian fluids the deviatoric stress is proportional to the *deviatoric rate of deformation tensor* $\mathbf{d}'$ via the *dynamic viscosity* μ

$$\boldsymbol{\sigma}' = 2\mu\mathbf{d}'. \tag{2.62}$$

The deviatoric rate of deformation tensor $\mathbf{d}'$ is defined by

$$\mathbf{d}' = \mathbf{d} - \frac{1}{3}\left(\nabla\cdot\mathbf{v}\right)\mathbf{I} \quad \text{where} \quad \mathbf{d} = \frac{1}{2}\left(\nabla\mathbf{v} + \nabla\mathbf{v}^T\right). \tag{2.63}$$

Applying summation approximation to each gradient term in equation (2.63) gives

$$\begin{aligned} 2\mathbf{d}'_a = \sum_b V_b\mathbf{v}_b \otimes \nabla w_b\left(\mathbf{x}_a, h_b\right) + \sum_b V_b\nabla w_b\left(\mathbf{x}_a, h_b\right)\otimes\mathbf{v}_b \\ -\frac{2}{3}\left(\sum_b V_b\mathbf{v}_b\cdot\nabla w_b\left(\mathbf{x}_a, h_b\right)\right)\mathbf{I}. \end{aligned} \tag{2.64}$$

By subtracting multiples of the zero term given in equation (2.43) from equation (2.64) we arrive at an SPH formulation for the deviatoric rate of deformation tensor written in terms of the velocity differences $\mathbf{v}_{ba} = \mathbf{v}_b - \mathbf{v}_a$ given by

$$\begin{aligned} 2\mathbf{d}'_a = \sum_b V_b\mathbf{v}_{ba} \otimes \nabla w_b\left(\mathbf{x}_a, h_b\right) + \sum_b V_b\nabla w_b\left(\mathbf{x}_a, h_b\right)\otimes\mathbf{v}_{ba} \\ -\frac{2}{3}\left(\sum_b V_b\mathbf{v}_{ba}\cdot\nabla w_b\left(\mathbf{x}_a, h_b\right)\right)\mathbf{I}. \end{aligned} \tag{2.65}$$

The evaluation of the $\frac{1}{3}\left(\nabla\cdot\mathbf{v}\right)\mathbf{I}$ term can be simplified by noting that $\nabla\cdot\mathbf{v} = \mathrm{tr}\left(\mathbf{d}\right)$ which can be calculated directly from the SPH approximation of $\mathbf{d}$.

Therefore, with these formula for the particle stresses the momentum equations for a Newtonian fluid can be written in full as

$$\begin{aligned} \dot{\mathbf{v}}_a &= \frac{1}{\rho_a}\sum_b V_b\left(\boldsymbol{\sigma}_a + \boldsymbol{\sigma}_b\right)\nabla w_b\left(\mathbf{x}_a, h_b\right) \\ \dot{\mathbf{v}}_a &= \sum_b m_b\left(\frac{\boldsymbol{\sigma}_a}{\rho_a^2} + \frac{\boldsymbol{\sigma}_b}{\rho_b^2}\right)\nabla w_b\left(\mathbf{x}_a, h_b\right) \end{aligned} \qquad \text{where } \boldsymbol{\sigma} = -P\mathbf{I} + 2\mu\mathbf{d}'. \tag{2.66}$$

2.5.4 Energy equation for Newtonian fluids

This section will derive the discrete SPH forms of the energy equation which in continuum form is given by

$$\frac{De}{Dt} = \frac{1}{\rho}\,\boldsymbol{\sigma} : \nabla \mathbf{v}. \tag{2.67}$$

In the case of Newtonian fluids equation (2.67) may be rewritten in terms of the deviatoric rate of deformation tensor by noting that $\mathbf{d}' : \nabla \mathbf{v} = \mathbf{d}' : \mathbf{d}'$ to give

$$\begin{aligned} \frac{De}{Dt} &= -\frac{P}{\rho}\nabla \cdot \mathbf{v} + \frac{2\mu}{\rho}\,\mathbf{d}' : \nabla \mathbf{v} \\ &= \frac{2\mu}{\rho}\,\mathbf{d}' : \mathbf{d}' - \frac{P}{\rho}\nabla \cdot \mathbf{v}. \end{aligned} \tag{2.68}$$

The first term of equation (2.68) can be approximated with the previously derived SPH equation for $\mathbf{d}'$.

The second term of equation (2.68) can be approximated by using either one of the discrete SPH equations for the continuity equation by noting that

$$-\frac{P}{\rho}\nabla \cdot \mathbf{v} = \frac{P}{\rho^2}\left(-\rho \nabla \cdot \mathbf{v}\right) = \frac{P}{\rho^2}\frac{D\rho}{Dt}\ . \tag{2.69}$$

Continuity method I

Using equation (2.44) for $\dot{\rho}$ gives

$$-\frac{P_a}{\rho_a}\nabla \cdot \mathbf{v}_a = \frac{P_a}{\rho_a}\sum_b V_b \left(\mathbf{v}_a - \mathbf{v}_b\right) \cdot \nabla w_b \left(\mathbf{x}_a, h_b\right). \tag{2.70}$$

While noting that

$$-\frac{P}{\rho}\nabla \cdot \mathbf{v} = \frac{1}{\rho}\left(\mathbf{v} \cdot \nabla P - \nabla \cdot (P\mathbf{v})\right) \tag{2.71}$$

gives an alternate discretization of the form

$$\begin{aligned} -\frac{P_a}{\rho_a}\nabla \cdot \mathbf{v}_a &= \frac{1}{\rho_a}\left(\mathbf{v}_a \cdot \sum_b V_b P_b \nabla w_b \left(\mathbf{x}_a, h_b\right) - \sum_b V_b \left(P_b \mathbf{v}_b\right) \cdot \nabla w_b \left(\mathbf{x}_a, h_b\right)\right) \\ &= \frac{1}{\rho_a}\sum_b V_b P_b \left(\mathbf{v}_a - \mathbf{v}_b\right) \cdot \nabla w_b \left(\mathbf{x}_a, h_b\right). \end{aligned} \tag{2.72}$$

Taking an average of equation (2.70) and equation (2.72) gives a symmetric formula for the pressure work given by

$$-\frac{P_a}{\rho_a}\nabla \cdot \mathbf{v}_a = \frac{1}{2\rho_a}\sum_b V_b \left(P_a + P_b\right) \mathbf{v}_{ab} \cdot \nabla w_b \left(\mathbf{x}_a, h_b\right) \tag{2.73}$$

where $\mathbf{v}_{ab} = \mathbf{v}_a - \mathbf{v}_b$.

Continuity method II

Using equation (2.45) for $\dot{\rho}$ gives

$$-\frac{P_a}{\rho_a}\nabla\cdot\mathbf{v}_a = \frac{P_a}{\rho_a^2}\sum_b m_b\left(\mathbf{v}_a-\mathbf{v}_b\right)\cdot\nabla w_b\left(\mathbf{x}_a,h_b\right). \tag{2.74}$$

While noting that

$$-\frac{P}{\rho}\nabla\cdot\mathbf{v} = \mathbf{v}\cdot\nabla\left(\frac{P}{\rho}\right)-\nabla\cdot\left(\frac{P}{\rho}\mathbf{v}\right) \tag{2.75}$$

gives an alternate discretization of the form

$$\begin{aligned} -\frac{P_a}{\rho_a}\nabla\cdot\mathbf{v}_a &= \mathbf{v}_a\cdot\sum_b V_b\frac{P_b}{\rho_b}\nabla w_b\left(\mathbf{x}_a,h_b\right)-\sum_b V_b\left(\frac{P_b}{\rho_b}\mathbf{v}_b\right)\cdot\nabla w_b\left(\mathbf{x}_a,h_b\right) \\ &= \sum_b m_b\frac{P_b}{\rho_b^2}\left(\mathbf{v}_a-\mathbf{v}_b\right)\cdot\nabla w_b\left(\mathbf{x}_a,h_b\right). \end{aligned} \tag{2.76}$$

Taking an average of equation (2.74) and equation (2.76) gives a symmetric formula for the pressure work given by

$$-\frac{P_a}{\rho_a}\nabla\cdot\mathbf{v}_a = \frac{1}{2}\sum_b m_b\left(\frac{P_a}{\rho^2}+\frac{P_b}{\rho^2}\right)\mathbf{v}_{ab}\cdot\nabla w_b\left(\mathbf{x}_a,h_b\right) \tag{2.77}$$

where $\mathbf{v}_{ab}=\mathbf{v}_a-\mathbf{v}_b$.

In this way two symmetric forms for the evolution of the total internal energy are given by

$$\frac{De_a}{Dt} = \frac{2\mu_a}{\rho_a}\,\mathbf{d}_a'\!:\mathbf{d}_a' \;+\; \frac{1}{2\rho_a}\sum_b V_b\left(P_a+P_b\right)\mathbf{v}_{ab}\cdot\nabla w_b\left(\mathbf{x}_a,h_b\right) \tag{2.78a}$$

$$\frac{De_a}{Dt} = \frac{2\mu_a}{\rho_a}\,\mathbf{d}_a'\!:\mathbf{d}_a' \;+\; \frac{1}{2}\sum_b m_b\left(\frac{P_a}{\rho^2}+\frac{P_b}{\rho^2}\right)\mathbf{v}_{ab}\cdot\nabla w_b\left(\mathbf{x}_a,h_b\right) \tag{2.78b}$$

2.5.5 Equation of state

The final element is an equation of state which relates the pressure to the density of the fluid and closes the above system of equations.

In order to apply explicit time integration schemes in SPH it is necessary to assume that the fluid is slightly compressible. This is a reasonable assumption since all real fluids are compressible to some degree with a sound speed much faster than the bulk flow of the fluid. This is measured by the *Mach number* which is defined as

$$M = \frac{\|\mathbf{v}_{\text{bulk}}\|}{c} \tag{2.79}$$

where $\|\mathbf{v}_{\text{bulk}}\|$ is the bulk flow speed and c is the speed of sound. Values of $M \leq 0.1$ imply a mostly incompressible flow behaviour.

It has been argued by Monaghan [91] that in order for relative density fluctuations to be less than 1% the Mach number should be between 0.1 & 0.001 which for a fluid flow with maximum velocity $\mathbf{v}_{\max}$ results in a sound speed ranging between

$$c = M^{-1} \|\mathbf{v}_{\max}\| = 10 \|\mathbf{v}_{\max}\| \longleftrightarrow 1000 \|\mathbf{v}_{\max}\| . \tag{2.80}$$

The equation of state used in this thesis is the same as given by Batchelor [7], but modified for fluid flow simulations in SPH

$$P_a = P_0 \left(\left(\frac{\rho_a}{\rho_0} \right)^{\gamma} - 1 \right) \tag{2.81}$$

where P_a is the pressure of particle a, ρ_0 is the material density and ρ_a is the density of particle a. γ is the fictitious ratio of principle specific heats (taken as 7 for water), and P_0 is an artificial isothermal bulk modulus.

The speed of sound for a fluid with an equation of state given by equation (2.81) is given by

$$c = \sqrt{\frac{\gamma P_0}{\rho}} . \tag{2.82}$$

Combining equations (2.80) & (2.82) gives a simple equation for P_0 which will ensure only small variations in density

$$P_0 = \frac{\left(M^{-1} \|\mathbf{v}_{\max}\|\right)^2 \rho}{\gamma} . \tag{2.83}$$

2.6 Timestepping schemes

Timestepping schemes both explicit and implicit can be implemented into SPH codes. The superscripts denote the timestep at which the variable in question is evaluated and time is updated by the relation $t^{n+1} = t^n + \Delta t^{n+1}$.

Continuity Equation

If the continuity approach is used to evaluate the material density the time derivative can be calculated by the forwards difference approximation to $\dot{\rho}_a$

$$\dot{\rho}_a^n \approx \frac{\rho_a^{n+1} - \rho_a^n}{\Delta t^{n+1}} \,. \tag{2.84}$$

In conjunction with equation (2.45) this gives

$$\rho_a^{n+1} = \rho_a^n + \Delta t^{n+1} \left(\sum_b m_b \left(\mathbf{v}_a^n - \mathbf{v}_b^n \right) \cdot \nabla w_b \left(\mathbf{x}_a^n, h_b \right) \right). \tag{2.85}$$

Alternatively, noting that the solution to the linear ordinary differential equation $\dot{x}(t) = Ax(t)$ with initial condition $x(t_0) = x_0$ has the solution $x(t) = x_0 e^{A(t-t_0)}$. In conjunction with equation (2.44) the density can be updated by

$$\rho_a^{n+1} = \rho_a^n e^{\Delta t^{n+1} \sum\limits_b V_b \left(\mathbf{v}_a^n - \mathbf{v}_b^n \right) \cdot \nabla w_b \left(\mathbf{x}_a^n, h_b \right)}. \tag{2.86}$$

Momentum Equation

In this thesis a simple leap-frog scheme has been implemented to update the particles position and velocity as shown in Figure 2.7.

The intermediate velocity of particle a of the current timestep denoted by $\mathbf{v}_a^{n+\frac{1}{2}}$ is estimated by

$$\mathbf{v}_a^{n+\frac{1}{2}} = \mathbf{v}_a^{n-\frac{1}{2}} + \overline{\Delta t} \mathbf{a}_a^n \tag{2.87}$$

where $\overline{\Delta t} = \frac{1}{2}\left(\Delta t^n + \Delta t^{n+1}\right)$ is the average of the current and previous timesteps and $\mathbf{a}_a^n$ is the current acceleration. The scheme is initialised by first calculating $\mathbf{v}_a^{\frac{1}{2}} = \mathbf{v}_a^0 + \frac{1}{2}\Delta t^1 \mathbf{a}_a^0$. This estimate for the velocity at the midpoint of the timestep is then used to update the position of particle a by

$$\mathbf{x}_a^{n+1} = \mathbf{x}_a^n + \Delta t^{n+1} \mathbf{v}_a^{n+\frac{1}{2}}. \tag{2.88}$$

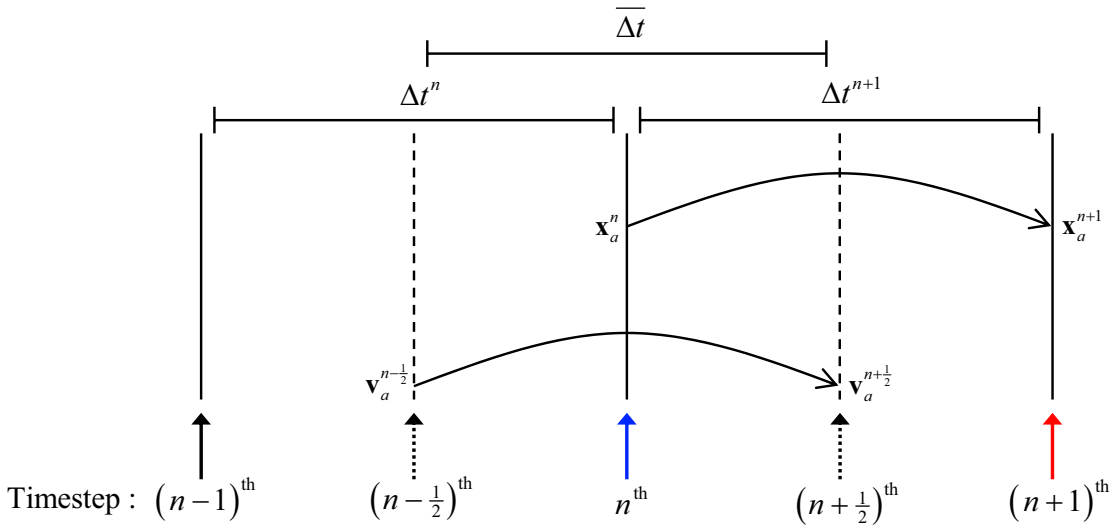


Figure 2.7: Leapfrog time integration scheme

The leap-frog scheme is $\mathcal{O}(\Delta t^3)$ and computationally very efficient since it avoids the need to calculate particle accelerations $\mathbf{a}_a^{n+\frac{1}{2}}$ at the midpoint of each timestep.

Timestep Calculation

The length of the timesteps are calculated in accordance to the *Courant-Fredreichs-Lewy stability condition* [35]. This states that the computational domain of dependence of the numerical scheme should include the physical domain of dependence. In other words the maximum speed of numerical propagation must exceed the maximum speed of physical propagation.

In SPH applications this results in the timestep being proportional to the smallest particle smoothing length h_{min} and the timestep is calculated by

$$\Delta t = \text{CFL}\frac{h_{\text{min}}}{\max\left(c_a + \|\mathbf{v}_a\|\right)} \tag{2.89}$$

where $\text{CFL} \in (0, 1]$ is a constant, typically $\text{CFL} \approx 0.1 - 0.2$. c_a is the sound speed of particle a given by equation (2.82).

It should be noted that increasing the speed of sound by reducing the Mach number M will significantly restrict the maximum stable timestep.

2.7 Nearest neighbour search algorithms

For grid based numerical methods the relative position and connectivity of the nodes remains fixed throughout a simulation. In SPH this is not the case. Neighbouring particles within the support of a kernel function will generally change as the problem evolves. These are known as nearest neighbour particles and need to be calculated for each particle at every timestep.

All particle interactions can be found by simply looping over all other particles and checking if the distance between them is smaller than the smoothing length of the particle. However, the complexity of this approach is $\mathcal{O}(n^2)$ which for large simulations consisting of thousands of particles is prohibitively slow.

In this section two efficient methods commonly used to calculate nearest neighbour particles are described.

Grid based searching

The complexity of the nearest neighbour particle search can be reduced to $\mathcal{O}(n)$ with the introduction of an underlying search grid and associated linked–list data structure.

An underlying grid covers the computational domain and is sufficiently large to contain all simulation particles. The cells of the grid are at least $2h$ in diameter and are identified by a unique cell number. In this way each particle need only check its neighbouring cells in order to identify all nearest neighbour particles as shown in Figure 2.8. The cell number for each particle is assigned and most efficiently stored in a linked–list structure with particles contained in the same cell chained together to minimise storage.

The efficiency can be increased further if a constant smoothing length is used since all particle interactions will be symmetric so particle pairs need only be checked once and the relevant contributions calculated and added to both particles. Under this assumption only five out of the nine cells need to be checked in two dimensions and 14 out of the 27 cells in three dimensions (see Figure 2.8).

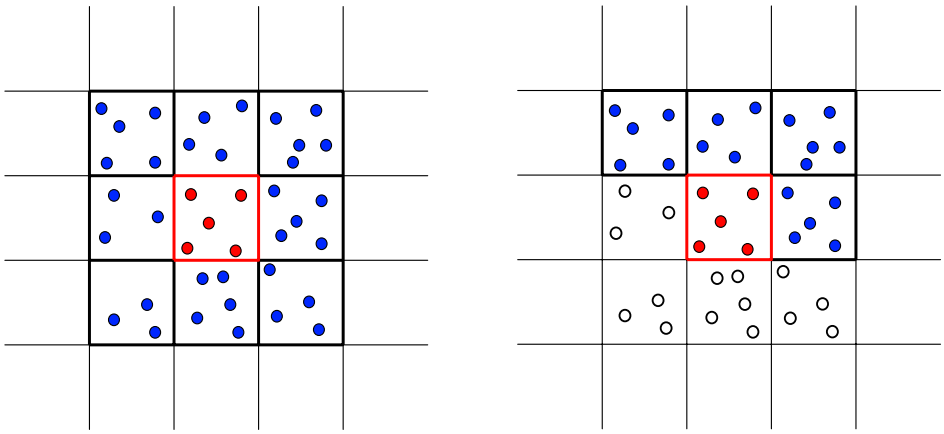


Figure 2.8: Search grids in 2D. Full grid (left), reduced symmetric grid (right)

Consequently, the grid method is particularly effective for simulations using a constant smoothing length. However, its efficiency will be reduced for simulations with variable smoothing lengths since the required cell width will not be optimal for all particles.

Tree based searching

The second method for nearest neighbour particle searching is to use ordered tree based data structures to store particle positions and to efficiently search the computational domain for neighbouring particles [20, 54, 76]. These approaches are better suited for large problems with variable smoothing lengths and can reduce the complexity of the nearest neighbour particle search to $\mathcal{O}(n \log(n))$.

Tree methods recursively bisect the problem domain into smaller subregions with particles inserted into the tree according to the subregion in which they reside. In this thesis the alternating digital tree (ADT) method [20] is implemented using a binary tree. Each node of the binary tree contains a single particle and has an associated left and right link. These links can either be empty or link to a node on the next hierarchy level of the tree structure. In this way each node of the tree can link to at most two other nodes. Consequently, this bisection process is easily represented by a binary tree as shown in Figure 2.9.

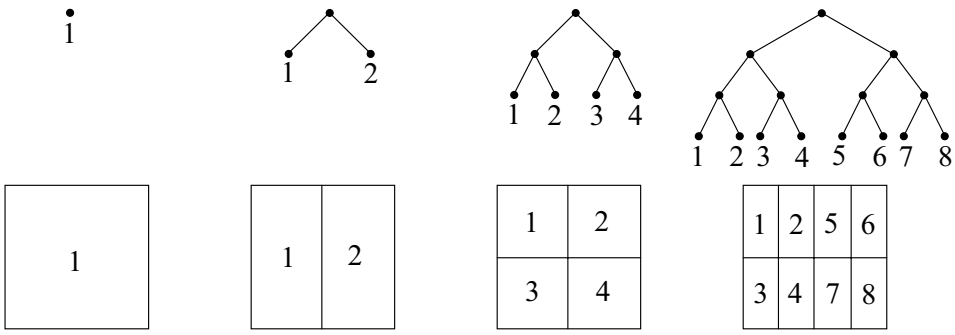


Figure 2.9: The binary tree bisection process in 2D

The root of the binary tree represents the entire computational domain. In two dimensions, bisecting the domain in the vertical direction results in two subregions; these left and right regions are assigned to the left and right links respectively. Each of these subregions can then be bisected in the horizontal direction and the resulting subregions assigned to the corresponding links at the next hierarchy level. In general N dimensional space the process continues indefinitely by choosing the bisection directions $x_1, x_2, \cdots, x_N$ in cyclic order.

Each particle is systematically added to the tree and lies inside the region corresponding to the node where it is stored. In this way the tree is built by first placing a particle at the root of the tree. Then subsequent particles are added by following the tree downwards until an empty node is reached; taking left or right branches according to whether the particle lies in the corresponding left or right subregion.

It is this geometric structure of the binary tree that reduces the cost of searching a region of the computational domain for particles. If the associated region of a given node k fails to intersect the search region then the complete set of particles contained within the entire subtree with root at node k can be disregarded from the search.

In this way the tree structure can be systematically searched as follows. First the coordinates of the particle in the current node are checked to see if they are inside the search region. Then, if the left link is not empty and its corresponding region intersects the search region the left subtree is checked. Similarly, if the right link is not empty and its corresponding region intersects the search region then the right subtree is checked.

2.8 Concluding remarks

This chapter has introduced all the theory required to produce a simple, fully functional SPH code for the solution of incompressible free surface fluid flows. The various stages of a typical SPH implementation are shown in Figure 2.10 with references to the relevant equations given in this chapter.

The only notable omission in the discussion has been the various methods which can be used to implement boundary conditions in SPH simulations. This has been deferred to Chapter 5 where a number of different approaches will be presented in detail.

In the next chapter several techniques will be introduced which improves the accuracy and stability of the basic SPH equations while maintaining their simple form. With these corrections it will be shown that the discrete SPH equations conserve both the linear and angular momentum of the system.

- Initialise particle properties.

Begin timestep :

- Calculate timestep Δt (eq. 2.89)
- Identify particle neighbours (sec. 2.7)
- Update densities ρ_a (eq. 2.85 or eq. 2.86)
- Update volumes V_a (eq. 2.18)
- Update pressures P_a (eq. 2.81)
- Calculate deviatoric stress tensors $\boldsymbol{\sigma}'_a = 2\mu\mathbf{d}'_a$ (eq. 2.65)
- Calculate body forces (eg. gravity $\mathbf{g}$)
- Calculate boundary forces (ch. 5)
- Calculate particle accelerations $\mathbf{a}_a$ (eq. 2.66)
- Update particle velocities and positions $\mathbf{v}_a$, $\mathbf{x}_a$ (eq. 2.87 & eq. 2.88)
- Update current time $t = t + \Delta t$
- Output particle data

End timestep

- if $(t < t_{\text{stop}})$ then continue next timestep
- if $(t \geq t_{\text{stop}})$ then STOP

Figure 2.10: Numerical algorithm of a standard SPH code

Chapter 3

Corrected SPH and stabilization

3.1 Introduction

In this chapter several methods which improve the accuracy and stability of the traditional SPH equations are introduced.

Correction terms are added to the kernel function in order to enforce consistency of the discrete summation approximations. First order correction terms ensure that any linear or constant function will be exactly reproduced by the summation approximation.

Unfortunately, the expression for the gradient of the corrected kernel is complicated and computationally expensive to evaluate. Instead, the gradient of the kernel is directly corrected to ensure first order consistency of the gradient of a function.

In Chapter 7 it will be necessary to introduce additional higher order terms into the expression for the corrected gradient of a function. This is achieved with the introduction of an extra Hessian term in the gradient of the kernel function and results in a fully corrected and stabilized SPH formulation.

Finally, it is proved that with first order kernel corrections the SPH method conserves both linear and angular momentum for simulations with variable smoothing lengths.

3.2 Kernel correction

Recall, the reproducing kernel approximation $\langle\cdot\rangle$ of a function is said to be order k accurate if any polynomial up to the k^{th} order is exactly reproduced.

In one dimension this implies that if

$$g(x) = g_0 + g_1 x + g_2 x^2 + \cdots + g_k x^k \tag{3.1}$$

is an arbitrary k^{th} order polynomial with constant coefficients then

$$\langle g(x)\rangle = g(x). \tag{3.2}$$

Replacing the reproducing kernel approximation with the discrete SPH approximation gives

$$g_h(x) = \sum_b V_b g(x_b) w_b(x, h_b) = g_0 + g_1 x + g_2 x^2 + \cdots + g_k x^k. \tag{3.3}$$

The *discrete consistency conditions* are obtained by setting $g_j = 1$ and all other constants equal to zero, for each $0 < j \leqslant k$.

In particular, the constant consistency condition, frequently referred to as a *partition of unity* is given by

$$1 = \sum_b V_b w_b(x, h_b), \tag{3.4}$$

while the k^{th} order consistency condition is given by

$$x^k = \sum_b V_b x_b^k w_b(x, h_b). \tag{3.5}$$

Any consistency the kernel function may have possessed is not enough to ensure that the discrete consistency conditions will be satisfied. This loss of consistency is a consequence of the approximate pointwise integration. However, with the addition of simple kernel correction terms this discrepancy can be eliminated.

The simplest means to guarantee a certain degree of consistency in SPH is to introduce a polynomial correction term into the definition of the kernel function as proposed by Liu [77, 81]

$$\hat{w}_b(\mathbf{x}, h_b) = [\alpha(\mathbf{x}) + \boldsymbol{\beta}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{x}_b) + (\mathbf{x} - \mathbf{x}_b) \cdot \boldsymbol{\Gamma}(\mathbf{x}) (\mathbf{x} - \mathbf{x}_b) + \cdots] w_b(\mathbf{x}, h_b)$$

where $\alpha(\mathbf{x})$, $\boldsymbol{\beta}(\mathbf{x})$, $\boldsymbol{\Gamma}(\mathbf{x})$ are scalar, vector, and second order tensor correction terms respectively.

In theory, consistency of any order can be enforced using the above correction method. However, only first order correction is considered in this thesis and the corrected kernel function is given by

$$\hat{w}_b(\mathbf{x}, h_b) = \alpha(\mathbf{x}) \left[1 + \boldsymbol{\beta}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{x}_b)\right] w_b(\mathbf{x}, h_b). \tag{3.6}$$

Introducing the corrected kernel into equation (3.3) for the SPH approximation gives

$$g_h(\mathbf{x}) = \sum_b V_b g(\mathbf{x}_b) \hat{w}_b(\mathbf{x}, h_b). \tag{3.7}$$

where $\alpha(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$ are the kernel correction terms. Expressions for $\alpha(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$ will be derived in the following sections.

3.2.1 Linear kernel correction

In this case an arbitrary linear function $g(\mathbf{x}) = A_0 + \mathbf{A}_1 \cdot \mathbf{x}$ should be exactly interpolated

$$A_0 + \mathbf{A}_1 \cdot \mathbf{x} = \sum_b V_b (A_0 + \mathbf{A}_1 \cdot \mathbf{x}_b) \hat{w}_b(\mathbf{x}, h_b). \tag{3.8}$$

Setting $\mathbf{A}_1 = 0$ in the above is equivalent to the zero order consistency condition given by equation (3.4) for the corrected kernel at the point $\mathbf{x}$

$$1 = \sum_b V_b \hat{w}_b(\mathbf{x}, h_b). \tag{3.9}$$

Substituting the corrected kernel into the above gives

$$1 = \alpha(\mathbf{x}) \sum_b V_b \left[1 + \boldsymbol{\beta}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{x}_b)\right] w_b(\mathbf{x}, h_b) \tag{3.10}$$

and the constant correction term $\alpha(\mathbf{x})$ is obtained as

$$\alpha(\mathbf{x}) = \frac{1}{\sum_b V_b \left[1 + \boldsymbol{\beta}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{x}_b)\right] w_b(\mathbf{x}, h_b)}. \tag{3.11}$$

Similarly, if $A_0 = 0$ then equation (3.8) becomes

$$\mathbf{A}_1 \cdot \mathbf{x} = \mathbf{A}_1 \cdot \sum_b V_b \mathbf{x}_b \hat{w}_b(\mathbf{x}, h_b) \tag{3.12}$$

and $\mathbf{x}$ can be written as

$$\mathbf{x} = \sum_b V_b \mathbf{x}_b \hat{w}_b(\mathbf{x}, h_b). \tag{3.13}$$

Using the zero order consistency condition given by equation (3.9) gives

$$\mathbf{x}\sum_b V_b\hat{w}_b\left(\mathbf{x},h_b\right)=\sum_b V_b\mathbf{x}_b\hat{w}_b\left(\mathbf{x},h_b\right) \tag{3.14}$$

and the first order consistency condition for the corrected kernel at point $\mathbf{x}$ is obtained as

$$\sum_b V_b\left(\mathbf{x}-\mathbf{x}_b\right)\hat{w}_b\left(\mathbf{x},h_b\right)=\mathbf{0}. \tag{3.15}$$

Substituting the corrected kernel into the above and noting that $\alpha\left(\mathbf{x}\right)$ is independent of particle positions $\mathbf{x}_b$ gives

$$\sum_b V_b\left(\mathbf{x}-\mathbf{x}_b\right)\left[1+\boldsymbol{\beta}\left(\mathbf{x}\right)\cdot\left(\mathbf{x}-\mathbf{x}_b\right)\right]w_b\left(\mathbf{x},h_b\right)=\mathbf{0}. \tag{3.16}$$

Finally, by rearranging the above equation the vector correction term $\boldsymbol{\beta}\left(\mathbf{x}\right)$ is found to be

$$\boldsymbol{\beta}\left(\mathbf{x}\right)=\left[\sum_b V_b w_b\left(\mathbf{x},h_b\right)\left[\left(\mathbf{x}-\mathbf{x}_b\right)\otimes\left(\mathbf{x}-\mathbf{x}_b\right)\right]\right]^{-1}\sum_b V_b\left(\mathbf{x}_b-\mathbf{x}\right)w_b\left(\mathbf{x},h_b\right). \tag{3.17}$$

With these expressions for $\alpha\left(\mathbf{x}\right)$ and $\boldsymbol{\beta}\left(\mathbf{x}\right)$ the Corrected SPH approximation ensures that linear functions are exactly interpolated and that their gradients exactly obtained when differentiated with respect to $\mathbf{x}$.

It has been proved by Kulasegaram [67] that these correction terms are in fact equivalent to the corrections introduced in the RKPM and MLSRKPM methods.

Unfortunately, calculating the gradient of the linearly corrected kernel is complicated and computationally expensive for large simulations since both $\alpha\left(\mathbf{x}\right)$ and $\boldsymbol{\beta}\left(\mathbf{x}\right)$ are functions of $\mathbf{x}$ and the resulting expressions would have to be computed for each particle at every timestep.

3.2.2 Constant kernel correction

In an effort to increase computational speed a compromise is to use only constant kernel correction. This is equivalent to setting $\boldsymbol{\beta}(\mathbf{x}) = \mathbf{0}$ and results in simplified equations for the corrected kernel and its gradient. In this case the kernel is given by

$$\hat{w}_b(\mathbf{x}, h_b) = \alpha(\mathbf{x})\, w_b(\mathbf{x}, h_b) \quad \text{where} \quad \alpha(\mathbf{x}) = \frac{1}{\sum_b V_b w_b(\mathbf{x}, h_b)}\,. \tag{3.18}$$

This leads to a simplified expression for the gradient of a function.

$$\nabla g_h(\mathbf{x}) = \sum_b V_b g(\mathbf{x}_b)\, \nabla \hat{w}_b(\mathbf{x}, h_b) \tag{3.19}$$

where $\nabla \hat{w}_b(\mathbf{x}, h_b)$ is the gradient of the constant corrected kernel function given by

$$\nabla \hat{w}_b(\mathbf{x}, h_b) = \alpha(\mathbf{x})\, \nabla w_b(\mathbf{x}, h_b) + w_b(\mathbf{x}, h_b)\, \nabla \alpha(\mathbf{x})\,. \tag{3.20}$$

In order to evaluate $\nabla\alpha(\mathbf{x})$ take the gradient of the zero order consistency condition given by equation (3.9) to give

$$\mathbf{0} = \sum_b V_b \left(\alpha(\mathbf{x})\, \nabla w_b(\mathbf{x}, h_b) + w_b(\mathbf{x}, h_b)\, \nabla \alpha(\mathbf{x})\right). \tag{3.21}$$

Rearranging the above gives $\nabla\alpha$ as a function of $\mathbf{x}$

$$\nabla \alpha(\mathbf{x}) = \frac{-\alpha(\mathbf{x}) \sum_b V_b \nabla w_b(\mathbf{x}, h_b)}{\sum_b V_b w_b(\mathbf{x}, h_b)}. \tag{3.22}$$

After simple algebra the gradient of the constant corrected kernel is obtained as

$$\nabla \hat{w}_b(\mathbf{x}, h_b) = \frac{\nabla w_b(\mathbf{x}, h_b) - w_b(\mathbf{x}, h_b)\, \boldsymbol{\gamma}(\mathbf{x})}{\sum_b V_b w_b(\mathbf{x}, h_b)} \tag{3.23}$$

where $\boldsymbol{\gamma}(\mathbf{x})$ is given by

$$\boldsymbol{\gamma}(\mathbf{x}) = \frac{\sum_b V_b \nabla w_b(\mathbf{x}, h_b)}{\sum_b V_b w_b(\mathbf{x}, h_b)}\,. \tag{3.24}$$

This correction is easier to implement than the full linear correction. However, in general the gradient will fail to satisfy linear consistency exactly. Nevertheless this procedure markedly improves the interpolation of SPH computations.

3.3 Kernel gradient correction

Alternatively, another simple way to ensure linear consistency of the gradient is to directly correct the gradient expression itself,

$$\nabla g_h(\mathbf{x}) = \sum_b V_b g(\mathbf{x}_b)\, \tilde{\nabla} w_b(\mathbf{x}, h_b) \tag{3.25}$$

where $\tilde{\nabla} w_b(\mathbf{x}, h_b)$ is the corrected gradient of the kernel function.

3.3.1 Constant gradient correction

In this case the SPH approximation should exactly interpolate the gradient of a constant function. Substituting $g(\mathbf{x}) = C$ into equation (3.25) gives the following constant consistency condition for $\tilde{\nabla} w_b(\mathbf{x}, h_b)$,

$$\mathbf{0} = \sum_b V_b \tilde{\nabla} w_b(\mathbf{x}_a, h_b) \qquad \text{for each} \quad \mathbf{x}_a. \tag{3.26}$$

This can be satisfied by defining $\tilde{\nabla} w_b(\mathbf{x}_a, h_b)$ as

$$\tilde{\nabla} w_b(\mathbf{x}_a, h_b) = \nabla w_b(\mathbf{x}_a, h_b) + \boldsymbol{\xi}(\mathbf{x}_a)\, \delta_{ab} \ . \tag{3.27}$$

and substituting $\tilde{\nabla} w_b(\mathbf{x}_a, h_b)$ into equation (3.26) to give

$$\mathbf{0} = \sum_b V_b \nabla w_b(\mathbf{x}_a, h_b) + V_a \boldsymbol{\xi}(\mathbf{x}_a) \, . \tag{3.28}$$

The correction term $\boldsymbol{\xi}(\mathbf{x}_a)$ is then obtained as

$$\boldsymbol{\xi}(\mathbf{x}_a) = \frac{-\sum_b V_b \nabla w_b(\mathbf{x}_a, h_b)}{V_a} \ . \tag{3.29}$$

Substituting this expression for $\boldsymbol{\xi}(\mathbf{x}_a)$ back into equation (3.25) gives

$$\nabla g_h(\mathbf{x}_a) = \sum_b V_b \left[g(\mathbf{x}_b) - g(\mathbf{x}_a) \right] \nabla w_b(\mathbf{x}_a, h_b) \tag{3.30}$$

and results in a formulation that ensures that the gradient of a constant function is correctly evaluated.

3.3.2 Linear gradient correction

A similar consistency condition can be obtained to ensure linear completeness of the gradient. In this case the gradient of a linear function $g(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x}$ should be correctly obtained as $\nabla g(\mathbf{x}) = \mathbf{A}$ and

$$\begin{aligned} \mathbf{A} &= \sum_b V_b (\mathbf{A} \cdot \mathbf{x}_b) \tilde{\nabla} w_b (\mathbf{x}_a, h_b) \\ &= \left[\sum_b V_b \tilde{\nabla} w_b (\mathbf{x}_a, h_b) \otimes \mathbf{x}_b \right] \mathbf{A}. \end{aligned} \tag{3.31}$$

Therefore, $\tilde{\nabla} w_b (\mathbf{x}_a, h_b)$ needs to satisfy the following linear consistency condition

$$\mathbf{I} = \sum_b V_b \tilde{\nabla} w_b (\mathbf{x}_a, h_b) \otimes \mathbf{x}_b \quad \text{for each } \mathbf{x}_a \ . \tag{3.32}$$

This would be automatically satisfied if the linearly corrected kernel function $\hat{w}_b (\mathbf{x}, h_b)$ had been used in the calculation of the gradient. However, it is possible to ensure linear completeness of the gradient without having to differentiate the linearly corrected kernel.

By introducing a correction matrix $\mathbf{L}_a$ into the equation for the constant corrected gradient (3.30) gives

$$\nabla g_h (\mathbf{x}_a) = \sum_b V_b \left[g(\mathbf{x}_b) - g(\mathbf{x}_a) \right] \mathbf{L}_a \nabla w_b (\mathbf{x}_a, h_b). \tag{3.33}$$

To obtain an expression for the correction matrix $\mathbf{L}_a$ substitute $g(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x}$ into the above equation and set $\nabla g_h (\mathbf{x}_a) = \mathbf{A}$ to obtain

$$\mathbf{I} = \sum_b V_b \mathbf{L}_a \nabla w_b (\mathbf{x}_a, h_b) \otimes (\mathbf{x}_b - \mathbf{x}_a) \tag{3.34}$$

from which the correction matrix $\mathbf{L}_a$ is given by

$$\mathbf{L}_a = \left[\sum_b V_b \nabla w_b (\mathbf{x}_a, h_b) \otimes (\mathbf{x}_b - \mathbf{x}_a) \right]^{-1}. \tag{3.35}$$

This method results in linear consistency of the gradient and is much simpler to implement than directly calculating the gradient of a function using the linearly corrected kernel function.

3.4 Mixed kernel and gradient correction

A simple and effective correction technique is to combine constant kernel correction with linear gradient correction. Although this method will not be as accurate as full linear correction it is computationally cheap and the simplest way to achieve first order consistency of the derivative.

In this case the interpolation of the function $g(\mathbf{x})$ is given by

$$g_h(\mathbf{x}) = \sum_b V_b g(\mathbf{x}_b)\, \hat{w}_b(\mathbf{x}, h_b) \quad \text{where} \quad \hat{w}_b(\mathbf{x}, h_b) = \frac{w_b(\mathbf{x}, h_b)}{\sum_b V_b w_b(\mathbf{x}, h_b)} \,. \tag{3.36}$$

As in the previous section a correction matrix $\mathbf{L}_a$ is introduced and the gradient is given by

$$\nabla g_h(\mathbf{x}_a) = \sum_b V_b g(\mathbf{x}_b)\, \tilde{\nabla} \hat{w}_b(\mathbf{x}_a, h_b) = \sum_b V_b g(\mathbf{x}_b)\, \mathbf{L}_a \nabla \hat{w}_b(\mathbf{x}_a, h_b) \tag{3.37}$$

where now $\tilde{\nabla} \hat{w}_b(\mathbf{x}_a, h_b)$ is the corrected gradient of the constant corrected kernel function, and $\nabla \hat{w}_b(\mathbf{x}_a, h_b)$ is the gradient of constant corrected kernel function given by equations (3.23) and (3.24).

As before the matrix $\mathbf{L}_a$ is determined by enforcing the linear gradient consistency condition by setting $g(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x}$ so that

$$\begin{aligned} \mathbf{A} &= \sum_b V_b (\mathbf{A} \cdot \mathbf{x}_b)\, \mathbf{L}_a \nabla \hat{w}_b(\mathbf{x}_a, h_b) \\ &= \mathbf{L}_a \left[\sum_b V_b \nabla \hat{w}_b(\mathbf{x}_a, h_b) \otimes \mathbf{x}_b \right] \mathbf{A} \,. \end{aligned} \tag{3.38}$$

Finally, the expression for $\mathbf{L}_a$ is given as

$$\mathbf{L}_a = \left[\sum_b V_b \nabla \hat{w}_b(\mathbf{x}_a, h_b) \otimes \mathbf{x}_b \right]^{-1} \,. \tag{3.39}$$

This correction scheme will be used throughout the remainder of the thesis. It has been found that it offers the best compromise between simplicity of implementation and improved accuracy for simulations.

3.5 Hessian stabilization

In Chapter 7 it will be necessary to introduce higher order terms into the equations for the corrected gradient of a function $g(\mathbf{x})$ with the introduction of an extra Hessian term. This corrected and stabilized gradient is given by

$$\tilde{\tilde{\nabla}} g_h(\mathbf{x}_a) = \sum_b V_b g(\mathbf{x}_b) \tilde{\nabla} \hat{w}_b(\mathbf{x}_a, h_b) + \frac{1}{2}\eta \boldsymbol{\mathcal{H}} g(\mathbf{x}_a)\, \mathbf{d} \tag{3.40}$$

where $\boldsymbol{\mathcal{H}} g(\mathbf{x}) := \nabla(\nabla g(\mathbf{x}))$ is the *Hessian* of $g(\mathbf{x})$, η is an arbitrary positive constant and $\mathbf{d} = [d_x, d_y]$ is a given direction. Commonly, $d_x = d_y = h$ where h is the smoothing length.

Written in terms of the corrected kernel function $\boldsymbol{\mathcal{H}} g(\mathbf{x})$ is then given by

$$\boldsymbol{\mathcal{H}} g(\mathbf{x}) = \nabla(\nabla g(\mathbf{x})) = \sum_b V_b g(\mathbf{x}_b)\, \boldsymbol{\mathcal{H}} \hat{w}_b(\mathbf{x}, h_b) \tag{3.41}$$

where $\boldsymbol{\mathcal{H}} \hat{w}_b(\mathbf{x}, h_b) = \nabla(\nabla \hat{w}_b(\mathbf{x}, h_b))$ is the Hessian of the corrected kernel function.

Using the linearly corrected kernel function ensures that the Hessian of constant and linear functions are correctly evaluated. However, calculating $\nabla(\nabla \hat{w}_b(\mathbf{x}, h_b))$ directly is computationally expensive and so instead the Hessian of the uncorrected kernel $\nabla(\nabla w_b(\mathbf{x}, h_b))$ is calculated and then corrected as in the previous sections.

Kernel Hessian evaluation

The Hessian of the uncorrected kernel function $\nabla(\nabla w_b(\mathbf{x}_a, h_b))$ is evaluated using the following vector identity

$$\nabla(\alpha(\mathbf{x})\, \boldsymbol{\beta}(\mathbf{x})) = \alpha(\mathbf{x})\, \nabla \boldsymbol{\beta}(\mathbf{x}) + \boldsymbol{\beta}(\mathbf{x})\, (\nabla \alpha(\mathbf{x}))^{\mathrm{T}} \tag{3.42}$$

where $\alpha(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$ are scalar and vector functions of $\mathbf{x}$ respectively. Recalling the equation for $\nabla w_b(\mathbf{x}_a, h_b)$ is given by

$$\nabla w_b(\mathbf{x}_a, h_b) = \frac{1}{r}\frac{dw_b}{dr}(\mathbf{x} - \mathbf{x}_b) \tag{3.43}$$

the kernel Hessian is obtained as

$$\nabla(\nabla w_b(\mathbf{x}_a, h_b)) = \nabla\left(\frac{1}{r}\frac{dw_b}{dr}(\mathbf{x} - \mathbf{x}_b)\right). \tag{3.44}$$

By setting $\alpha(\mathbf{x}) = \frac{1}{r}\frac{dw_b}{dr}$ and $\boldsymbol{\beta}(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_b)$ in equation (3.42) and noting that

$$\nabla\alpha(r(\mathbf{x})) = \frac{d\alpha}{dr}\nabla r \quad \text{and} \quad \nabla\boldsymbol{\beta}(\mathbf{x}) = \mathbf{I}_{3\times 3}, \tag{3.45}$$

the final expression for the kernel Hessian is found to be

$$\begin{aligned}\nabla(\nabla w_b(\mathbf{x}_a, h_b)) &= \alpha(\mathbf{x})\mathbf{I}_{3\times 3} + \frac{1}{r^2}\left(\frac{d^2 w_b}{dr^2} - \alpha(\mathbf{x})\right)(\mathbf{x} - \mathbf{x}_b)(\mathbf{x} - \mathbf{x}_b)^{\mathrm{T}} \\ &= \frac{1}{r}\frac{dw_b}{dr}\mathbf{I}_{3\times 3} + \frac{1}{r^2}\left(\frac{d^2 w_b}{dr^2} - \frac{1}{r}\frac{dw_b}{dr}\right)(\mathbf{x} - \mathbf{x}_b)(\mathbf{x} - \mathbf{x}_b)^{\mathrm{T}}. \end{aligned} \tag{3.46}$$

3.5.1 Corrected kernel Hessian

The corrected Hessian denoted by $\tilde{\boldsymbol{\mathcal{H}}} w_b(\mathbf{x}_a, h_b)$ is given by

$$\tilde{\boldsymbol{\mathcal{H}}} w_b(\mathbf{x}_a, h_b) = \boldsymbol{\mathcal{H}} w_b(\mathbf{x}_a, h_b) + \delta_{ab}\mathbf{B}_a + \boldsymbol{\mathcal{A}}_a \nabla w_b(\mathbf{x}_a, h_b) \tag{3.47}$$

where $\mathbf{B}_a$ and $\boldsymbol{\mathcal{A}}_a$ are second and third order tensor correction terms respectively. These correction terms are determined by enforcing that the Hessian of a constant or linear function should vanish.

In this case, suppose $g(\mathbf{x}) = C$ where C is a constant then

$$\mathbf{0} = \boldsymbol{\mathcal{H}} g(\mathbf{x}) = \sum_b V_b C \boldsymbol{\mathcal{H}} w_b(\mathbf{x}, h_b)$$

and the constant Hessian consistency condition is obtained as

$$\mathbf{0} = \sum_b V_b \boldsymbol{\mathcal{H}} w_b(\mathbf{x}, h_b). \tag{3.48}$$

Now suppose that $g(\mathbf{x}) = \mathbf{A}\cdot\mathbf{x}$ where $\mathbf{A} \neq \mathbf{0}$ is an arbitrary vector then

$$\mathbf{0} = \boldsymbol{\mathcal{H}} g(\mathbf{x}) = \boldsymbol{\mathcal{H}}(\mathbf{A}\cdot\mathbf{x}) = \sum_b V_b (\mathbf{A}\cdot\mathbf{x}) \boldsymbol{\mathcal{H}} w_b(\mathbf{x}, h_b). \tag{3.49}$$

By noting that $(\mathbf{a}\cdot\mathbf{b})\mathbf{C} = (\mathbf{C}\otimes\mathbf{b})\mathbf{a}$ for any vectors $\mathbf{a}$, $\mathbf{b}$ and second order tensor $\mathbf{C}$ the above can be written as

$$\mathbf{0} = \left[\sum_b V_b (\boldsymbol{\mathcal{H}} w_b(\mathbf{x}, h_b) \otimes \mathbf{x}_b)\right]\mathbf{A}.$$

Since this must hold for any vector $\mathbf{A}$,

$$\mathbf{0} = \sum_b V_b (\boldsymbol{\mathcal{H}} w_b(\mathbf{x}, h_b) \otimes \mathbf{x}_b). \tag{3.50}$$

Using equation (3.48) and subtracting the zero term $\sum_b V_b \left(\boldsymbol{\mathcal{H}} w_b \left(\mathbf{x}, h_b\right) \otimes \mathbf{x}\right)$ gives the linear Hessian consistency condition

$$\mathbf{0} = \sum_b V_b \left(\boldsymbol{\mathcal{H}} w_b \left(\mathbf{x}, h_b\right) \otimes \left(\mathbf{x}_b - \mathbf{x}\right)\right) . \tag{3.51}$$

By evaluating the consistency conditions (3.48) and (3.51) at each particle a using the corrected kernel Hessian, the correction parameters $\mathbf{B}_a$ and $\boldsymbol{\mathcal{A}}_a$ can be obtained to enforce constant and linear Hessian consistency at each particle,

$$\mathbf{0} = \sum_b V_b \tilde{\boldsymbol{\mathcal{H}}} w_b \left(\mathbf{x}_a, h_b\right) \qquad \text{for each } a \ , \tag{3.52}$$

$$\mathbf{0} = \sum_b V_b \left(\tilde{\boldsymbol{\mathcal{H}}} w_b \left(\mathbf{x}_a, h_b\right) \otimes \left(\mathbf{x}_b - \mathbf{x}_a\right)\right) \quad \text{for each } a. \tag{3.53}$$

Substituting corrected Hessian into equation (3.52) gives

$$\sum_b V_b \left[\boldsymbol{\mathcal{H}} w_b \left(\mathbf{x}_a, h_b\right) + \delta_{ab} \mathbf{B}_a + \boldsymbol{\mathcal{A}}_a \nabla w_b \left(\mathbf{x}_a, h_b\right)\right] = \mathbf{0} \tag{3.54}$$

from which $\mathbf{B}_a$ is obtained as

$$\mathbf{B}_a = -\frac{1}{V_a} \left(\sum_b V_b \boldsymbol{\mathcal{H}} w_b \left(\mathbf{x}_a, h_b\right) + \sum_b V_b \boldsymbol{\mathcal{A}}_a \nabla w_b \left(\mathbf{x}_a, h_b\right) \right) . \tag{3.55}$$

Similarly, substituting corrected Hessian into equation (3.53) gives

$$\mathbf{0} = \sum_b V_b \left[\boldsymbol{\mathcal{H}} w_b \left(\mathbf{x}_a, h_b\right) + \delta_{ab} \mathbf{B}_a + \boldsymbol{\mathcal{A}}_a \nabla w_b \left(\mathbf{x}_a, h_b\right)\right] \otimes \left(\mathbf{x}_b - \mathbf{x}_a\right). \tag{3.56}$$

Noting that $\delta_{ab} \mathbf{B}_a \otimes \left(\mathbf{x}_b - \mathbf{x}_a\right) = \mathbf{0}$ gives

$$\boldsymbol{\mathcal{A}}_a \sum_b V_b \nabla w_b \left(\mathbf{x}_a, h_b\right) \otimes \left(\mathbf{x}_b - \mathbf{x}_a\right) = - \sum_b V_b \boldsymbol{\mathcal{H}} w_b \left(\mathbf{x}_a, h_b\right) \otimes \left(\mathbf{x}_b - \mathbf{x}_a\right) \tag{3.57}$$

from which $\boldsymbol{\mathcal{A}}_a$ is obtained as

$$\boldsymbol{\mathcal{A}}_a = - \left[\sum_b V_b \boldsymbol{\mathcal{H}} w_b \left(\mathbf{x}_a, h_b\right) \otimes \mathbf{x}_{ba} \right] \left[\sum_b V_b \nabla w_b \left(\mathbf{x}_a, h_b\right) \otimes \mathbf{x}_{ba} \right]^{-1} \tag{3.58}$$

where $\mathbf{x}_{ba} = \mathbf{x}_b - \mathbf{x}_a$.

Substituting the corrected Hessian into the equation (3.40) for $\tilde{\tilde{\nabla}} g_h(\mathbf{x}_a)$ gives

$$\begin{aligned}
\tilde{\tilde{\nabla}} g_h(\mathbf{x}_a) &= \sum_b V_b g(\mathbf{x}_b)\, \tilde{\nabla}\hat{w}_b(\mathbf{x}_a, h_b) + \frac{1}{2}\eta \tilde{\boldsymbol{\mathcal{H}}} g(\mathbf{x}_a)\, \mathbf{d} \\
&= \sum_b V_b g(\mathbf{x}_b)\, \tilde{\nabla}\hat{w}_b(\mathbf{x}_a, h_b) + \frac{1}{2}\eta \left(\sum_b V_b g(\mathbf{x}_b)\, \tilde{\boldsymbol{\mathcal{H}}} w_b(\mathbf{x}_a, h_b) \right) \mathbf{d} \\
&= \sum_b V_b g(\mathbf{x}_b) \left(\tilde{\nabla}\hat{w}_b(\mathbf{x}_a, h_b) + \frac{1}{2}\eta \tilde{\boldsymbol{\mathcal{H}}} w_b(\mathbf{x}_a, h_b)\, \mathbf{d} \right) \\
&= \sum_b V_b g(\mathbf{x}_b)\, \tilde{\tilde{\nabla}}\hat{w}_b(\mathbf{x}_a, h_b)
\end{aligned} \tag{3.59}$$

where the corrected and stabilized kernel function $\tilde{\tilde{\nabla}}\hat{w}_b(\mathbf{x}_a, h_b)$ is identified as

$$\tilde{\tilde{\nabla}}\hat{w}_b(\mathbf{x}_a, h_b) := \tilde{\nabla}\hat{w}_b(\mathbf{x}_a, h_b) + \frac{1}{2}\eta \tilde{\boldsymbol{\mathcal{H}}} w_b(\mathbf{x}_a, h_b)\, \mathbf{d} \ . \tag{3.60}$$

$\tilde{\tilde{\nabla}} g_h(\mathbf{x}_a)$ will still correctly interpolate the gradient of constant and linear functions since the first term of the stabilized kernel is given by $\tilde{\nabla}\hat{w}_b(\mathbf{x}_a, h_b)$ and the additional stabilizing term will be identically zero.

3.6 Conservation properties of corrected SPH

In absence of any external forces the traditional SPH method with constant smoothing lengths will conserve the linear momentum and the angular momentum of the system (if the stress tensor in question is isotropic [19]).

The conservation properties of the corrected SPH method were studied in greater detail in Lok [84]. This section will demonstrate that kernel corrections ensure the conservation of linear and angular momentum in SPH simulations with variable smoothing lengths.

Linear momentum

The total linear momentum of a system of N SPH particles can be written as the sum of the linear momentum of each particle

$$\mathbf{M} = \sum_{a=1}^{N} m_a \mathbf{v}_a \ . \tag{3.61}$$

Ignoring external forces and using Newton's 2nd Law, the time rate of change of linear momentum may be written as

$$\dot{\mathbf{M}} = \sum_{a=1}^{N} m_a \dot{\mathbf{v}}_a = -\sum_{a=1}^{N} \mathbf{T}_a \tag{3.62}$$

where $\mathbf{T}_a$ is the internal force acting on particle a. $\mathbf{T}_a$ is given by the sum of interaction forces between pairs of particles $\mathbf{T}_{ab}$.

Therefore conservation of linear momentum is ensured by satisfying the following discrete condition for any distribution of stress

$$\sum_{a=1}^{N} \mathbf{T}_a = \mathbf{0} \ . \tag{3.63}$$

From the discretizations of momentum equation found in equation (2.66) two forms for the particle interaction forces are given by

$$\mathbf{T}_{ab} = V_a V_b \left(\boldsymbol{\sigma}_a + \boldsymbol{\sigma}_b\right) \nabla w_b \left(\mathbf{x}_a\right) \quad \text{or} \quad \mathbf{T}_{ab} = m_a m_b \left(\frac{\boldsymbol{\sigma}_a}{\rho_a^2} + \frac{\boldsymbol{\sigma}_b}{\rho_b^2}\right) \nabla w_b \left(\mathbf{x}_a\right) . \tag{3.64}$$

The condition in equation (3.63) is automatically satisfied if constant smoothing lengths are used since $\nabla w_a \left(\mathbf{x}_b, h\right) = -\nabla w_b \left(\mathbf{x}_a, h\right)$ and consequently $\mathbf{T}_{ab} = -\mathbf{T}_{ba}$. Therefore the sum of all interaction pairs will vanish as required.

Linear momentum can also be conserved when using variable smoothing lengths but only if kernel correction is employed. Using equation (2.57) with corrected gradients the sum of internal forces can now be written as

$$\sum_a \mathbf{T}_a = \sum_{a,b} V_a V_b \boldsymbol{\sigma}_b \tilde{\nabla} \hat{w}_a \left(\mathbf{x}_b, h_a\right) = \sum_b V_b \boldsymbol{\sigma}_b \left(\sum_a V_a \tilde{\nabla} \hat{w}_a \left(\mathbf{x}_b, h_a\right)\right) = \mathbf{0} \ . \tag{3.65}$$

The term in brackets is identically equal to zero since it is the constant consistency condition given by equation (3.26). Consequently equation (3.63) is satisfied and linear momentum will be conserved as long as the kernel is at least constantly corrected.

Angular momentum

The angular momentum of a particle a of mass m_a with position $\mathbf{x}_a$ and moving with constant velocity $\mathbf{v}_a$ is defined to be the moment of linear momentum about the origin

$$\mathbf{A}_a = \mathbf{x}_a \times m_a \mathbf{v}_a. \tag{3.66}$$

The total angular momentum of a system of N SPH particles can be written as the sum of the angular momentum of each particle as

$$\mathbf{A} = \sum_{a=1}^{N} \mathbf{x}_a \times m_a \mathbf{v}_a. \tag{3.67}$$

The rate of change of angular momentum of the particle a is then calculated given by

$$\begin{aligned} \dot{\mathbf{A}}_a &= \dot{\mathbf{x}}_a \times m_a \mathbf{v}_a + \mathbf{x}_a \times m_a \dot{\mathbf{v}}_a \\ &= \mathbf{x}_a \times m_a \dot{\mathbf{v}}_a \qquad (\text{since } \mathbf{v} \times m\mathbf{v} = \mathbf{0}) \\ &= -\left(\mathbf{x}_a \times \mathbf{T}_a\right). \end{aligned} \tag{3.68}$$

Therefore, if the resultant moment about a fixed point of all the forces acting on a particle is zero over a timestep, then the angular momentum, $\mathbf{A}_a$ about that point must be constant.

As before ignoring external forces and using Newton's 2$^{\text{nd}}$ Law, the rate of change of angular momentum for the system of particles may be written as

$$\dot{\mathbf{A}} = \sum_{a=1}^{N} \mathbf{x}_a \times m_a \dot{\mathbf{v}}_a = -\sum_{a=1}^{N} \mathbf{x}_a \times \mathbf{T}_a \ . \tag{3.69}$$

Consequently, if the total moment of the internal forces equals zero for any distribution of stress then angular momentum will be conserved

$$\sum_{a=1}^{N} \mathbf{x}_a \times \mathbf{T}_a = \mathbf{0} \ . \tag{3.70}$$

Rearranging the above equation gives

$$\dot{\mathbf{A}} = -\sum_{a} \mathbf{x}_a \times \mathbf{T}_a = \sum_{a} \mathbf{T}_a \times \mathbf{x}_a = \sum_{a} \boldsymbol{\varepsilon} : (\mathbf{T}_a \otimes \mathbf{x}_a) \tag{3.71}$$

where $\boldsymbol{\varepsilon}$ is the alternating tensor ($\varepsilon^{ijk} = -1, 0, 1$). Substituting equation (2.57) for $\mathbf{T}_a$ gives $\dot{\mathbf{A}}$ as

$$\begin{aligned} \dot{\mathbf{A}} &= \sum_a \boldsymbol{\varepsilon} : \left(\sum_b V_a V_b \boldsymbol{\sigma}_b \tilde{\nabla} \hat{w}_a \left(\mathbf{x}_b, h_a \right) \otimes \mathbf{x}_a \right) \\ &= \sum_b V_b \boldsymbol{\varepsilon} : \boldsymbol{\sigma}_b \left(\sum_a V_a \tilde{\nabla} \hat{w}_a \left(\mathbf{x}_b, h_a \right) \otimes \mathbf{x}_a \right) . \end{aligned} \tag{3.72}$$

The above product will only vanish if the following condition is satisfied for any stress distribution

$$\sum_a V_a \tilde{\nabla} \hat{w}_a \left(\mathbf{x}_b, h_a \right) \otimes \mathbf{x}_a = \mathbf{I} . \tag{3.73}$$

This condition is the linear consistency condition for the kernel gradient given by equation (3.32). Consequently, when deviatoric stress tensors are implemented angular momentum will only be preserved if linear correction or the mixed kernel and gradient correction scheme is implemented.

3.7 Concluding remarks

This chapter has described several different techniques which can significantly improve the accuracy and stability of the standard SPH equations with minimal computational expense.

Kernel and gradient corrections have been introduced as a way to enforce constant and linear consistency of the discrete SPH approximations. Additional higher order Hessian correction terms have also been introduced to the kernel gradient to further stabilize the method when required.

Conservation of linear and angular momentum can now be established for SPH simulations without the restriction of uniform particle smoothing lengths when kernel and gradient corrections are implemented.

These correction terms are an essential ingredient in the variable resolution SPH formulation that will be derived in subsequent chapters and in Chapter 7 the kernel Hessian corrections will provide the necessary stabilization in a number of the particle refinement simulations.

Chapter 4

Variational formulation of SPH

4.1 Introduction

In order to implement particle refinement into the SPH framework the underlying formulation must first be able to cope with non-uniform particle masses and smoothing lengths. In this chapter such a formulation is derived from variational principles and the equivalent internal forces in the continuum are obtained.

The particles represent points in the continuum rather than discrete free particles. The derivation depends upon which of the two SPH formulations for the density are used. In both cases the resulting expressions for the internal forces are found to take the same form as those derived in Section 2.5.

A further correction term γ is introduced into the direct density variational formulation in order to improve the density evaluation in the vicinity of solid boundaries. This term leads to an additional internal force that can be identified as a boundary contact force due to the presence of the boundary.

In the final section the conservation properties that were derived in Section 3.6 using kernel corrections are derived from variational principles. Linear consistency of the SPH approximation for the gradient of functions is shown to be essential if the method is to preserve angular momentum.

4.2 Variational derivation

The continuum is to be discretized by a large set of particles. Each particle a is described by its mass m_a, position $\mathbf{x}_a$, and velocity $\mathbf{v}_a$ as shown in Figure 4.1. Let $\mathbf{v}$ denote the set of all particle velocities and $\mathbf{x}$ the set of all particle positions that define the state of the continuum.

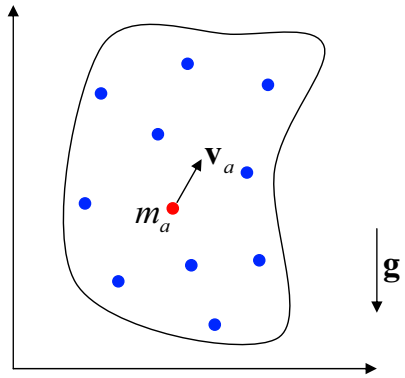


Figure 4.1: Discretized continuum

The *total kinetic energy* of the system is approximated by the sum of the kinetic energy of each particle

$$K(\mathbf{v}) = \frac{1}{2}\sum_a m_a(\mathbf{v}_a \cdot \mathbf{v}_a). \tag{4.1}$$

The work done by external forces resulting from a gravitational field $\mathbf{g}$ gives the *total external energy* as

$$\Pi_{\text{ext}}(\mathbf{x}) = -\sum_a m_a(\mathbf{x}_a \cdot \mathbf{g}). \tag{4.2}$$

The *total internal energy* of the system depends on the constitutive characteristics of the continuum. The total internal energy is expressed as the sum of energy accumulated per unit mass π multiplied by the particle masses

$$\Pi_{\text{int}}(\mathbf{x}) = \sum_a m_a \pi(\rho_a, \ldots) \tag{4.3}$$

π will depend on the deformation, density, or any other constitutive parameters of the material in question.

From the Euler-Lagrange Equations of Motion the equilibrium equation for the

system of particles representing the continuum is expressed as

$$\begin{gathered}\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{v}_a}\right) - \frac{\partial \mathcal{L}}{\partial \mathbf{x}_a} = 0 \\ \text{where} \quad \mathcal{L}\left(\mathbf{x}, \mathbf{v}\right) = K\left(\mathbf{v}\right) - \Pi_{\text{int}}\left(\mathbf{x}\right) - \Pi_{\text{ext}}\left(\mathbf{x}\right).\end{gathered} \tag{4.4}$$

Substituting equations (4.1–4.3) into the above expression leads to the standard Newton's 2nd Law for each particle

$$m_a \mathbf{a}_a = -\frac{\partial \Pi_{\text{ext}}}{\partial \mathbf{x}_a} - \frac{\partial \Pi_{\text{int}}}{\partial \mathbf{x}_a} = \mathbf{F}_a - \mathbf{T}_a \tag{4.5}$$

where $\mathbf{a}_a$ is the acceleration of particle a and $\mathbf{F}_a$ is the *external force* acting on particle a. For the simple gravitational case this is given by

$$\mathbf{F}_a = -\frac{\partial \Pi_{\text{ext}}}{\partial \mathbf{x}_a} = m_a \mathbf{g}. \tag{4.6}$$

$\mathbf{T}_a$ is the *internal force* acting on particle a given by

$$\mathbf{T}_a = \frac{\partial \Pi_{\text{int}}}{\partial \mathbf{x}_a} = \frac{\partial}{\partial \mathbf{x}_a} \sum_b m_b \pi\left(\rho_b, \ldots\right) . \tag{4.7}$$

When the above forces are evaluated in accordance with the above formulation the resulting expressions will be consistent with the preservation of linear and angular momentum [19] which will be discussed further in Section 4.6.

It should also be noted that the above formulation does not include any dissipative terms. Boundary friction and viscous effects can be included with the addition of a dissipative potential into the Euler-Lagrange equation

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{v}_a}\right) - \frac{\partial \mathcal{L}}{\partial \mathbf{x}_a} = -\frac{\partial \Pi_{\text{dis}}}{\partial \partial \mathbf{v}_a} . \tag{4.8}$$

Assuming that $\mathbf{F}_a$ and $\mathbf{T}_a$ can be calculated at each particle according to equation (4.6) and equation (4.7) it is simple to obtain the acceleration of each particle as

$$\mathbf{a}_a = \frac{1}{m_a}\left(\mathbf{F}_a - \mathbf{T}_a - \mathbf{T}_a^{\text{dev}}\right) \tag{4.9}$$

where $\mathbf{T}_a^{\text{dev}}$ is the deviatoric component of the internal force (see Section 4.4). Consequently, particle positions may be updated simply using a leap-frog integration scheme as detailed in Section 2.6.

It remains to derive discrete equations for the internal forces $\mathbf{T}_a$. To this end equation (4.3) is linearised at the current position, in the direction of the set of virtual velocities $\delta\mathbf{v}$ as shown in Figure 4.2.

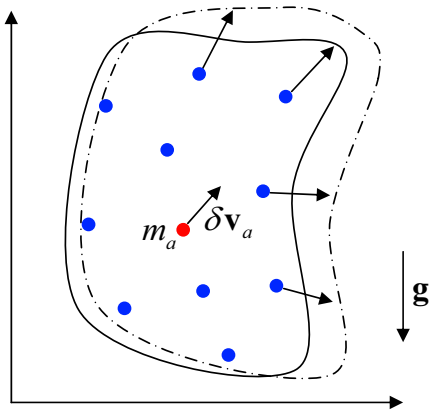


Figure 4.2: Linearised continuum with respect to $\delta \mathbf{v}$

The internal and contact forces can now be identified by the differentiation of the internal energy equation

$$\begin{aligned} D\Pi_{\text{int}}\left[\delta \mathbf{v}\right] &= \frac{d}{d\varepsilon}\Pi_{\text{int}}\left(\mathbf{x}_1 + \varepsilon\delta\mathbf{v}_1, \cdots, \mathbf{x}_N + \varepsilon\delta\mathbf{v}_N\right)\Big|_{\varepsilon=0} \\ &= \sum_{a=1} \frac{\partial \Pi_{\text{int}}}{\partial\left(\mathbf{x}_a + \varepsilon\delta\mathbf{v}_a\right)} \cdot \frac{d}{d\varepsilon}\left(\mathbf{x}_a + \varepsilon\delta\mathbf{v}_a\right)\Big|_{\varepsilon=0} \\ &= \sum_{a=1} \frac{\partial \Pi_{\text{int}}}{\partial \mathbf{x}_a} \cdot \delta\mathbf{v}_a \\ D\Pi_{\text{int}}\left[\delta \mathbf{v}\right] &= \sum_{a=1} \mathbf{T}_a \cdot \delta\mathbf{v}_a \ . \end{aligned} \tag{4.10}$$

This provides the framework to construct the dynamic equilibrium equations.

Recall, the total internal energy of the continuum described by a set of particles in state $\mathbf{x}$ is given as

$$\Pi_{\text{int}}\left(\mathbf{x}\right) = \sum_{a} m_a \pi\left(\rho_a\right) \tag{4.11}$$

where $\pi\left(\rho\right)$ is the internal energy per unit mass.

For fluids under reversible, adiabatic conditions the derivative of the internal energy is related to the pressure by

$$\frac{d\pi}{d\rho} = \frac{P}{\rho^2} \tag{4.12}$$

where it is understood that a negative pressure P indicates tension.

In this case the expression for $D\Pi_{\text{int}}[\delta\mathbf{v}]$ becomes

$$\begin{aligned} D\Pi_{\text{int}}[\delta\mathbf{v}] &= \sum_a m_a \frac{d\pi}{d\rho_a} D\rho_a[\delta\mathbf{v}] \\ &= \sum_a m_a \frac{P_a}{\rho_a^2} D\rho_a[\delta\mathbf{v}]. \end{aligned} \tag{4.13}$$

From this point, on the expression for the derivative of the density $D\rho_a[\delta\mathbf{v}]$ will depend on which SPH formulation for the density is followed. The continuity method and the direct density method will result in different expressions for the rate of change of density and consequently different expressions for the internal forces.

4.3 SPH with variable smoothing length

In this section the different expressions for the linearization of the particle density, $D\rho_a[\delta\mathbf{v}]$ will be derived. It is assumed throughout that the individual particle smoothing lengths are not equal. However, simplifications to the resulting expressions that arise from assuming a uniform smoothing length will be given where appropriate.

4.3.1 Continuity density form

In the first case the rate of change of density is related to the divergence of the velocity by the continuity equation

$$\frac{D\rho}{Dt} = -\rho\ \nabla\cdot\mathbf{v}. \tag{4.14}$$

Replacing the actual velocity with a virtual velocity field $\delta\mathbf{v}$ gives the variation in ρ

$$D\rho[\delta\mathbf{v}] = -\rho\ \nabla\cdot\delta\mathbf{v}. \tag{4.15}$$

Substituting equation (4.15) into equation (4.13) gives

$$D\Pi_{\text{int}}[\delta\mathbf{v}] = -\sum_a V_a P_a\,(\nabla\cdot\delta\mathbf{v}_a). \tag{4.16}$$

The divergence of the virtual velocities $\nabla\cdot\delta\mathbf{v}_a$ are formulated in the SPH framework by using the summation approximation for $\delta\mathbf{v}$ given by

$$\delta\mathbf{v} = \sum_b V_b \delta\mathbf{v}_b w_b(\mathbf{x}, h_b). \tag{4.17}$$

With constant consistency enforced $\nabla \cdot \delta \mathbf{v}_a$ is given by

$$\nabla \cdot \delta \mathbf{v}_a = \sum_b V_b \left(\delta \mathbf{v}_b - \delta \mathbf{v}_a \right) \cdot \nabla w_b \left(\mathbf{x}_a, h_b \right) \tag{4.18}$$

and $D\rho_a \left[\delta \mathbf{v} \right]$ is obtained as

$$D\rho_a \left[\delta \mathbf{v} \right] = -\rho_a \sum_b V_b \left(\delta \mathbf{v}_b - \delta \mathbf{v}_a \right) \cdot \nabla w_b \left(\mathbf{x}_a, h_b \right). \tag{4.19}$$

Remark

The variational formulation of the time derivative of the density using the continuity equation is now obtained by setting $\delta \mathbf{v}_a = \mathbf{v}_a$ and $\delta \mathbf{v}_b = \mathbf{v}_b$ in equation (4.19) to give

$$\begin{aligned} \dot{\rho}_a = -\rho_a \left(\nabla \cdot \mathbf{v}_a \right) &= -\rho_a \sum_b V_b \left(\mathbf{v}_b - \mathbf{v}_a \right) \cdot \nabla w_b \left(\mathbf{x}_a, h_b \right) \\ &= \rho_a \sum_b V_b \left(\mathbf{v}_a - \mathbf{v}_b \right) \cdot \nabla w_b \left(\mathbf{x}_a, h_b \right). \end{aligned} \tag{4.20}$$

It should be noted that the above expression coincides with equation (2.44) from the previous chapter. When the corrected kernel is used the above equation simplifies to give

$$\dot{\rho}_a = -\rho_a \sum_b V_b \mathbf{v}_b \cdot \tilde{\nabla} \hat{w}_b \left(\mathbf{x}_a, h_b \right). \tag{4.21}$$

Internal forces

The internal forces can now be obtained by substituting the expression for $\nabla \cdot \delta \mathbf{v}_a$ as given by equation (4.18) into equation (4.16)

$$\begin{aligned} D\Pi_{\text{int}} \left[\delta \mathbf{v} \right] &= -\sum_a V_a P_a \sum_b V_b \left(\delta \mathbf{v}_b - \delta \mathbf{v}_a \right) \cdot \nabla w_b \left(\mathbf{x}_a, h_b \right) \\ &= -\sum_{a,b} V_a V_b P_a \nabla w_b \left(\mathbf{x}_a, h_b \right) \cdot \delta \mathbf{v}_b + \sum_{a,b} V_a V_b P_a \nabla w_b \left(\mathbf{x}_a, h_b \right) \cdot \delta \mathbf{v}_a. \end{aligned}$$

Rewriting in terms of $\delta \mathbf{v}_a$ gives

$$D\Pi_{\text{int}} \left[\delta \mathbf{v} \right] = \sum_a \left(\sum_b V_a V_b \left(P_a \nabla w_b \left(\mathbf{x}_a, h_b \right) - P_b \nabla w_a \left(\mathbf{x}_b, h_a \right) \right) \right) \cdot \delta \mathbf{v}_a. \tag{4.22}$$

From the above expression the *internal pressure force* is obtained as

$$\mathbf{T}_a^P = \sum_b V_a V_b \left(P_a \nabla w_b \left(\mathbf{x}_a, h_b \right) - P_b \nabla w_a \left(\mathbf{x}_b, h_a \right) \right). \tag{4.23}$$

If all particle smoothing lengths are identical so that $h_a = h_b = h$ then the pressure force can be rewritten as

$$\mathbf{T}_a^P = -\sum_b V_a V_b \left(P_a + P_b \right) \nabla w_a \left(\mathbf{x}_b, h \right) ,$$

or

$$\mathbf{T}_a^P = \sum_b V_a V_b \left(P_a + P_b \right) \nabla w_b \left(\mathbf{x}_a, h \right)$$

and the above equations coincide with the pressure term in the top equation of (2.66).

If the corrected kernel is used then the pressure force reduces to its simplest form given by

$$\mathbf{T}_a^P = -\sum_b V_a V_b P_b \tilde{\nabla} \hat{w}_a \left(\mathbf{x}_b, h_a \right). \tag{4.24}$$

4.3.2 Direct density form

The second approach is to use the classical SPH equation for the density to derive the internal forces. In this case the density is given by

$$\rho(\mathbf{x}) = \sum_b m_b w_b(\mathbf{x}, h_b). \tag{4.25}$$

This equation can be reinterpreted as a smoothing of a discrete density approximation defined by a collection of point masses m_b with position $\mathbf{x}_b$

$$\hat{\rho}(\mathbf{x}) = \sum_b m_b \delta(\mathbf{x} - \mathbf{x}_b) \tag{4.26}$$

where $\delta(\mathbf{x} - \mathbf{x}_b)$ is the Dirac delta function based at the point b (see Figure 4.3).

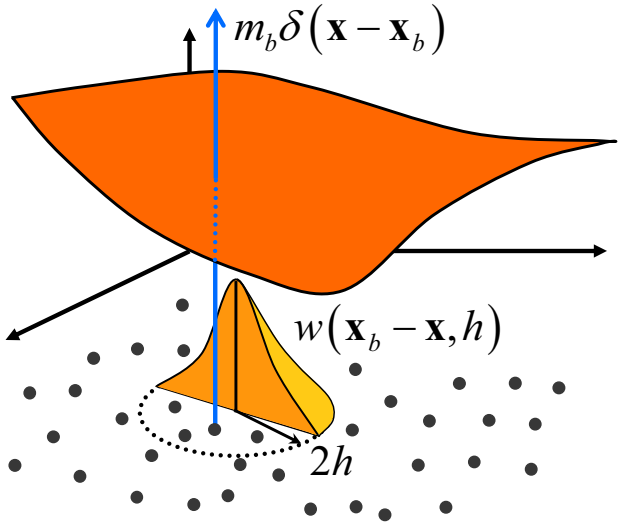


Figure 4.3: Discrete density approximation at point $\mathbf{x}$

Smoothing $\hat{\rho}(\mathbf{x})$ with the kernel function $w(\mathbf{x})$ gives

$$\rho(\mathbf{x}) = \frac{\int \hat{\rho}(\mathbf{x}')\, w(\mathbf{x} - \mathbf{x}', h)\, d\mathbf{x}'}{\int w(\mathbf{x}, h)\, d\mathbf{x}}. \tag{4.27}$$

Therefore, for a given collection of particle masses the discrete density is given by

$$\rho(\mathbf{x}_a) = \frac{\sum_b m_b w_b(\mathbf{x}_a, h_b)}{\gamma(\mathbf{x}_a, h_a)} \quad \text{where} \quad \gamma(\mathbf{x}_a, h_a) := \int w_a(\mathbf{x}, h_a)\, d\mathbf{x}. \tag{4.28}$$

Gamma function

The *gamma function* is defined to be

$$\gamma\left(\mathbf{x}_a, h_a\right) := \int w_a\left(\mathbf{x}, h_a\right) d\mathbf{x}. \tag{4.29}$$

The introduction of the correction function $\gamma\left(\mathbf{x}_a, h_a\right)$ ensures that the density is accurately evaluated in the vicinity of rigid boundaries. Without this term the direct density evaluation can lead to poor interpolations adjacent to boundaries since the number of neighbour particles reduces as the kernel's support falls partially outside the domain.

It is clear that if the kernel function is normalised to give a unit integral and is based sufficiently far from any boundary then $\gamma\left(\mathbf{x}_a, h_a\right) = 1$ and equation (4.28) is equal to the classical SPH equation for density as given by equation (4.25).

However, when a boundary falls within the compact support ($2h$ – for the quintic kernel) of a particle this is not the case and the correction term $\gamma\left(\mathbf{x}_a, h_a\right) \neq 1$ and will contribute to the density evaluation for that particle.

In the vicinity of a straight boundary the gamma function can be evaluated by considering it as a function of the perpendicular distance between the particle and boundary as shown in Figure 4.4.

In this case

$$\gamma\left(\mathbf{x}_a, h_a\right) = \gamma\left(\varepsilon_a\right) \quad \text{where} \quad \varepsilon_a = \frac{d_a}{h_a} \quad \text{and} \quad d_a = \left(\mathbf{x}_a - \mathbf{x}_B\right) \cdot \mathbf{n}_B \tag{4.30}$$

where d_a is the distance of particle a from boundary, $\mathbf{x}_B$ is any point on the boundary with unit inward normal $\mathbf{n}_B$. Then the derivative $D\gamma_a\left[\delta\mathbf{v}\right]$ is obtained by linearising equation (4.30) to give

$$D\gamma_a\left[\delta\mathbf{v}\right] = \frac{1}{h_a}\gamma'\left(\varepsilon_a\right)\mathbf{n}_B \cdot \delta\mathbf{v}_a. \tag{4.31}$$

In Chapter 5 a general method for calculating $\gamma\left(\mathbf{x}_a, h_a\right)$ and $D\gamma_a\left[\delta\mathbf{v}\right]$ is given for general boundaries in 2-D.

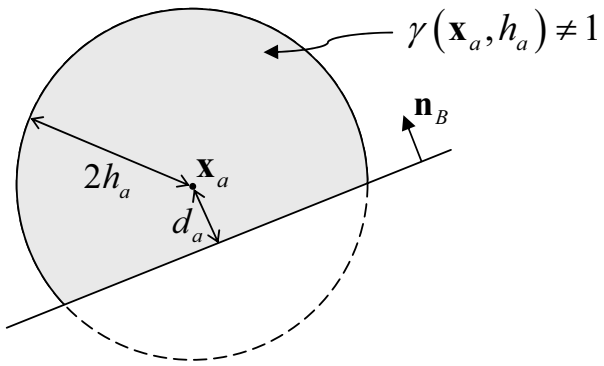


Figure 4.4: γ function in the vicinity of a single boundary

Assuming that the gamma function and its derivative can be suitably calculated it remains to linearise the following density equation with respect to particle positions

$$\rho(\mathbf{x}_a) = \frac{\sum_b m_b w_b(\mathbf{x}_a, h_b)}{\gamma(\mathbf{x}_a, h_a)} \,. \tag{4.32}$$

The derivation now depends on whether particle smoothing lengths h_a remain constant throughout the simulation $h_a = h$ or are allowed to vary as a function of space and time $h_a(\mathbf{x}, t)$ as in [21, 112].

Since the focus of this thesis is on nearly incompressible flows it will be assumed that the smoothing lengths of the particles are constant. However, with the addition of particle refinement introduced in Chapter 6 particle smoothing lengths may change as the simulation evolves.

Under this assumption equation (4.32) gives

$$\gamma_a D\rho_a[\delta\mathbf{v}] + \rho_a D\gamma_a[\delta\mathbf{v}] = D\left(\sum_b m_b w_b(\mathbf{x}_a, h_b)\right)[\delta\mathbf{v}] \,. \tag{4.33}$$

Linearising the right hand side gives

$$\sum_b m_b Dw_b(\mathbf{x}_a, h_b)[\delta\mathbf{v}] = \sum_b m_b \frac{dw}{dr_{ab}} Dr_{ab}[\delta\mathbf{v}] \tag{4.34}$$

where $r_{ab}^2 = (\mathbf{x}_a - \mathbf{x}_b)\cdot(\mathbf{x}_a - \mathbf{x}_b)$ and

$$Dr_{ab}[\delta\mathbf{v}] = \frac{1}{r_{ab}}(\mathbf{x}_a - \mathbf{x}_b)\cdot(\delta\mathbf{v}_a - \delta\mathbf{v}_b) \,. \tag{4.35}$$

Finally, $D\rho_a[\delta\mathbf{v}]$ is obtained as

$$\begin{aligned} D\rho_a[\delta\mathbf{v}] &= \frac{1}{\gamma_a}\sum_b m_b \frac{1}{r_{ab}}\frac{dw}{dr_{ab}}(\mathbf{x}_a-\mathbf{x}_b)\cdot(\delta\mathbf{v}_a-\delta\mathbf{v}_b) - \frac{\rho_a}{\gamma_a}D\gamma_a[\delta\mathbf{v}] \\ &= \frac{1}{\gamma_a}\sum_b m_b \nabla w_b(\mathbf{x}_a,h_b)\cdot(\delta\mathbf{v}_a-\delta\mathbf{v}_b) - \frac{\rho_a}{\gamma_a}D\gamma_a[\delta\mathbf{v}]. \end{aligned} \tag{4.36}$$

Remark

The variational formulation of the time derivative of the density using the direct density equation is now obtained by setting $\delta\mathbf{v}_a = \mathbf{v}_a$ and $\delta\mathbf{v}_b = \mathbf{v}_b$ in equation (4.36) to give

$$\dot{\rho}_a = \frac{1}{\gamma_a}\sum_b m_b \nabla w_b(\mathbf{x}_a,h_b)\cdot(\mathbf{v}_a-\mathbf{v}_b) - \frac{\rho_a}{\gamma_a}D\gamma_a[\delta\mathbf{v}]. \tag{4.37}$$

Noting that away from any boundaries $\gamma_a = 1$ and $\gamma_a' = 0$ the above density equation reduces to equation (2.45) found in the previous chapter.

Internal and contact forces

The internal and contact forces can now be obtained by substituting $D\rho_a[\delta\mathbf{v}]$ given by equation (4.36) into equation (4.13)

$$\begin{aligned} D\Pi_{\text{int}}[\delta\mathbf{v}] &= \sum_a m_a \frac{P_a}{\rho_a^2}\left(\frac{1}{\gamma_a}\sum_b m_b\nabla w_b(\mathbf{x}_a,h_b)\cdot(\delta\mathbf{v}_a-\delta\mathbf{v}_b) - \frac{\rho_a}{\gamma_a}D\gamma_a[\delta\mathbf{v}]\right) \\ &= \sum_a\sum_b m_a\frac{P_a}{\rho_a^2}\left(\frac{1}{\gamma_a}m_b\nabla w_b(\mathbf{x}_a,h_b)\cdot(\delta\mathbf{v}_a-\delta\mathbf{v}_b)\right) \\ &\qquad - \sum_a \frac{m_a P_a}{\rho_a\gamma_a}D\gamma_a[\delta\mathbf{v}]. \end{aligned} \tag{4.38}$$

Expanding the first term in equation (4.38)

$$\sum_a\sum_b m_a m_b \frac{P_a}{\rho_a^2\gamma_a}\nabla w_b(\mathbf{x}_a,h_b)\cdot\delta\mathbf{v}_a - \sum_a\sum_b m_a m_b\frac{P_a}{\rho_a^2\gamma_a}\nabla w_b(\mathbf{x}_a,h_b)\cdot\delta\mathbf{v}_b$$

and writing in terms of $\delta\mathbf{v}_a$ yields,

$$\begin{aligned} &\sum_a\sum_b m_a m_b\frac{P_a}{\rho_a^2\gamma_a}\nabla w_b(\mathbf{x}_a,h_b)\cdot\delta\mathbf{v}_a - \sum_a\sum_b m_b m_a\frac{P_b}{\rho_b^2\gamma_b}\nabla w_a(\mathbf{x}_b,h_a)\cdot\delta\mathbf{v}_a \\ &\quad = \sum_a\sum_b m_a m_b\left(\frac{P_a}{\rho_a^2\gamma_a}\nabla w_b(\mathbf{x}_a,h_b) - \frac{P_b}{\rho_b^2\gamma_b}\nabla w_a(\mathbf{x}_b,h_a)\right)\cdot\delta\mathbf{v}_a. \end{aligned} \tag{4.39}$$

From the above equations expressions for the *internal pressure force* denoted by $\mathbf{T}_a^P$ and the *boundary contact force* denoted by $\mathbf{T}_a^B$ are obtained respectively as

$$\mathbf{T}_a^P = \sum_b m_a m_b \left(\frac{P_a}{\rho_a^2 \gamma_a} \nabla w_b \left(\mathbf{x}_a, h_b \right) - \frac{P_b}{\rho_b^2 \gamma_b} \nabla w_a \left(\mathbf{x}_b, h_a \right) \right), \tag{4.40}$$

$$\mathbf{T}_a^B = -\frac{m_a P_a}{\rho_a \gamma_a} \nabla \gamma_a \tag{4.41}$$

and the total internal force is given by the sum of the internal pressure force and boundary contact force

$$\mathbf{T}_a = \mathbf{T}_a^P + \mathbf{T}_a^B. \tag{4.42}$$

If all particle smoothing lengths are identical so that $h_a = h_b = h$ the pressure forces are finally obtained as

$$\mathbf{T}_a^P = \sum_b m_a m_b \left(\frac{P_a}{\rho_a^2 \gamma_a} + \frac{P_b}{\rho_b^2 \gamma_b} \right) \nabla w_b \left(\mathbf{x}_a, h \right) \tag{4.43}$$

and the above equation coincides with the pressure term in the bottom equation of (2.66) in the absence of any solid boundary.

It was shown by Lok [84] that it is essential to apply these variational SPH equations in a consistent manner. For example, it should be noted that SPH kernel corrections cannot be used for updating particle densities when using the direct density method and that expressions for the particle force and density evaluation from different derivations should not be mixed.

4.4 Viscous forces

Thus far, two different formulations for the internal force of a given particle have been derived but in both cases only isotropic stress tensors were considered. The same approach can be used to obtain the total internal force at a particle for general materials with constitutive equations which include a stress tensor with a deviatoric component $\boldsymbol{\sigma} = \boldsymbol{\sigma}_{\text{vol}} + \boldsymbol{\sigma}'$.

For this purpose and in order to remain within a variational framework an additional *viscous potential* $\psi_{\text{vis}}(\mathbf{d})$ per unit volume is introduced. The viscous potential is assumed to be a function of the particle velocities and is written in terms of the rate of deformation tensor $\mathbf{d}$ defined as the symmetric part of the velocity gradient tensor

$$\mathbf{d} = \frac{1}{2}\left(\nabla\mathbf{v} + (\nabla\mathbf{v})^T\right). \tag{4.44}$$

As before $\mathbf{d}_a$ is calculated from the expression for $\nabla\mathbf{v}_a$ given by

$$\nabla\mathbf{v}_a = \sum_b V_b\left(\mathbf{v}_b - \mathbf{v}_a\right) \otimes \nabla w_b\left(\mathbf{x}_a, h_b\right) \tag{4.45}$$

and the viscous potential, written in terms of the set of particle velocities $\mathbf{v}$, is given by

$$\Pi_{\text{vis}}(\mathbf{v}) = \sum_a V_a \psi_{\text{vis}}(\mathbf{d}_a). \tag{4.46}$$

This can be interpreted as the sum of the rate of energy dissipated by viscous forces per unit volume for each particle in the continuum. The deviatoric stress tensor is then written in terms of the dissipative potential as

$$\boldsymbol{\sigma}' = \frac{\partial\psi_{\text{vis}}(\mathbf{d})}{\partial\mathbf{d}}. \tag{4.47}$$

The internal viscous forces are obtained by considering the derivative of the viscous potential in the direction of a set of virtual velocities $\delta\mathbf{v}$ give by

$$\begin{aligned} D\Pi_{\text{vis}}[\delta\mathbf{v}] &= \sum_a V_a \left(\frac{\partial\psi_{\text{vis}}}{\partial\left[\mathbf{d}_a\right]^{ij}} \frac{d\left[\mathbf{d}_a\left(\mathbf{v}_a + \varepsilon\delta\mathbf{v}\right)\right]^{ij}}{d\varepsilon}\right)\Bigg|_{\varepsilon=0} \\ &= \sum_a V_a \frac{\partial\psi_{\text{vis}}}{\partial\mathbf{d}_a} : D\mathbf{d}_a[\delta\mathbf{v}] \end{aligned} \tag{4.48}$$

where the derivative of the rate of deformation tensor is given by

$$\begin{aligned} D\mathbf{d}\left[\delta\mathbf{v}\right] &= \frac{d}{d\varepsilon}\left(\frac{1}{2}\left[\nabla\left(\mathbf{v}+\varepsilon\delta\mathbf{v}\right)+\left(\nabla\left(\mathbf{v}+\varepsilon\delta\mathbf{v}\right)\right)^{T}\right]\right)\Big|_{\varepsilon=0} \\ &= \frac{1}{2}\frac{d}{d\varepsilon}\left(\nabla\mathbf{v}+\nabla\mathbf{v}^{T}+\varepsilon\left(\delta\mathbf{v}+\delta\mathbf{v}^{T}\right)\right)\Big|_{\varepsilon=0} \\ &= \frac{1}{2}\left(\nabla\delta\mathbf{v}+\left(\nabla\delta\mathbf{v}\right)^{T}\right). \end{aligned} \tag{4.49}$$

The gradient of the virtual velocities $\nabla\delta\mathbf{v}_a$ are then obtained with constant consistency using equation (4.45) by

$$\nabla\delta\mathbf{v}_a = \sum_b V_b\left(\delta\mathbf{v}_b-\delta\mathbf{v}_a\right)\otimes\nabla w_b\left(\mathbf{x}_a,h_b\right). \tag{4.50}$$

Substituting equations (4.49, 4.50) into equation (4.48) gives

$$D\Pi_{\text{vis}}\left[\delta\mathbf{v}\right] = \sum_a V_a\boldsymbol{\sigma}'_a : \frac{1}{2}\left(\nabla\delta\mathbf{v}_a+\left(\nabla\delta\mathbf{v}_a\right)^{\mathrm{T}}\right) =$$

$$\frac{1}{2}\sum_{a,b} V_aV_b\left(\boldsymbol{\sigma}'_a : \left(\delta\mathbf{v}_b-\delta\mathbf{v}_a\right)\otimes\nabla w_b\left(\mathbf{x}_a,h_b\right)+\boldsymbol{\sigma}'_a : \nabla w_b\left(\mathbf{x}_a,h_b\right)\otimes\left(\delta\mathbf{v}_b-\delta\mathbf{v}_a\right)\right)$$

and since $\boldsymbol{\sigma}'_a$ is symmetric the above equation simplifies to

$$D\Pi_{\text{vis}}\left[\delta\mathbf{v}\right] = \sum_a\sum_b V_aV_b\boldsymbol{\sigma}'_a\nabla w_b\left(\mathbf{x}_a,h_b\right)\cdot\left(\delta\mathbf{v}_b-\delta\mathbf{v}_a\right). \tag{4.51}$$

Writing the derivative of the viscous potential in terms of $\delta\mathbf{v}_a$ gives

$$D\Pi_{\text{vis}}\left[\delta\mathbf{v}\right] = \sum_a\left(\sum_b V_aV_b\left(\boldsymbol{\sigma}'_a+\boldsymbol{\sigma}'_b\right)\nabla w_a\left(\mathbf{x}_b,h_a\right)\right)\cdot\delta\mathbf{v}_a \tag{4.52}$$

and the deviatoric components of the internal force is obtained from the above equation as

$$\mathbf{T}_a^{\text{dev}} = \sum_b V_aV_b\left(\boldsymbol{\sigma}'_a+\boldsymbol{\sigma}'_b\right)\nabla w_a\left(\mathbf{x}_b,h_a\right). \tag{4.53}$$

When the continuity method is used the volumetric and viscous components of the internal force can be added to give

$$\mathbf{T}_a = \sum_b V_aV_b\left(\boldsymbol{\sigma}_a+\boldsymbol{\sigma}_b\right)\nabla w_a\left(\mathbf{x}_b,h_a\right), \qquad \boldsymbol{\sigma} = -P\mathbf{I}+\boldsymbol{\sigma}'. \tag{4.54}$$

If kernel correction is used in the above equation the equation for the total internal force reduces to

$$\mathbf{T}_a = \sum_b V_aV_b\boldsymbol{\sigma}_b\tilde{\nabla}\hat{w}_a\left(\mathbf{x}_b,h_a\right), \qquad \boldsymbol{\sigma} = -P\mathbf{I}+\boldsymbol{\sigma}'. \tag{4.55}$$

4.4.1 Stress tensor for Newtonian fluids

For Newtonian fluids the viscous potential ψ_{vis} above is a function of the Von Mises equivalent strain rate $\dot{\varepsilon}$, where $\dot{\varepsilon}$ is defined as the second invariant of the deviatoric rate of deformation tensor $\mathbf{d}'$ given by

$$\dot{\varepsilon}^2 = \frac{2}{3}\mathbf{d}' : \mathbf{d}' \qquad \text{where} \qquad \mathbf{d}' = \mathbf{d} - \tfrac{1}{3}\text{tr}\,(\mathbf{d})\,\mathbf{I}\,. \tag{4.56}$$

The deviatoric stress tensor is therefore obtained as

$$\boldsymbol{\sigma}' = \frac{\partial \psi_{\text{vis}}\,(\dot{\varepsilon})}{\partial \mathbf{d}} = \frac{\partial \psi_{\text{vis}}}{\partial \dot{\varepsilon}}\frac{\partial \dot{\varepsilon}}{\partial \mathbf{d}}\,. \tag{4.57}$$

Rewriting equation (4.56) in the form

$$\begin{aligned}
\dot{\varepsilon}^2 &= \frac{2}{3}\left(\mathbf{d} - \tfrac{1}{3}\text{tr}\,(\mathbf{d})\,\mathbf{I}\right) : \left(\mathbf{d} - \tfrac{1}{3}\text{tr}\,(\mathbf{d})\,\mathbf{I}\right) \\
&= \frac{2}{3}\left[(\mathbf{d}:\mathbf{d}) - \tfrac{1}{3}\,(\text{tr}\;\mathbf{d})^2 - \tfrac{1}{3}\,(\text{tr}\;\mathbf{d})^2 + \tfrac{1}{3}\,(\text{tr}\;\mathbf{d})^2\right] \\
&= \frac{2}{3}\left[(\mathbf{d}:\mathbf{d}) - \tfrac{1}{3}\,(\text{tr}\;\mathbf{d})^2\right]
\end{aligned} \tag{4.58}$$

and differentiating with respect to the rate of deformation tensor $\mathbf{d}$ gives

$$2\dot{\varepsilon}\frac{\partial \dot{\varepsilon}}{\partial \mathbf{d}} = \frac{2}{3}\left[2\mathbf{d} - \frac{2}{3}\,(\text{tr}\mathbf{d})\,\mathbf{I}\right] = \frac{4}{3}\mathbf{d}'$$

$$\frac{\partial \dot{\varepsilon}}{\partial \mathbf{d}} = \frac{2}{3\dot{\varepsilon}}\mathbf{d}'. \tag{4.59}$$

Substituting the above expression into equation (4.57) gives

$$\boldsymbol{\sigma}' = \frac{\partial \psi_{\text{vis}}\,(\dot{\varepsilon})}{\partial \mathbf{d}} = 2\left(\frac{1}{3\dot{\varepsilon}}\frac{\partial \psi_{\text{vis}}}{\partial \dot{\varepsilon}}\right)\mathbf{d}' = 2\mu\mathbf{d}' \tag{4.60}$$

where μ is the *dynamic viscosity*. Its value can be obtained through experimentation. A typical value for water at one atmosphere and at room temperature is approximately $0.0011\text{kgm}^{-1}\text{s}^{-1}$.

Note that μ and ψ_{vis} are related by

$$\mu = \frac{1}{3\dot{\varepsilon}}\frac{\partial \psi_{\text{vis}}}{\partial \dot{\varepsilon}}\,. \tag{4.61}$$

Specifically for a constant viscosity, the viscous potential ψ_{vis} is obtained by integrating equation (4.61) to give

$$\psi_{\text{vis}} = \frac{3}{2}\mu\dot{\varepsilon}^2. \tag{4.62}$$

4.5 Internal energy

An expression for the total internal energy of the system of SPH particles can be obtained by simply integrating the identity

$$\frac{d\pi}{d\rho} = \frac{P}{\rho^2} \,. \tag{4.63}$$

This relates the internal energy (per unit mass) with the pressure under reversible, adiabatic conditions where the pressure is given by the equation of state

$$P = P_0 \left(\left(\frac{\rho}{\rho_0} \right)^{\gamma} - 1 \right). \tag{4.64}$$

In this case the internal energy of a particle a is given by

$$\begin{aligned} \pi_a &= \int P_0 \left(\frac{\rho_a^{\gamma-2}}{\rho_0^{\gamma}} - \frac{1}{\rho_a^2} \right) d\rho \\ &= P_0 \left(\frac{\rho_a^{\gamma-1}}{(\gamma-1)\,\rho_0^{\gamma}} + \frac{1}{\rho_a} + K \right). \end{aligned} \tag{4.65}$$

Fixing the constant of integration such that $\pi(\rho_0) = 0$ gives

$$K = -\frac{1}{\rho_0} \left(\frac{\gamma}{\gamma - 1} \right) \tag{4.66}$$

and the total internal energy is found to be

$$\Pi_{\text{int}} = \sum_a m_a P_0 \left(\frac{1}{\rho_a} + \frac{1}{\rho_0 (\gamma - 1)} \left(\left(\frac{\rho_a}{\rho_0} \right)^{\gamma-1} - \gamma \right) \right). \tag{4.67}$$

4.6 Conservation properties of the variational formulation

To conclude this chapter the conservation properties of the variational formulation are presented. It will be shown that internal forces obtained from a potential function Π_{int} satisfy the necessary conditions for the preservation of linear and angular momentum [19, 84].

Linear momentum

It was shown in Section 3.6 that conservation of linear momentum is ensured by satisfying the following discrete condition

$$\sum_{a=1}^{N} \mathbf{T}_a = \mathbf{0}. \tag{4.68}$$

As long as the internal forces are derived from a potential function Π_{int} that is invariant with respect to rigid body translations its variation with respect to an arbitrary uniform velocity field $\mathbf{v}_0$ will vanish [3]. This condition is satisfied by the present formulation and consequently $D\Pi_{\text{int}}[\delta\mathbf{v}_0]$ satisfies

$$0 = D\Pi_{\text{int}}[\delta\mathbf{v}_0] = \left(\sum_{a=1}^{N} \mathbf{T}_a\right) \cdot \delta\mathbf{v}_0 \ . \tag{4.69}$$

Since this equations holds for arbitrary $\mathbf{v}_0$ equation (4.68) is satisfied and linear momentum will be preserved.

Angular momentum

It was shown in Section 3.6 that conservation of angular momentum is ensured by satisfying the following discrete condition

$$\sum_{a=1}^{N} \mathbf{x}_a \times \mathbf{T}_a = \mathbf{0}. \tag{4.70}$$

If the potential function Π_{int} is invariant with respect to rigid body rotations it will be shown that angular momentum is preserved [3].

Given an angular velocity vector $\delta\mathbf{w}$, the corresponding rigid body rotation is described by the set of velocities defined by $\delta\mathbf{v}_a = \delta\mathbf{w} \times \mathbf{x}_a$. In this case the variation of Π_{int} satisfies

$$0 = D\Pi_{\text{int}}\left[\delta\mathbf{w} \times \mathbf{x}_a\right] = \sum_{a=1}^{N} \mathbf{T}_a \cdot (\delta\mathbf{w} \times \mathbf{x}_a) = \delta\mathbf{w} \cdot \left(\sum_{a=1}^{N} \mathbf{x}_a \times \mathbf{T}_a\right). \tag{4.71}$$

Since this equations holds for arbitrary $\delta\mathbf{w}$ equation (4.70) is satisfied and angular momentum will be preserved.

It remains to check the invariance of Π_{int} with respect to rigid body rotations.

In such cases the stress introduced due to the rotation should vanish. For a Newtonian fluid with stress tensor $\boldsymbol{\sigma}(\mathbf{d})$ this implies that the discrete particle approximation of $\mathbf{d}$ should vanish.

Considering a rigid body rotation defined by $\mathbf{w} = [w_x, w_y, w_z]^{\mathrm{T}}$. Then the resulting velocity field is given by $\mathbf{v}(\mathbf{x}) = \mathbf{w} \times \mathbf{x}$ and the exact velocity gradient is given by

$$\nabla\mathbf{v} = \mathbf{W} = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix}. \tag{4.72}$$

In this case both $\mathbf{d} = \frac{1}{2}\left(\nabla\mathbf{v} + \nabla\mathbf{v}^{\mathrm{T}}\right) = \mathbf{0}$ and its trace $\nabla \cdot \mathbf{v} = 0$. Therefore, without discretization the potential function Π_{int} would be invariant with respect to rigid body rotation. However, in practice the velocity gradient is obtained via the SPH approximation

$$\begin{aligned} \nabla\mathbf{v}_a &= \sum_b V_b \left(\mathbf{W}\mathbf{x}_b - \mathbf{W}\mathbf{x}_a\right) \otimes \nabla w_b\left(\mathbf{x}_a, h_b\right) \\ &= \mathbf{W} \sum_b V_b \left(\mathbf{x}_b - \mathbf{x}_a\right) \otimes \nabla w_b\left(\mathbf{x}_a, h_b\right) \end{aligned} \tag{4.73}$$

and the correct velocity gradients will be obtained only if

$$\sum_b V_b \left(\mathbf{x}_b - \mathbf{x}_a\right) \otimes \nabla w_b\left(\mathbf{x}_a, h_b\right) = \mathbf{I} \qquad \text{for each particle } a. \tag{4.74}$$

This is precisely the linear gradient consistency condition derived in the previous chapter.

Using the linearly corrected gradient of the kernel function $\tilde{\nabla}\hat{w}_b\left(\mathbf{x}_a, h_b\right)$ ensures this condition is satisfied and the velocity gradient is obtained correctly as

$$\begin{aligned}\nabla\mathbf{v}_a &= \sum_b V_b \mathbf{W}\mathbf{x}_b \otimes \tilde{\nabla}\hat{w}_b\left(\mathbf{x}_a, h_b\right) \\ &= \mathbf{W}\left(\sum_b V_b \mathbf{x}_b \otimes \tilde{\nabla}\hat{w}_b\left(\mathbf{x}_a, h_b\right)\right) \\ &= \mathbf{W}. \end{aligned} \tag{4.75}$$

Consequently, when deviatoric stress tensors are implemented angular momentum will only be preserved if linear correction or the mixed kernel and gradient correction scheme is implemented.

4.7 Concluding remarks

In this chapter the governing equations for Newtonian fluids have been derived from variational principles. The conservation of linear and angular momentum has been established with the necessary kernel corrections.

The formulation does not presume particles possess identical masses or smoothing lengths. Consequently, it provides the ideal framework for the particle refinement algorithm and variable resolution simulations in later chapters. The corresponding expressions for the continuity equation and internal forces are found to take the same form as the traditional SPH equations derived in Chapter 2.

A boundary correction term has been included in the variational formulation which improves the density evaluation in the vicinity solid boundaries and introduces an additional boundary contact force to the derivation. Chapter 5 will present a simple and accurate method for the evaluation of this boundary force for general boundaries in two dimensions.

Chapter 5

Boundary methods

5.1 Introduction

Various different approaches have been used in the past to implement solid boundaries in SPH simulations. This chapter presents a new method for calculating boundary contact forces for any general boundary in two dimensions.

In the first part of the chapter the four most commonly used methods; the bounce back method, image particles, penalty methods and Lennard–Jones potentials are described. A variation of the Lennard–Jones potential that uses boundary particle averaging is presented. This new method avoids instabilities that traditionally occur when non-uniform boundary particle distributions are used.

In the remaining sections a simple and inexpensive method for exactly calculating the variational boundary contact force term derived in Chapter 4 is presented. The resulting boundary force for a single straight boundary is compared to the approximation previously used in the literature.

Finally, several applications using the variational boundary force are presented. The accuracy of the new formulation is compared to previous methods with a simple breaking dam example. A more complex boundary is used to model a simple flood defence problem and a water droplet falling into a curved bowl demonstrates the methods ability to model curved boundaries.

5.2 Summary of boundary implementations in SPH

5.2.1 Bounce back

The bounce back boundary implementation is conceptually the simplest method used to enforce boundary conditions within the SPH framework. Particles that are identified as having come into contact with a solid boundary are simply reflected back into the computational domain according to Newton's law of restitution (see Figure 5.1).

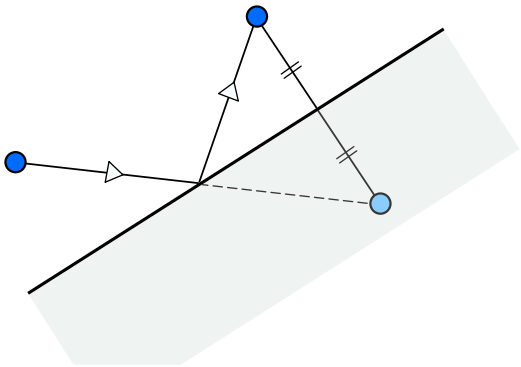


Figure 5.1: Bounce back method

If the interaction between the boundary and the particle is assumed to be perfectly elastic (corresponding to a coefficient of restitution equal to one) then the linear momentum of the system will be conserved.

5.2.2 Image particles

The image (or ghost) particle method models a solid boundary by modifying the equilibrium and continuity equations for particles in its vicinity by introducing artificial particles positioned behind the boundary as shown in Figure 5.2. This method helps eliminate the problem of poor interpolation of physical quantities at points near the boundary by increasing the number of particles that appear in the SPH approximations.

Image particles are not stored or evolved during the simulation. They are only used to add extra terms into the governing equations. The image particles are generated by reflecting a particle's position across the boundary as shown in Figure 5.2. In this

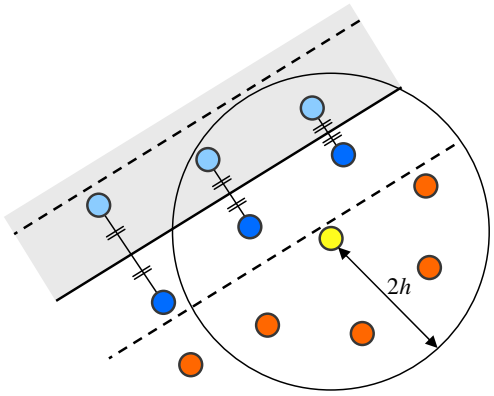


Figure 5.2: Image particle generation in the vicinity of a straight boundary

case the image particles have the same volume, density and pressure as the original particle but with opposite velocity component normal to the boundary. When a particle is in the vicinity of a corner region then three image particles are needed. Two being generated from the reflection of the particle across each line segment and the third coming from the reflection over the corner point.

Introducing image particles into discrete SPH equation (2.15) of a function f gives

$$f(\mathbf{x}) = \sum_{b \in M_{\mathbf{x}}} V_b f(\mathbf{x}_b) w_b(\mathbf{x}) + \sum_{b' \in M'_{\mathbf{x}}} V_{b'} f(\mathbf{x}_{b'}) w_{b'}(\mathbf{x}). \tag{5.1}$$

Here $M'_{\mathbf{x}}$ is the set of neighbouring image particles that contribute to the summation. In this way the corresponding governing equations which incorporate the image particles are easily derived.

The image particle method can be computationally expensive (especially in three dimensions) due to the extra terms in the governing equations but it does improve the interpolation on the boundary without needing to use the corrected kernel method, slip and non-slip boundaries can also be easily simulated.

5.2.3 Penalty methods

Meshless methods typically encounter difficulties in imposing essential boundary conditions. The problem is largely due to the fact that the approximations generated by reproducing kernel or moving least square methods do not interpolate the independent variables at particle points.

For example, when an independent variable such as the velocity $\mathbf{v}_B$ is specified at

a boundary particle B the SPH velocity field in general will result in the inequality $\mathbf{v}_h(\mathbf{x}_B) \neq \mathbf{v}_B$. By introducing a simple penalty treatment on the boundary particles essential boundary conditions can be easily implemented in SPH [13, 67].

With kernel correction the desired boundary condition is given by

$$\mathbf{v}_h(\mathbf{x}_B) = \sum_{b \in M_B} V_b \mathbf{v}_b \hat{w}_b(\mathbf{x}_B) = \mathbf{v}_B. \tag{5.2}$$

This condition can be enforced by a penalty boundary potential defined by

$$\begin{aligned} \Pi_{\text{bp}} &= \frac{1}{2}\kappa_p \sum_{B=1}^{N_{\text{bp}}^B} A_B (\mathbf{v}_B - \mathbf{v}_h(\mathbf{x}_B)) \cdot (\mathbf{v}_B - \mathbf{v}_h(\mathbf{x}_B)) \\ &= \frac{1}{2}\kappa_p \sum_{B=1}^{N_{\text{bp}}^B} A_B \left(\mathbf{v}_B - \sum_{b \in M_B} V_b \mathbf{v}_b \hat{w}_b(\mathbf{x}_B)\right) \cdot \left(\mathbf{v}_B - \sum_{b \in M_B} V_b \mathbf{v}_b \hat{w}_b(\mathbf{x}_B)\right) \end{aligned} \tag{5.3}$$

where A_B is the surface area associated to boundary particle B, κ_p is a penalty value and N_{bp}^B are the relevant boundary points.

Differentiating Π_{bp} with respect to particle velocities gives

$$\mathbf{T}_a^{\text{bp}} = \frac{\partial \Pi_{\text{bp}}}{\partial \mathbf{v}_a} = \sum_{B \in M_a^B} \kappa_p A_B (\mathbf{v}_B - \mathbf{v}_h(\mathbf{x}_B)) V_a \hat{w}_a(\mathbf{x}_B). \tag{5.4}$$

By adding this boundary potential force to the internal force as given by equation (4.7) the total force incorporating the essential boundary condition is obtained as

$$\mathbf{T}_a^{\text{Tot}} = \mathbf{T}_a + \mathbf{T}_a^{\text{bp}}. \tag{5.5}$$

5.2.4 Lennard–Jones potential

The Lennard–Jones boundary implementation uses boundary particles to exert short range repulsive forces on surrounding fluid particles to model solid boundaries. The expression for these radial forces are given by a Lennard–Jones potential based on the force found between interacting pairs of molecules [49]. A typical interaction between fluid and boundary particles can be seen in Figure 5.3.

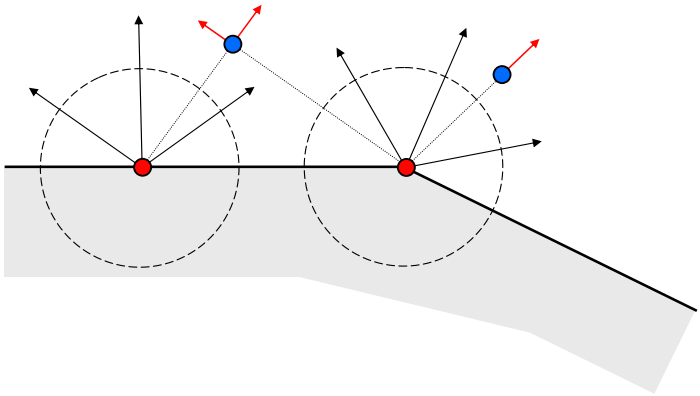


Figure 5.3: Lennard–Jones forces

For a particle a, distance r from a boundary particle b_{LJ} the force per unit mass exerted by the boundary particle is given by

$$\mathbf{T}^{\mathrm{LJ}}(r) = \begin{cases} K\left[\left(\frac{r_0}{r}\right)^{n_1} - \left(\frac{r_0}{r}\right)^{n_2}\right]\frac{\mathbf{r}}{r^2} & \text{if } r \leq r_0 \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

where n_1 and n_2 are integer parameters commonly taken as $n_1 = 4$ and $n_2 = 2$ (or $n_1 = 12$ & $n_2 = 4$), and $\mathbf{r} = \mathbf{x}_a - \mathbf{x}_{b_{\mathrm{LJ}}}$.

The value of r_0 defines the cutoff distance for the Lennard–Jones force. When $r > r_0$ the force is set to zero since the Lennard–Jones potential becomes negative in this region. This ensures that the boundary force remains purely repulsive. It is important to set the value of r_0 carefully. If it is too large then too many particles will feel the influence of the boundary particles in the initial configuration. If it is too small then particles will be able to artificially penetrate the boundary.

The coefficient K has units velocity–squared and governs the strength of the boundary force. This should be chosen in proportion to the maximum internal energy of the problem. For example in a simulation where particles are under gravity, and the maximum particle height is given as $H_{\max}$ then K should be chosen $K \approx 5gH_{\max}$.

Smoothed Lennard–Jones potential

The above method is reliant on a uniform distribution of boundary particles. Grouping many boundary particles in close proximity to each other will result in large repulsive boundary forces in that region. Resulting in local instabilities and inaccuracies. However, in many real–world applications complex boundary geometries

need to be modelled and it is useful to be able to arbitrarily place boundary particles to obtain the detail required.

A solution which allows for variable boundary particle spacing is motivated by the corrections implemented in the corrected SPH method discussed in Chapter 3 to develop a Smoothed Lennard–Jones potential.

The total Lennard-Jones boundary force for a given particle a can be summarised as the sum of all the interactions with neighbouring boundary particles. Now suppose that the Lennard–Jones potential function is simplified to be a function of the normal distance from the particle to a boundary particle (see Figure 5.4)

$$\mathbf{T}_a^{\mathrm{LJ}} = \sum_{\mathrm{bp}} f\left(r_{\mathrm{bp}}^{\mathbf{n}}\right) \mathbf{n}_{\mathrm{bp}}. \tag{5.7}$$

In this formulation all neighbouring boundary particles will still contribute with equal weight to the total boundary force. It is this fact that prevents the use of non-uniformly distributed boundary particles.

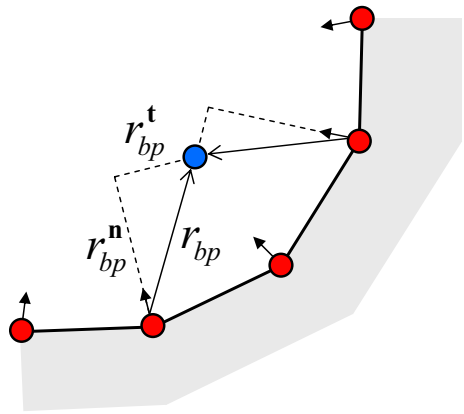


Figure 5.4: Components of Lennard–Jones force

The smoothed version of the above equation is given by

$$\mathbf{T}_a^{\mathrm{LJ}} = \sum_{\mathrm{bp}} f\left(r_{\mathrm{bp}}^{\mathbf{n}}\right) \mathbf{n}_{\mathrm{bp}} V_{\mathrm{bp}} \hat{w}_{\mathrm{bp}}\left(\mathbf{x}_a, h_{\mathrm{bp}}\right) \tag{5.8}$$

where $\hat{w}$ is the corrected kernel over the boundary particles only

$$\hat{w}\left(\mathbf{x}\right) = \alpha\left(\mathbf{x}\right) w\left(\mathbf{x}\right) \qquad \text{where} \qquad \alpha\left(\mathbf{x}\right) = \frac{1}{\sum\limits_{\mathrm{bp}} V_{\mathrm{bp}} w_{\mathrm{bp}}\left(\mathbf{x}\right)} \ . \tag{5.9}$$

In this formulation more weight is added to boundary particles that are closer to the fluid particle a. Moreover, consider the case in two dimensions where boundary particles are distributed on a straight boundary. In this case all the $f\left(r_{\mathrm{bp}}^{\mathbf{n}}\right)$ terms are equal for all boundary particles (see Figure 5.5) and since the corrected kernel is used the total boundary force expression simplifies to

$$\begin{aligned} \mathbf{T}_a^{\mathrm{LJ}} &= f\left(r_{\mathrm{bp}}^{\mathbf{n}}\right)\mathbf{n}_{\mathrm{bp}}\left(\sum_{\mathrm{bp}} V_{\mathrm{bp}}\hat{w}_{\mathrm{bp}}\left(\mathbf{x}_a, h_{\mathrm{bp}}\right)\right) \\ &= f\left(r_{\mathrm{bp}}^{\mathbf{n}}\right)\mathbf{n}_{\mathrm{bp}}. \end{aligned} \tag{5.10}$$

So for straight boundaries the smoothed Lennard–Jones formulation ignores the distribution of boundary particles and is equivalent to a single boundary particle acting on the fluid particle at a distance $r_{\mathbf{n}}$.

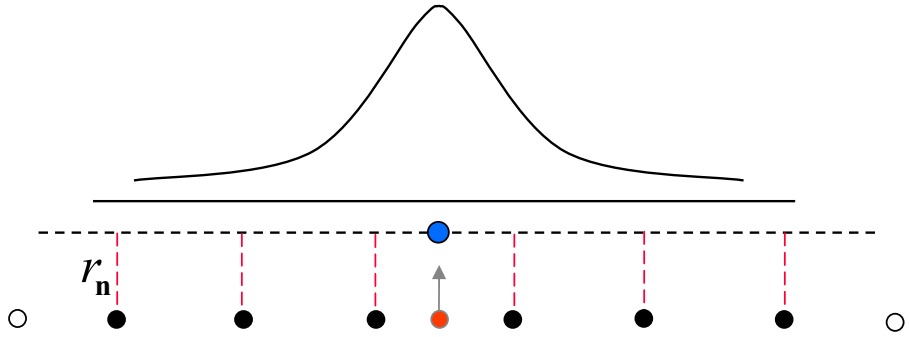


Figure 5.5: Smoothed Lennard–Jones on a straight boundary

Examples using the smoothed Lennard–Jones boundary particle implementation

A simple test consisting of a small collapsing square of fluid was run as a test example for the smoothed Lennard–Jones boundary implementation. The flat boundary is defined by a number of stationary boundary particles. On the lefthand side of the boundary there is a region of closely spaced boundary particles, as shown in Figure 5.6.

The results after a few timesteps are shown in Figure 5.7. Without smoothing the dense configuration of boundary particles produces an excessively large contact force and the resulting simulation is unstable. However, with the smoothing applied the dense configuration is smoothed out as expected and the simulation is stable.

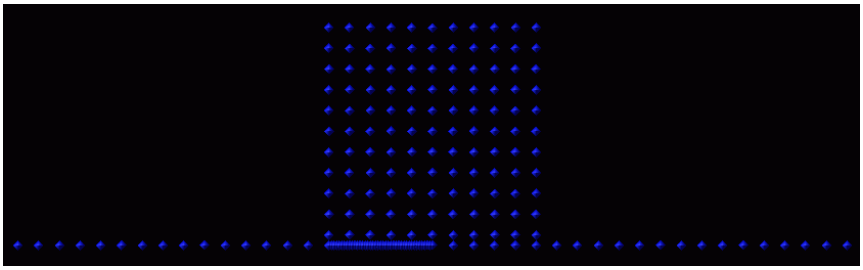


Figure 5.6: Initial test configuration

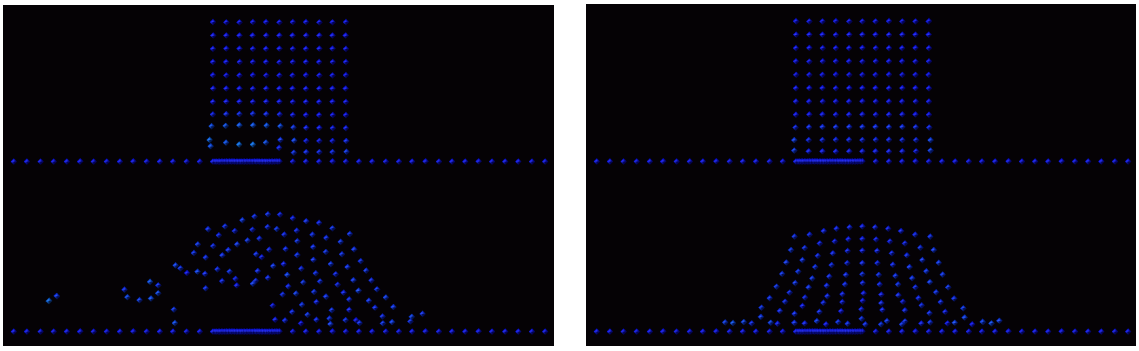


Figure 5.7: Results before/after smoothed Lennard-Jones implementation

Finally, the smoothed Lennard–Jones boundary implementation is tested on a larger problem with a curved boundary and irregular boundary particle distribution as shown in Figure 5.8. In this case there are four times as many boundary particles making up the righthand side of the boundary than on the left. However, this does not effect the stability of the system and the simulation remains perfectly symmetric.

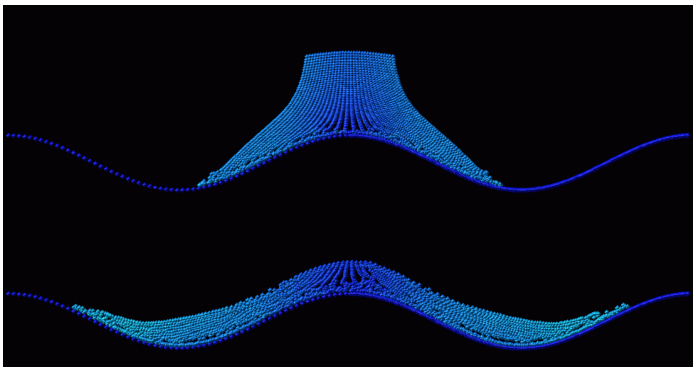


Figure 5.8: Example of a non–regular boundary using the smoothed Lennard-Jones implementation (coarse boundary particle distribution on left/fine distribution on right)

5.3 A variational boundary contact force

The remainder of this chapter is concerned with the calculation and implementation of the variational boundary contact force term in two dimensions which was derived in Chapter 4 and is given by

$$\mathbf{T}_a^B = -\frac{m_a P_a}{\rho_a \gamma_a} \nabla \gamma_a \tag{5.11}$$

where m_a is the mass, P_a the pressure and ρ_a the density of the fluid particle. The γ_a and $\nabla \gamma_a$ terms come from the additional correction term introduced into the variational formulation for the density (see Figure 5.9)

$$\rho\left(\mathbf{x}_a\right) = \frac{\sum_b m_b w_b\left(\mathbf{x}_a, h_b\right)}{\gamma\left(\mathbf{x}_a, h_a\right)} \qquad \text{where} \qquad \gamma\left(\mathbf{x}_a, h_a\right) := \int w_a\left(\mathbf{x}, h_a\right) d\mathbf{x}. \tag{5.12}$$

In order to calculate $\mathbf{T}_a^B$ it is necessary to evaluate γ_a and $\nabla \gamma_a$ for each particle at every timestep. To calculate γ directly using equation (5.12) would be too computationally expensive for large dynamic simulations or for problems with complex boundaries.

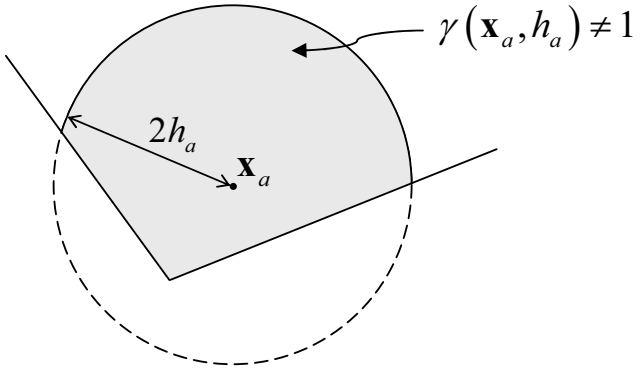


Figure 5.9: γ function in the vicinity of a general boundary

5.3.1 Approximate boundary force evaluation in 2D

In the paper by Kulasegaram [69] the gamma function for a single straight boundary in two dimensions was calculated numerically by splitting the contributing region into a finite number of points p_i each with associated area A_i as shown in Figure 5.10.

Then γ_a is approximated by

$$\gamma_a \approx \sum_i A_i w_a\left(p_i, h_a\right). \tag{5.13}$$

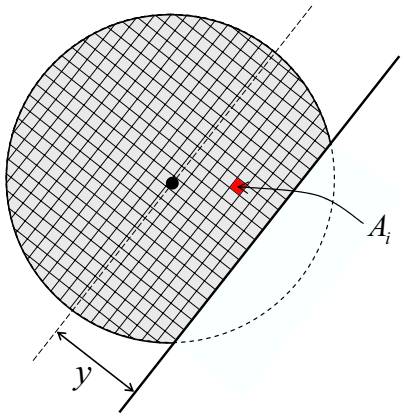


Figure 5.10: Discrete approximation to γ in the vicinity of a single boundary

By calculating these approximations to γ at various normal distances to the boundary and then fitting this data using curve fitting methods γ and γ' were approximated by simple polynomial expressions

$$\begin{aligned} \gamma(\varepsilon) &= 1 + (0.0625 - 0.0531\varepsilon)(\varepsilon - 2)^3 \\ \gamma'(\varepsilon) &= (0.2937 - 0.2124\varepsilon)(\varepsilon - 2)^2 \end{aligned} \qquad \text{where } \varepsilon = \frac{y}{h} \,. \tag{5.14}$$

Given the normal distance of a particle from the boundary and using the above equations to calculate values for γ and γ' the boundary contact force in the vicinity of a single straight boundary can be calculated using equation (4.31) as

$$\mathbf{T}_a^B = -m_a \frac{P_a \gamma_a'}{\rho_a \gamma_a h_a} \mathbf{n}_B \tag{5.15}$$

where $\mathbf{n}_B$ is the unit inward pointing normal to the boundary. This is a very simple and accurate way to deal with single straight boundaries in two dimensions.

Approximating corner regions

However, a method for calculating γ in the vicinity of corner regions like the one shown in Figure 5.9 is required for the majority of practical engineering problems. Two approximate methods are presented below.

Method 1

Consider a particle in the vicinity of a corner region as shown in Figure 5.11. Then the exact value of γ is due to the presence of both boundary segments Γ_1 and Γ_2.

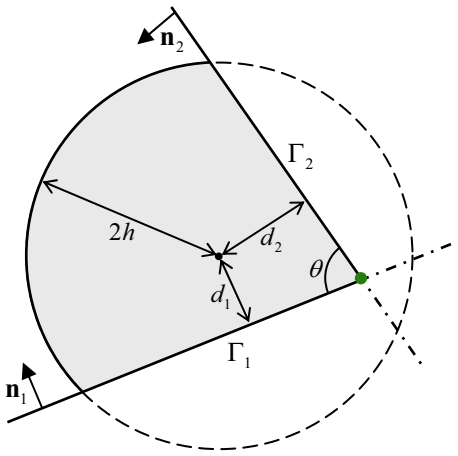


Figure 5.11: Method I: Approximation of γ in 2D

The first method of approximation is to use a bilinear combination of the correction factors from each boundary

$$\gamma = c_1 + c_2\gamma_1 + c_3\gamma_2 + c_4\gamma_1\gamma_2 \tag{5.16}$$

where γ_1 and γ_2 are the correction factors coming from boundary segments Γ_1 and Γ_2 respectively. The constant coefficients c_1, c_2, c_3 and c_4 are to be calculated for a given corner region by enforcing the known values of γ. These are given as

$$\begin{array}{llllll} \gamma = 1.0 & \text{when } d_1 > 2h & \text{and} & d_2 > 2h & (\gamma_1 = 1.0,\ \gamma_2 = 1.0) \\ \gamma = 0.5 & \text{when } d_1 = 0 & \text{and} & d_2 > 2h & (\gamma_1 = 0.5,\ \gamma_2 = 1.0) \\ \gamma = 0.5 & \text{when } d_1 > 2h & \text{and} & d_2 = 0 & (\gamma_1 = 1.0,\ \gamma_2 = 0.5) \\ \gamma = \theta/2\pi & \text{when } d_1 = 0 & \text{and} & d_2 = 0 & (\gamma_1 = 0.5,\ \gamma_2 = 0.5) \end{array} \tag{5.17}$$

where d_1 and d_2 are the distances from the particle to each boundary respectively, and θ is the interior angle at the corner.

With this bilinear form for γ its derivative is then obtained as

$$D\gamma\left[\delta\mathbf{v}\right] = \left(c_2 + c_4\gamma_2\right) D\gamma_1\left[\delta\mathbf{v}\right] + \left(c_3 + c_4\gamma_1\right) D\gamma_2\left[\delta\mathbf{v}\right]. \tag{5.18}$$

By using equation (4.31) and rearranging gives

$$\begin{gathered} D\gamma\left[\delta\mathbf{v}\right] = \frac{1}{h}\mathbf{N}\cdot\delta\mathbf{v}, \\ \text{where} \quad \mathbf{N} = \left(c_2 + c_4\gamma_2\right)\gamma_1'\mathbf{n}_1 + \left(c_3 + c_4\gamma_1\right)\gamma_2'\mathbf{n}_2\ . \end{gathered} \tag{5.19}$$

and the boundary contact force is given by

$$\mathbf{T}_a^B = -m_a \frac{P_a}{\rho_a \gamma_a h_a} \mathbf{N}_a. \tag{5.20}$$

Method 2

The second method of approximation is to replace the boundary intersection with a single representative boundary that can be simply handled using equation (5.14) and equation (5.15).

Figure 5.12 shows a particle in the vicinity of convex and concave corner sections. The points P_1 and P_2 are the points of intersection of the boundary of the kernel support with each line segment. These points are easily calculated by analytic geometry.

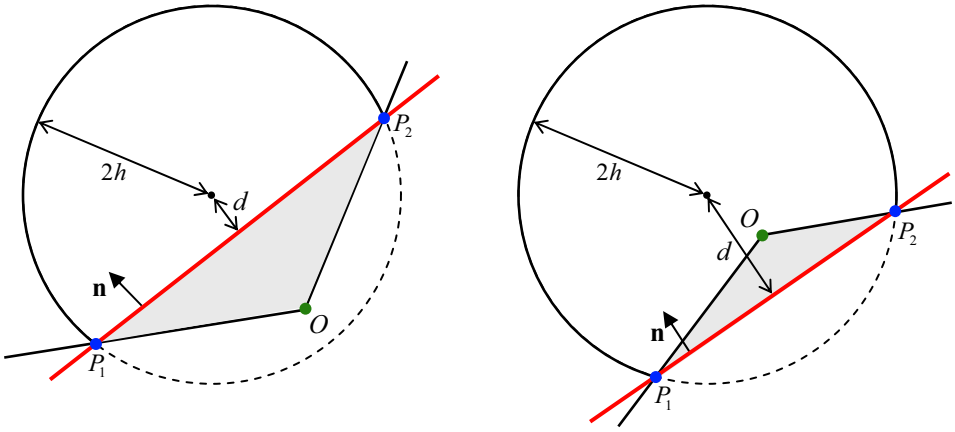


Figure 5.12: Method II: Approximation of γ in 2D

The replacement line segment is then identified as the line connecting these two points. Using the particle distance d from the single boundary the correction factor γ and the resulting boundary contact force is calculated.

It can be seen from Figure 5.12 that by using the single boundary the area of integration is not correctly represented. In the case of a convex corner γ is under evaluated by the shaded area P_1OP_2 while for a concave corner γ is over evaluated by the shaded area P_1OP_2.

In practical applications these discrepancies will be small since the size of the kernel support is comparatively small compared to that of the boundary. Since this approach is conceptually and computationally simpler to implement it is an attractive method for dealing with corner regions.

5.3.2 Exact boundary force evaluation in 2D

In this section a simple and inexpensive method for evaluating γ terms exactly for any boundary in two dimensions is presented.

With this aim in mind a vector function $\mathbf{W}$ is sought such that

$$\nabla \cdot \mathbf{W} = w \quad \text{where} \quad \mathbf{W} = f(r) \begin{bmatrix} x \\ y \end{bmatrix} \quad \text{and} \quad r = \frac{\|\mathbf{x}\|}{h} \,. \tag{5.21}$$

The scalar function f is to be found and depends on the kernel function used.

Using the Divergence Theorem, the integral of the kernel function in equation (5.12) which defines γ is transformed into the integral of $\mathbf{W} \cdot \mathbf{n}$ over the boundary ∂B

$$\gamma(\mathbf{x}_a, h_a) := \int_B w_a(\mathbf{x}, h_a)\, d\mathbf{x} = \int_B \nabla \cdot \mathbf{W}\; d\mathbf{x} = \int_{\partial B} \mathbf{W} \cdot \mathbf{n}\; ds \tag{5.22}$$

where $\mathbf{n}$ is the outward normal of the boundary of the kernel support ∂B as shown in Figure 5.13.

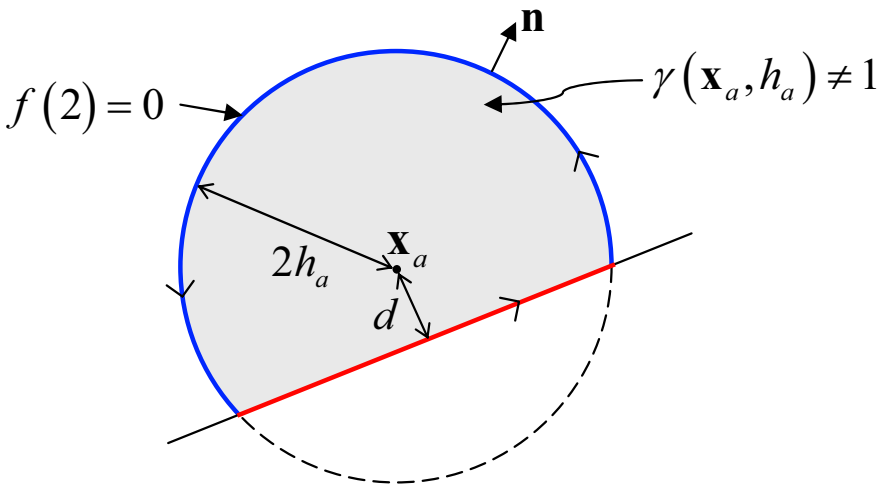


Figure 5.13: Calculating γ by applying the Divergence Theorem

The gamma function has now been rewritten in terms of the boundary of the kernel support that lies inside the problem domain rather than the area of the kernel inside the domain. In two dimensions it is now easy to calculate these line integrals and so to calculate γ exactly for any particle in the vicinity of any boundary.

From the definition of $\mathbf{W}$ the function f satisfies the following differential equation

$$\begin{aligned} w = \nabla \cdot \mathbf{W} &= (\nabla \cdot \mathbf{x})\, f + \nabla f \cdot \mathbf{x} \\ &= 2f + (\nabla r f') \cdot \mathbf{x} \\ &= 2f + \frac{1}{r}(\mathbf{x} \cdot \mathbf{x})\, f' \\ &= 2f + r f'. \end{aligned} \tag{5.23}$$

Multiplying both sides by r, the function f is obtained as the general solution

$$f = \frac{1}{r^2} \int r w \; dr. \tag{5.24}$$

The choice of kernel function w appearing in the derivation above is arbitrary. However, for the remainder of the section the quintic kernel in two dimensions will be used.

$$w(r,h) = \frac{3}{16h^2\pi} \begin{cases} \left(2 - \frac{r}{h}\right)^5 - 16\left(1 - \frac{r}{h}\right)^5 & 0 \leqslant r \leqslant h \\ \left(2 - \frac{r}{h}\right)^5 & h < r \leqslant 2h \\ 0 & r > 2h \end{cases} . \tag{5.25}$$

After some simple algebra the required function $\mathbf{W}$ is obtained as

$$\mathbf{W} = \frac{3}{16h^2\pi} f(r) \begin{bmatrix} x \\ y \end{bmatrix} \quad \text{where}$$

$$f(r) = \begin{cases} 8 - 20r^2 + 24r^3 - (35/3)\, r^4 + (15/7)\, r^5 & \\ \qquad + C_1 (1/r^2) & \text{if } \; 0 < r < 1 \\ 16 - (80/3)\, r + 20r^2 - 8r^3 + (5/3)\, r^4 - (1/7)\, r^5 & \\ \qquad + C_2 (1/r^2) & \text{if } \; 1 \leqslant r \leqslant 2 \end{cases} \tag{5.26}$$

In this case the resulting general solution contains two arbitrary constants of integration C_1 and C_2. All possible choices for these constants will result in a solution function f that satisfies the differential equation and consequently $\nabla \cdot \mathbf{W} = w$. However, these constants should be chosen such that the resulting function f is sufficiently smooth in order that the Divergence Theorem may be applied.

The following computations are simplified greatly by choosing the solution that satisfies $f(2) = 0$. With this choice $\mathbf{W}$ will vanish on the boundary of the kernel support and only the region of the boundary inside will contribute to the calculation of γ. This is shown in Figure 5.13 where only the red region of the boundary contributes to γ; the blue region is zero and so can be ignored in all computations.

With this in mind the constant C_2 is chosen such that f vanishes on the boundary of the kernel and the constant C_1 is chosen to enforce continuity of f at $r = 1$. As such f is given by

$$f(r) = \begin{cases} 8 - 20r^2 + 24r^3 - (35/3)\, r^4 + (15/7)\, r^5 \\ \qquad -56/21\, (1/r^2) \\ 16 - (80/3)\, r + 20r^2 - 8r^3 + (5/3)\, r^4 - (1/7)\, r^5 \\ \qquad -64/21\, (1/r^2) \end{cases} \begin{array}{l} \text{if } \ 0 < r < 1 \\ \\ \text{if } \ 1 \leqslant r \leqslant 2 \end{array} \tag{5.27}$$

With this choice for the constants equation (5.27) is found to be discontinuous at $r = 0$ and so the Divergence Theorem cannot be applied directly. However, this singularity can be removed from inside the domain by placing a small ball of radius ϵh about the origin of the kernel function as shown in Figure 5.14.

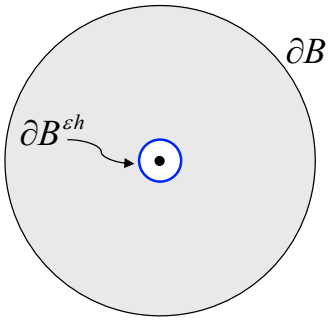


Figure 5.14: Removal of singularity

Therefore γ is given by

$$\gamma = \lim_{\epsilon h \to 0} \int_{\partial B^{\epsilon h}} \mathbf{W} \cdot \mathbf{n}\, ds + \int_{\partial B} \mathbf{W} \cdot \mathbf{n}\, ds. \tag{5.28}$$

When a particle is sufficiently far from a boundary γ should be equal to one. Since the second term in the above equation will vanish since $f(2) = 0$ it remains to check that the first term converges to unity as $\epsilon h \to 0$.

$$\begin{aligned}\gamma = \int\limits_{\partial B^{\epsilon h}} \mathbf{W}\cdot\mathbf{n}\; ds &= \frac{3}{16h^2\pi} f(\epsilon) \int\limits_{t=0}^{t=2\pi} \begin{pmatrix} \epsilon h\cos(t) \\ \epsilon h\sin(t) \end{pmatrix} \cdot \begin{pmatrix} \cos(t) \\ \sin(t) \end{pmatrix} \epsilon h\; dt \\ &= \frac{3}{16h^2\pi} (\epsilon h)^2 f(\epsilon) 2\pi \\ &= \frac{6}{16}\left[\epsilon^2 f(\epsilon)\right] \end{aligned} \tag{5.29}$$

Since

$$\lim_{\epsilon\to 0}\left[\epsilon^2 f(\epsilon)\right] = \frac{56}{21} \tag{5.30}$$

we have as expected

$$\gamma = \frac{6}{16}\lim_{\epsilon\to 0}\left[\varepsilon^2 f(\epsilon)\right] = \frac{6}{16}\,\frac{56}{21} = 1. \tag{5.31}$$

Therefore, the final expression for γ in the vicinity of a general boundary in two dimensions using the quintic kernel function is given by

$$\gamma(\mathbf{x}_a, h_a) = 1 + \int\limits_{\partial V} \mathbf{W}\cdot\mathbf{n}\; ds. \tag{5.32}$$

Calculating $\nabla\gamma$

It is also necessary to obtain an expression for the gradient of the gamma function in the vicinity of a boundary.

Consider the variation of $\nabla\gamma$ with respect to a small perturbation $\mathbf{dx}$ from its initial position $\mathbf{x}$ as shown in Figure 5.15. Given this small change in position of the centre of the kernel function the change in γ is given by

$$D\gamma[\mathbf{dx}] = \frac{\partial\gamma}{\partial x}\,dx + \frac{\partial\gamma}{\partial y}\,dy = \nabla\gamma\cdot\mathbf{dx} \tag{5.33}$$

where

$$\nabla\gamma = \frac{\partial\gamma}{\partial\mathbf{x}} = \begin{pmatrix} \partial\gamma/\partial x \\ \partial\gamma/\partial y \end{pmatrix}. \tag{5.34}$$

Assuming the perturbation $\mathbf{dx}$ is small, the change in γ is given by

$$D\gamma[\mathbf{dx}] = -\int\limits_{l} w(s)(\mathbf{dx}\cdot\mathbf{n})\, ds = \mathbf{dx}\cdot\left(-\int\limits_{l} w(s)\,\mathbf{n}\; ds\right). \tag{5.35}$$

where $\mathbf{n}$ is the outward normal to the boundary.

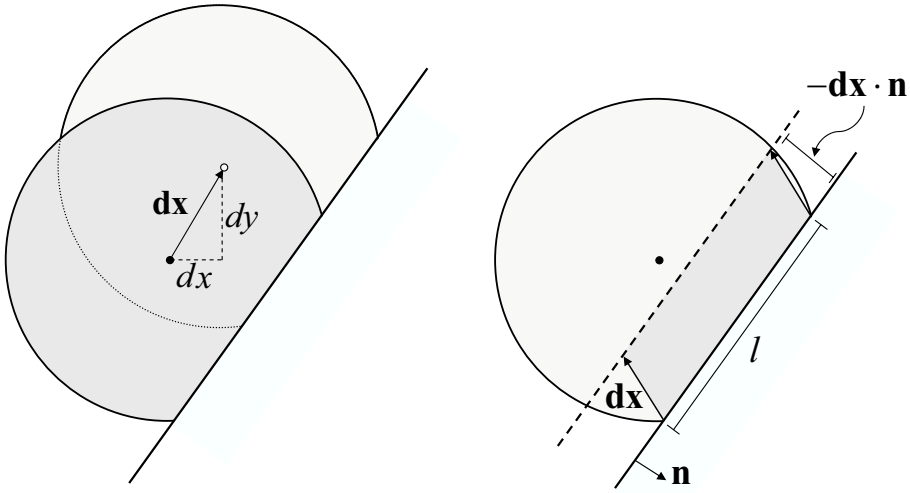


Figure 5.15: Evaluating $\nabla\gamma$

Equating equations (5.33) and (5.35) the gradient of γ is given by the integral

$$\nabla\gamma = -\mathbf{n}\int_{l} w\left(s\right) ds. \tag{5.36}$$

Evaluating γ and $\nabla\gamma$ for single straight boundaries

Using the preceding theory γ and $\nabla\gamma$ can now be evaluated in the vicinity of a single straight line boundary at a distance d from the centre of the kernel. Since the kernel function is symmetric the orientation of the boundary is unimportant and the boundary is assumed to be parallel to the y–axis as shown in Figure 5.16.

$$\gamma = 1 + \frac{3}{16h^2\pi}\int_{\Gamma} f\left(s\right)\begin{pmatrix} x \\ y \end{pmatrix}\cdot\begin{pmatrix} 1 \\ 0 \end{pmatrix} ds = 1 + \frac{3d}{16h^2\pi}\int_{\Gamma} f\left(s\right) ds \tag{5.37}$$

Since the distance in the x–direction is fixed the above integral can be written in terms of y only. Where $s\left(y\right) = \frac{\sqrt{d^2+y^2}}{h}$ and

$$\gamma = 1 + \left(2 \times \frac{3d}{16h^2\pi}\int_{y=0}^{y_{\max}} f\left(s\left(y\right)\right) dy\right). \tag{5.38}$$

Integrating f above with respect to y over Γ gives a closed form for γ in the vicinity of a single straight boundary, written in terms of the primitive function $F\left(x, y\right)$

$$\gamma = 1 + \frac{3d}{8h^2\pi}\left(F\left(d, y_{\max}\right) - F\left(d, 0\right)\right). \tag{5.39}$$

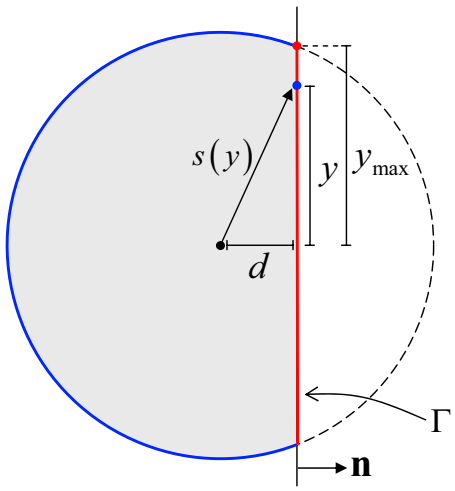


Figure 5.16: Calculating γ and $\nabla\gamma$ for a single straight boundary

Similarly, a closed form for $\nabla\gamma$ in the vicinity of a single straight boundary is calculated by integrating equation (5.36) over Γ

$$\nabla\gamma = -2\mathbf{n}\int_{y=0}^{y_{\max}} w\left(s\left(y\right)\right)dy = -2\mathbf{n}\left(G\left(d, y_{\max}\right) - G\left(d, 0\right)\right). \tag{5.40}$$

The primitive functions $F\left(x, y\right)$ and $G\left(x, y\right)$ for the quintic kernel are given in full in Appendix B.

Therefore, given the spherical symmetry of the kernel function and knowing the minimum distance from the centre of the kernel to the boundary both γ and $\nabla\gamma$ can be calculated exactly.

The values of γ and $\nabla\gamma$ for a single straight line boundary using the primitive functions derived above are compared to the approximations of Kulasegaram [69] in Figure 5.17 and Figure 5.18.

It can be seen that the approximate γ function was in good agreement with the exact γ function. However, the approximate $\nabla\gamma$ function was a less accurate representation of the exact $\nabla\gamma$ function.

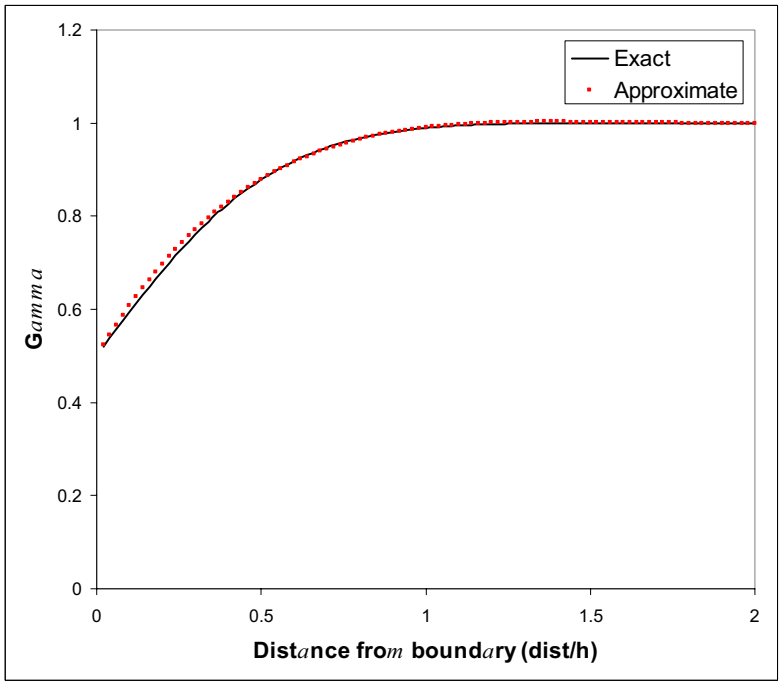


Figure 5.17: Graph of γ in the vicinity of a single straight boundary

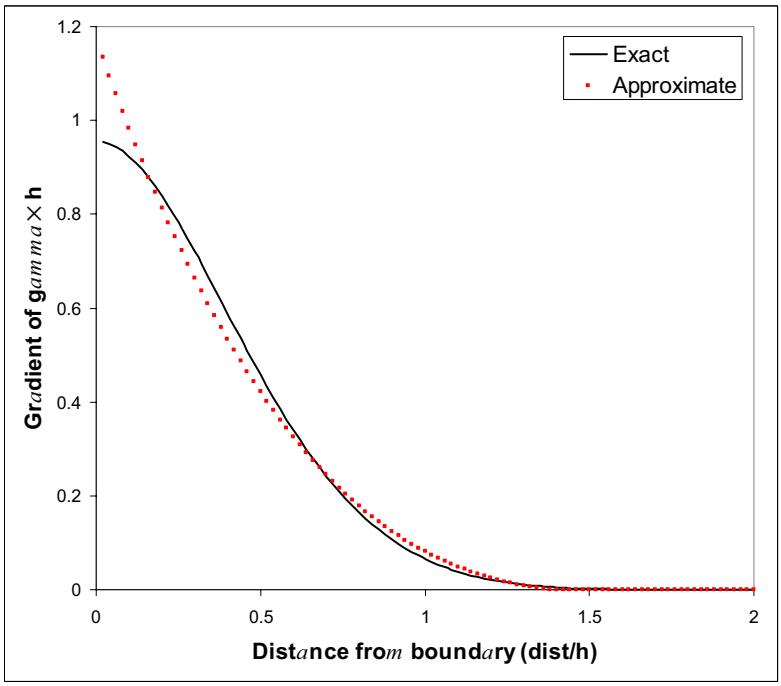


Figure 5.18: Graph of $\nabla\gamma$ in the vicinity of a single straight boundary

Evaluating γ and $\nabla\gamma$ for corner regions

Corner regions are simply handled by applying the theory developed for single straight line boundaries to a general boundary intersection like the corner region shown in Figure 5.19.

Contributions from each line segment Γ_1 and Γ_2 are calculated using the particle normal distances d_1 and d_2 from each boundary segment. These are then summed to obtained the exact values for γ and $\nabla\gamma$ for any given boundary in two dimensions.

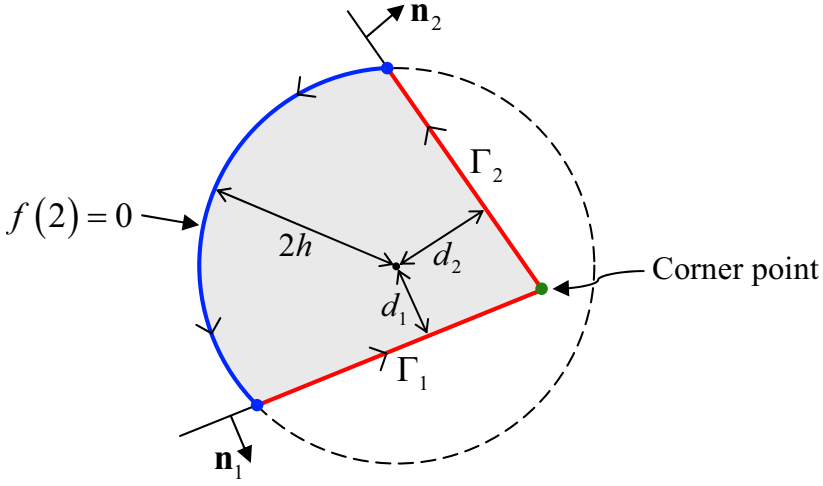


Figure 5.19: Calculating γ and $\nabla\gamma$ in the vicinity of a corner

$$\gamma = 1 + \int_{\Gamma_1 \cup \Gamma_2} \mathbf{W} \cdot \mathbf{n} \, ds \qquad \nabla\gamma = - \int_{\Gamma_1 \cup \Gamma_2} w\mathbf{n} \, ds. \tag{5.41}$$

Using equation (5.41) the boundary contact force for particles in the vicinity of a boundary can easily be calculated by

$$\mathbf{T}_a^B = -\frac{m_a P_a}{\rho_a \gamma_a} \nabla\gamma_a \; . \tag{5.42}$$

The computed values for γ and $\nabla\gamma$ in the vicinity of several example boundaries in two dimensions can be found in Figure 5.20. It can be seen that the γ value is correctly obtained at the corner points and the resulting gradients are smooth.

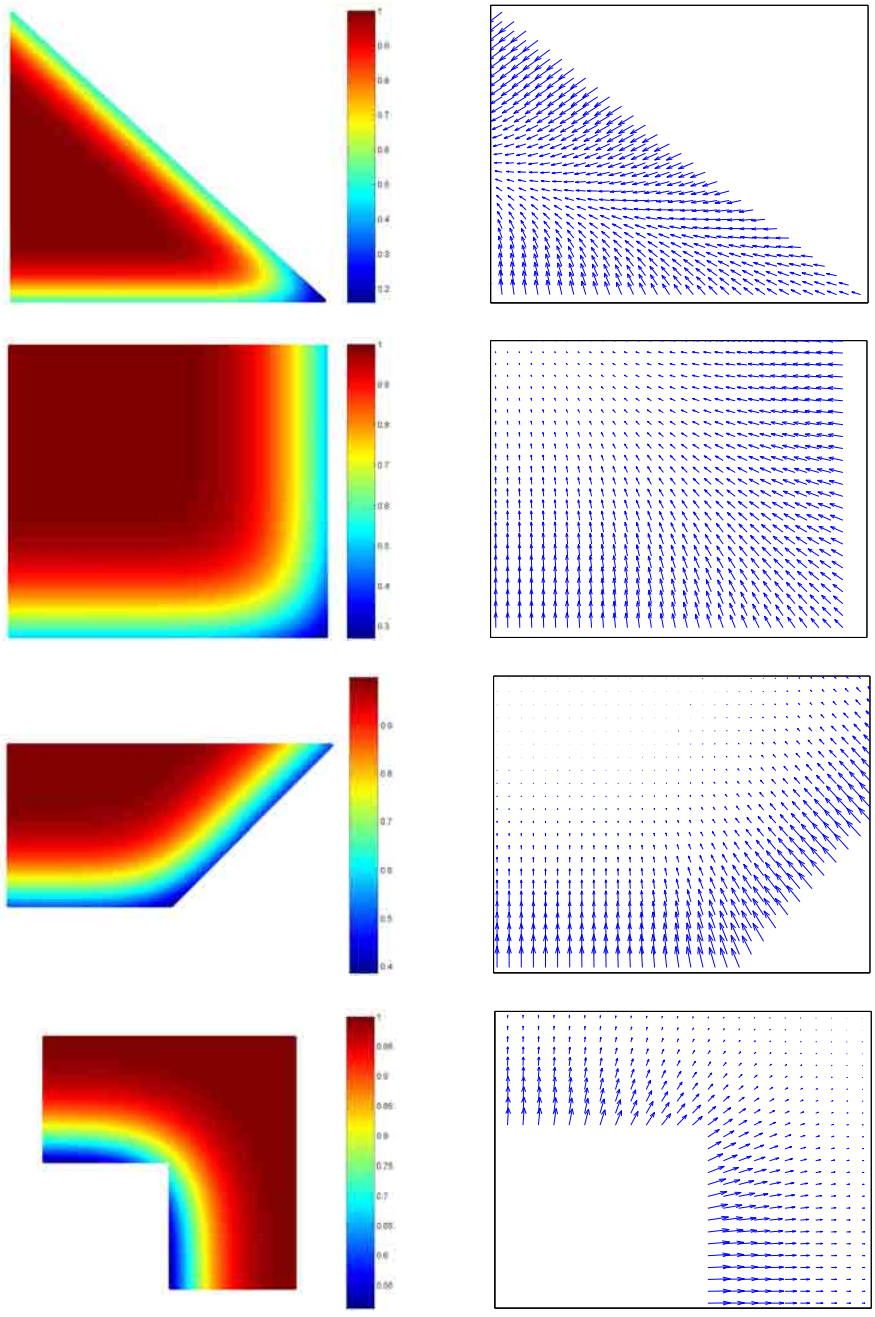


Figure 5.20: γ and $\nabla\gamma$ plotted in the vicinity of general boundaries in 2D

Evaluating γ and $\nabla\gamma$ for curved boundaries

It is a simple generalisation to apply the gamma boundary implementation to curved boundaries. Assuming that the curvature of the boundary is small in comparison to the smoothing length of the particle then the boundary can simply be approximated by a straight line segment. As shown in Figure 5.21 this is the line tangent to the closest point of the curve a distance d from the particle.

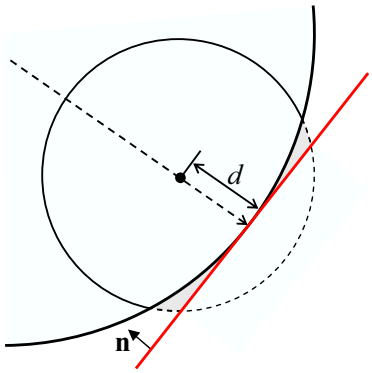


Figure 5.21: γ evaluation for curved boundary

When the curvature of the boundary is greater, modified approximations to γ and $\nabla\gamma$ can be calculated which take into account the curved boundary as discussed Section 5.3.1. The modified γ function for positive and negative values of curvature can be seen in Figure 5.22.

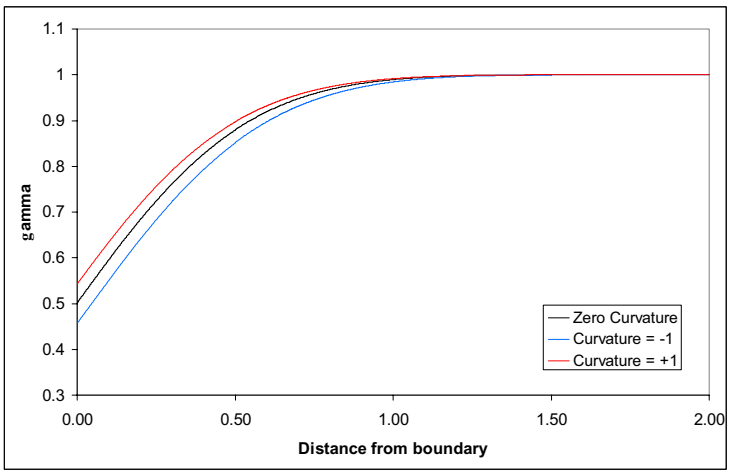


Figure 5.22: γ function for boundaries with varying curvature

5.3.3 Incompressibility issues

In the sections above a pressure based, variationally consistent, boundary contact force $\mathbf{T}_a^B = \mathbf{T}(P_a)$ has been derived that can be calculated exactly for general boundaries in two dimensions.

$$\mathbf{T}(P_a) = -\frac{m_a P_a}{\rho_a \gamma_a} \nabla \gamma_a \tag{5.43}$$

The vector term $\nabla \gamma_a$ is in the inward–normal direction to the boundary and the parameters m_a, ρ_a and γ_a are strictly positive valued.

In contrast, the pressure P_a can either be positive or negative valued due to the artificial compressibility imposed by the explicit time integration. In this case the boundary contact force $\mathbf{T}(P_a)$ acts repulsively when $P_a > 0$ or attractively when $P_a < 0$. Consequently, particles in local regions of expansion (where the pressure is negative) will be attracted to the boundary rather than repelled.

This is clearly a non–physical property of the current implementation. However, it is a consequence of the artificial compressibility rather than the derivation of the contact force itself.

This discrepancy would not occur in a fully–incompressible SPH formulation such as in the work of Shao [83,114]. In this case incompressibility is strictly enforced by ensuring $\nabla \cdot \mathbf{v} = 0$ and therefore the pressure field is correctly evaluated throughout the entire domain. Thereby, removing any non–physical negative pressures in the fluid caused by artificial compression waves.

While remaining within the current compressible SPH formulation several different modifications to the above variational contact force have been tested to assess the effect these attractive contact forces have on the simulations.

The first modification was to only apply the boundary contact force to particles with positive pressure

$$\mathbf{T}_a^B = \begin{cases} \mathbf{T}(P_a) & \text{if} \quad P_a > 0 \\ 0 & \text{if} \quad P_a \leqslant 0 \end{cases} . \tag{5.44}$$

The second modification was to apply a repulsive boundary contact forces irrespective of the sign of particle pressure

$$\mathbf{T}_a^B = \mathbf{T}(|P_a|) \quad \text{for all } P_a. \tag{5.45}$$

In both these cases the attractive contact forces have been removed and only repulsive boundary contact forces are present. However, it was found that both these modifications introduced instabilities into the code and the unmodified boundary contact force given by

$$\mathbf{T}_a^B = \mathbf{T}\left(P_a\right) \quad \text{for all } P_a \tag{5.46}$$

performed the best in most situations.

The relatively poor performance of these modifications is likely to be due to the fact that the pressures used in the modified expressions for the boundary contact forces are not consistent with those given by the equation of state which are used to evaluate the internal forces.

5.4 Variational boundary force examples

This section presents several applications using the variational boundary contact force derived in the previous section. A flexible SPH code has been developed for free surface fluid problems written in Fortran90. Complete details of the code and algorithms implemented can be found in Appendix D.

The boundaries are modelled by line segments and curved segments; no boundary particles are required. The contact forces are calculated using the theory developed in Section 5.3.2 for particles within $2h$ of a boundary.

The motion of the SPH particles are evolved using the equation of motion given by

$$m_a \mathbf{a}_a = \mathbf{F}_a - \mathbf{T}_a^P - \mathbf{T}_a^{\text{dev}} - \mathbf{T}_a^B \tag{5.47}$$

where the internal and contact forces are calculated using the following expressions

$$\mathbf{T}_a^P = \sum_b m_a m_b \left(\frac{P_a}{\rho_a^2 \gamma_a} + \frac{P_b}{\rho_b^2 \gamma_b}\right) \nabla w_b\left(\mathbf{x}_a, h_b\right), \tag{5.48a}$$

$$\mathbf{T}_a^{\text{dev}} = \sum_b V_a V_b \boldsymbol{\sigma}_b' \tilde{\nabla} \hat{w}_a\left(\mathbf{x}_b, h_a\right), \tag{5.48b}$$

$$\mathbf{T}_a^B = -\frac{m_a P_a}{\rho_a \gamma_a} \nabla \gamma_a \tag{5.48c}$$

and the only external force present is gravity given by $\mathbf{F}_a = m_a \mathbf{g}$.

The accuracy of the variational boundary force is investigated by the simple breaking dam benchmark. In the second example a more complex boundary is used to model a simple flood defence problem. The final example models a water droplet falling into a curved bowl demonstrating the methods ability to model curved boundaries.

5.4.1 Breaking dam example

The water column is 0.1m wide and 0.1m tall and initially held in hydrostatic equilibrium against the solid wall on the left as shown in Figure 5.23. At time $t = 0.0$s the water is released and allowed to collapse under gravity.

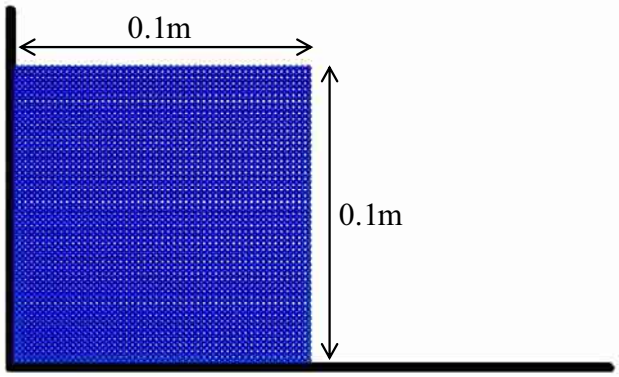


Figure 5.23: Breaking dam initial configuration

In the numerical model the water column is represented by 2500 equally spaced fluid particles with material density of $\rho_0 = 1000\text{kgm}^{-3}$ and viscosity $\mu = 0.5\text{kgm}^{-1}\text{s}^{-1}$. The boundary is identified by only three points defining the two straight line segments; no boundary particles are required. A constant timestep of 10^{-4}s is used throughout the simulation.

The artificial bulk modulus in the equation of state is calculated using equation (2.83) where $\mathbf{v}_{\max}$ is determined using the theory of conservation of energy. By considering a particle on the top surface of the dam its potential energy will be converted into kinetic energy yielding

$$mgH = \frac{1}{2}m \left\|\mathbf{v}_{\text{typ}}\right\|^2 \tag{5.49}$$

and so the typical fluid velocity is given by

$$\left\|\mathbf{v}_{\text{typ}}\right\| = \sqrt{2gH}. \tag{5.50}$$

With this value of $\mathbf{v}_{\text{typ}}$ the artificial bulk modulus is obtained as

$$P_0 = \frac{100 \times 2gH \times \rho}{\gamma}. \tag{5.51}$$

For this example $\gamma = 7$ and the bulk modulus is calculated to be $P_0 = 28028.57\ \text{Nm}^{-2}$.

In order to reflect the gravitational force acting on the water column while in equilibrium the initial density of the individual particles is calculated by combining the equation of state (2.81) with the hydrostatic pressure $P_a = \rho g h$ to give

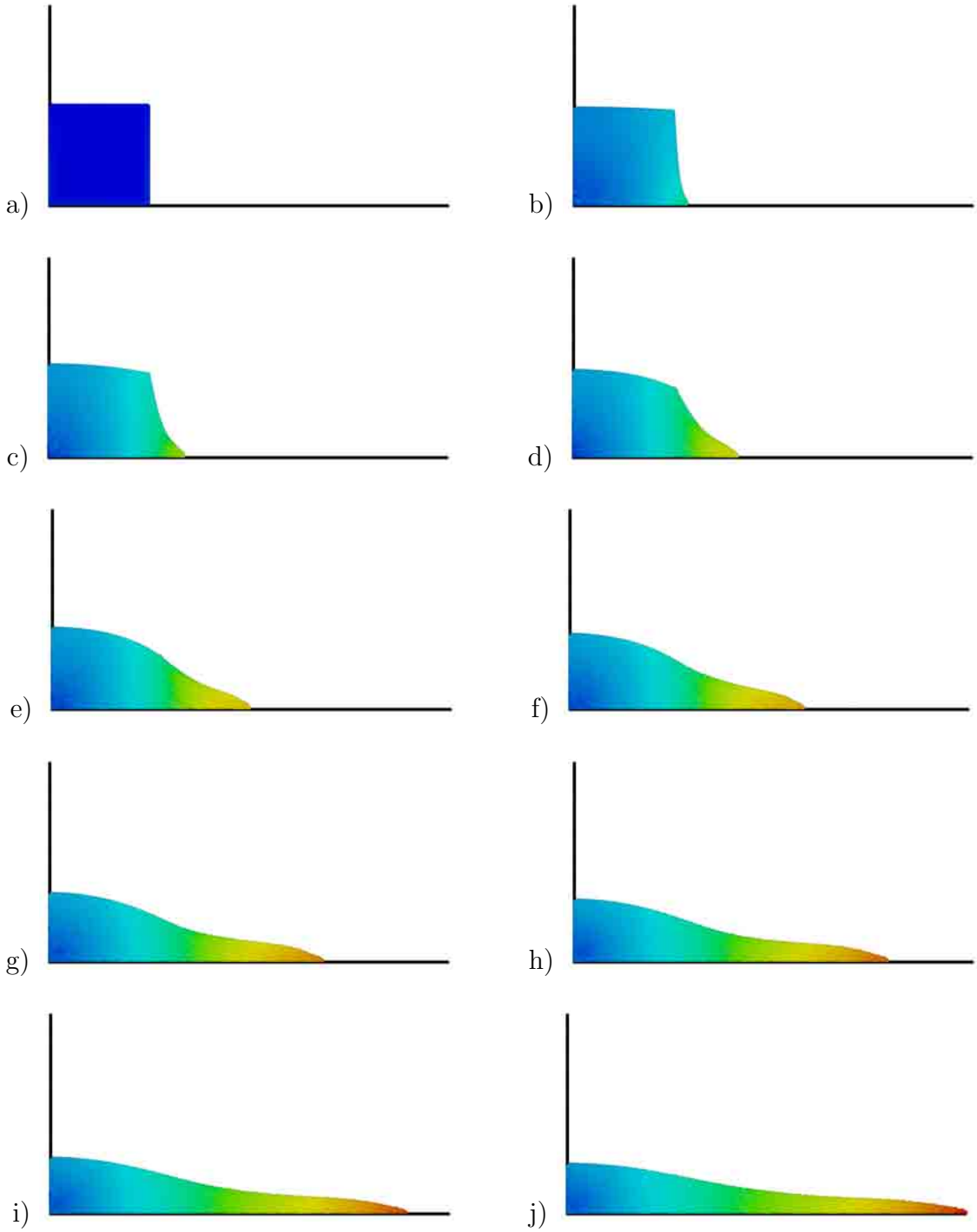
$$\rho_a = \rho_0 \left(1 + \frac{\rho_0 g\,(H - y_a)}{P_0}\right)^{\frac{1}{\gamma}} \tag{5.52}$$

where H is the initial height of the dam and y_a is the distance from the base of the dam to particle a.

Using the code developed for this thesis the results of Lok [84] where reproduced using the particle bounce–back method and the same simulation was run using the gamma function boundary implementation.

Figure 5.24 shows the dam velocities at various stages of collapse. In Figure 5.25 the surge front velocities obtained using the gamma function boundary implementation and the particle bounce–back method are compared to those obtained from experimental data [87]. For comparison the variables are non–dimensionalised with respect to the initial width of the dam, w. The non–dimensional time is given by $t\sqrt{g/w}$ and the non–dimensional surge front is given by x/w.

Both the current SPH simulations using the variational boundary force and the particle bounce–back method overestimate the surge front velocities when compared to the experimental data. This suggests that friction along the horizontal base should not be neglected in the computational model. This is corroborated by the fact that the bounce–back method which includes a coefficient of restitution term was in closer agreement to the experimental data than the new variational boundary force that does not include any dissipative term.



a) $t = 0.0$s b) $t = 0.035$s c) $t = 0.06$s d) $t = 0.085$s e) $t = 0.11$s
f) $t = 0.135$s g) $t = 0.16$s h) $t = 0.185$s i) $t = 0.21$s j) $t = 0.235$s

Figure 5.24: Breaking dam velocities

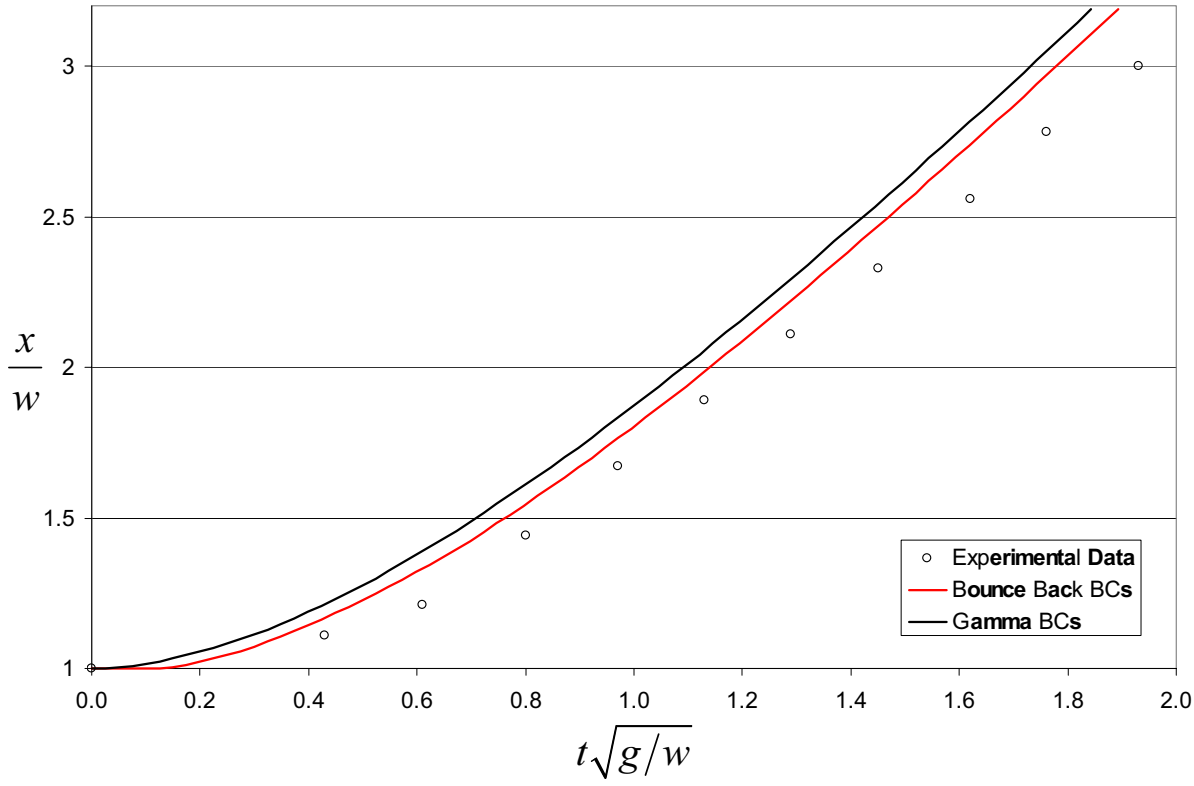


Figure 5.25: Surge front comparisons

5.4.2 Flood defense example

The second example applies the variational boundary contact force to a more complicated boundary to model a dam breaking against a flood defence as shown in Figure 5.26.

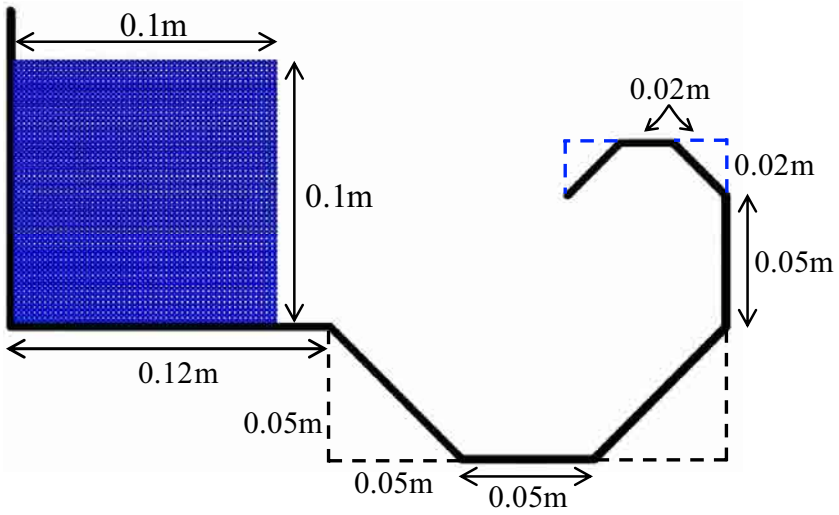


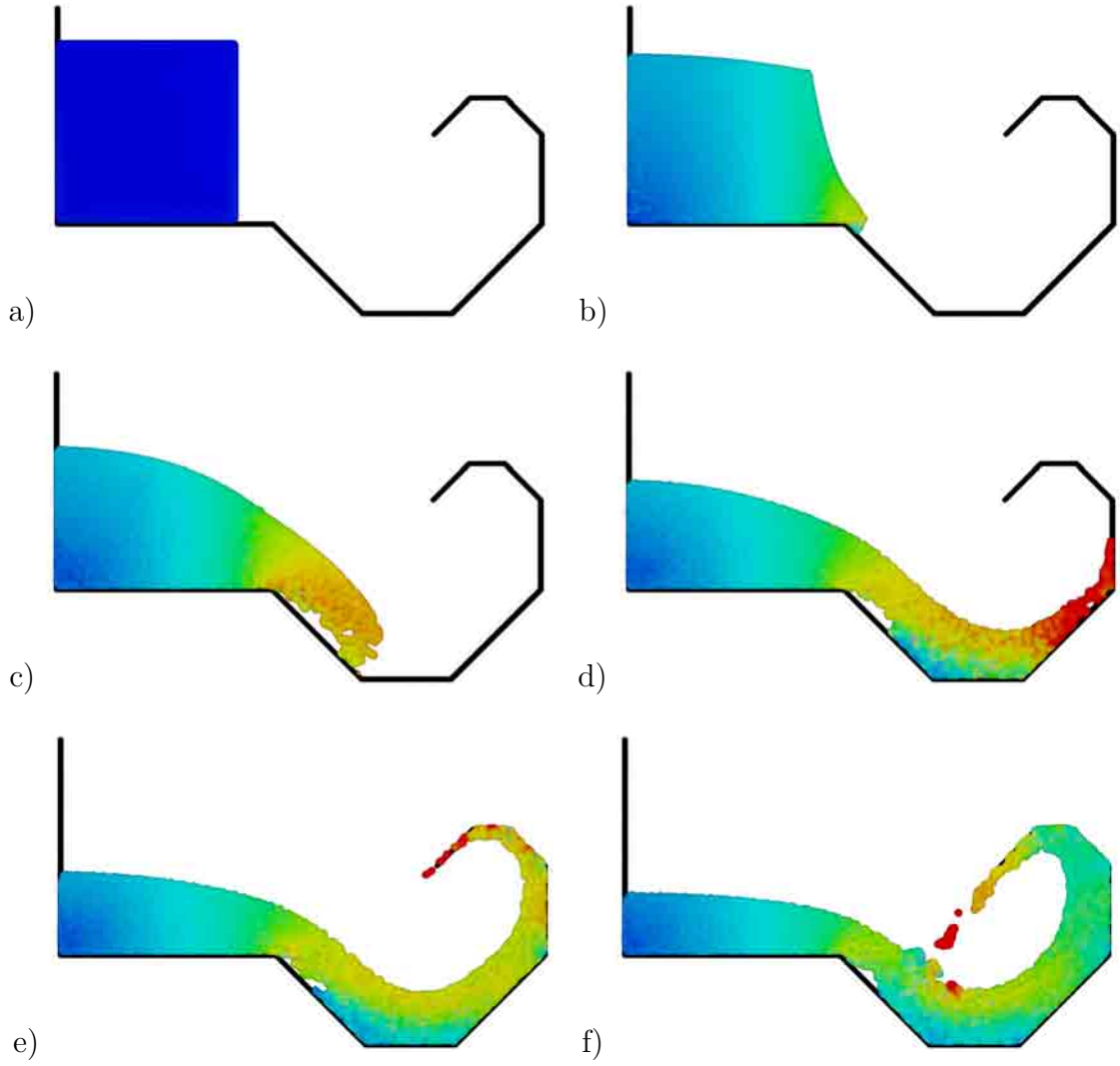
Figure 5.26: Flood defense initial configuration

The numerical discretization of the flood defence problem is identical to dam given in Section 5.4.1 except for the solid boundary which is represented by nine straight line segments. A constant timestep of 0.25×10^{-4}s is used throughout the simulation.

The resulting fluid flow is given in Figures 5.27–5.30 and demonstrate the new boundary implementation is capable of accurately modelling complicated straight line geometries in two dimensions.

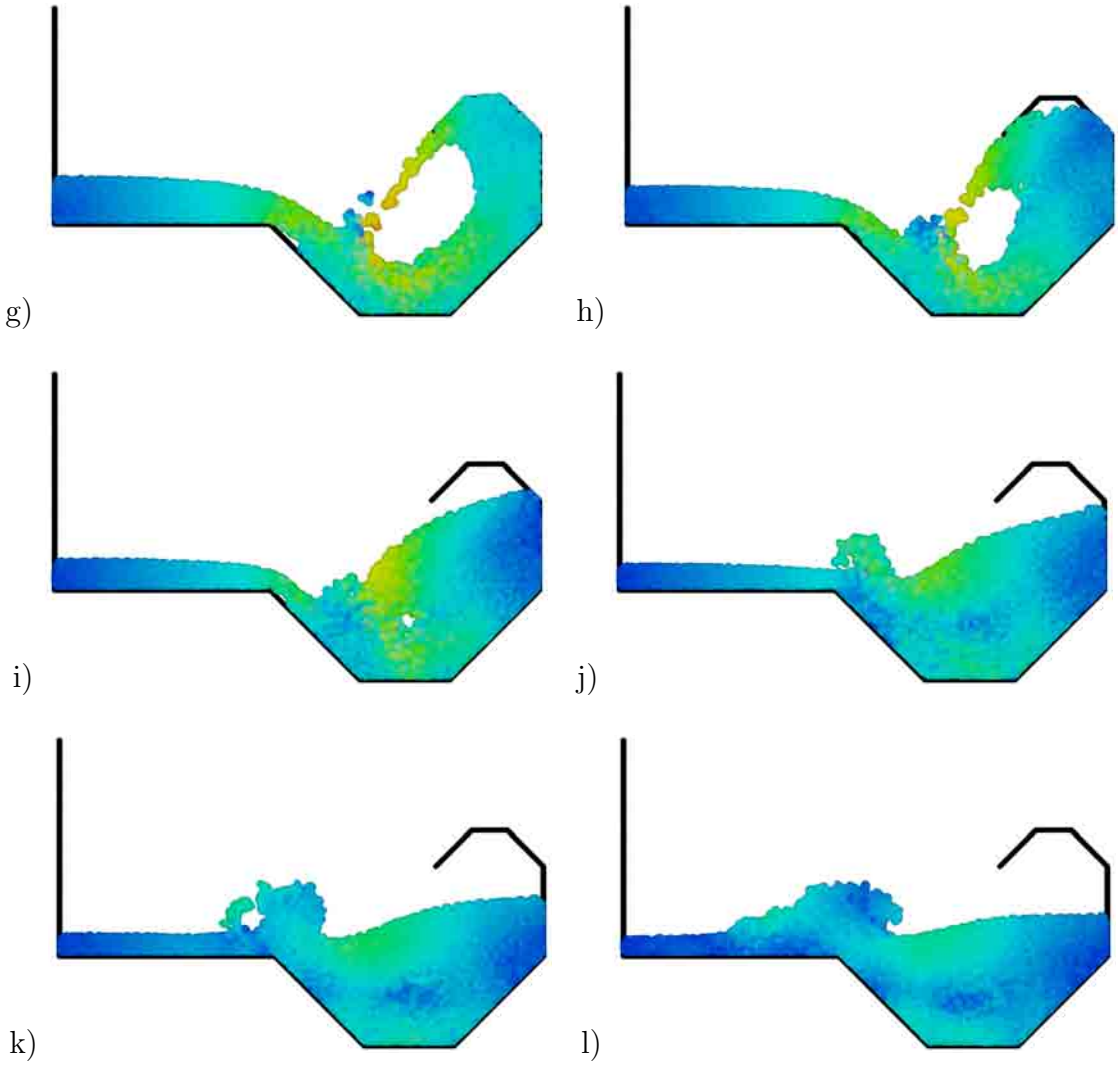
Figures 5.27 and 5.28 show the velocity of the fluid at various instances of the simulation. They show the void created by the fluid breaking away from boundary as it flows down the slope and its subsequent recirculation. The free surface generated as the wave breaks over the flood defence is clearly captured by the method.

Figures 5.29 and 5.30 show velocity vector plots of the fluid at various instances of the simulation. They clearly show the regions of stagnation and the evolution of the eddy generated by the converging flow caused by the breaking wave.



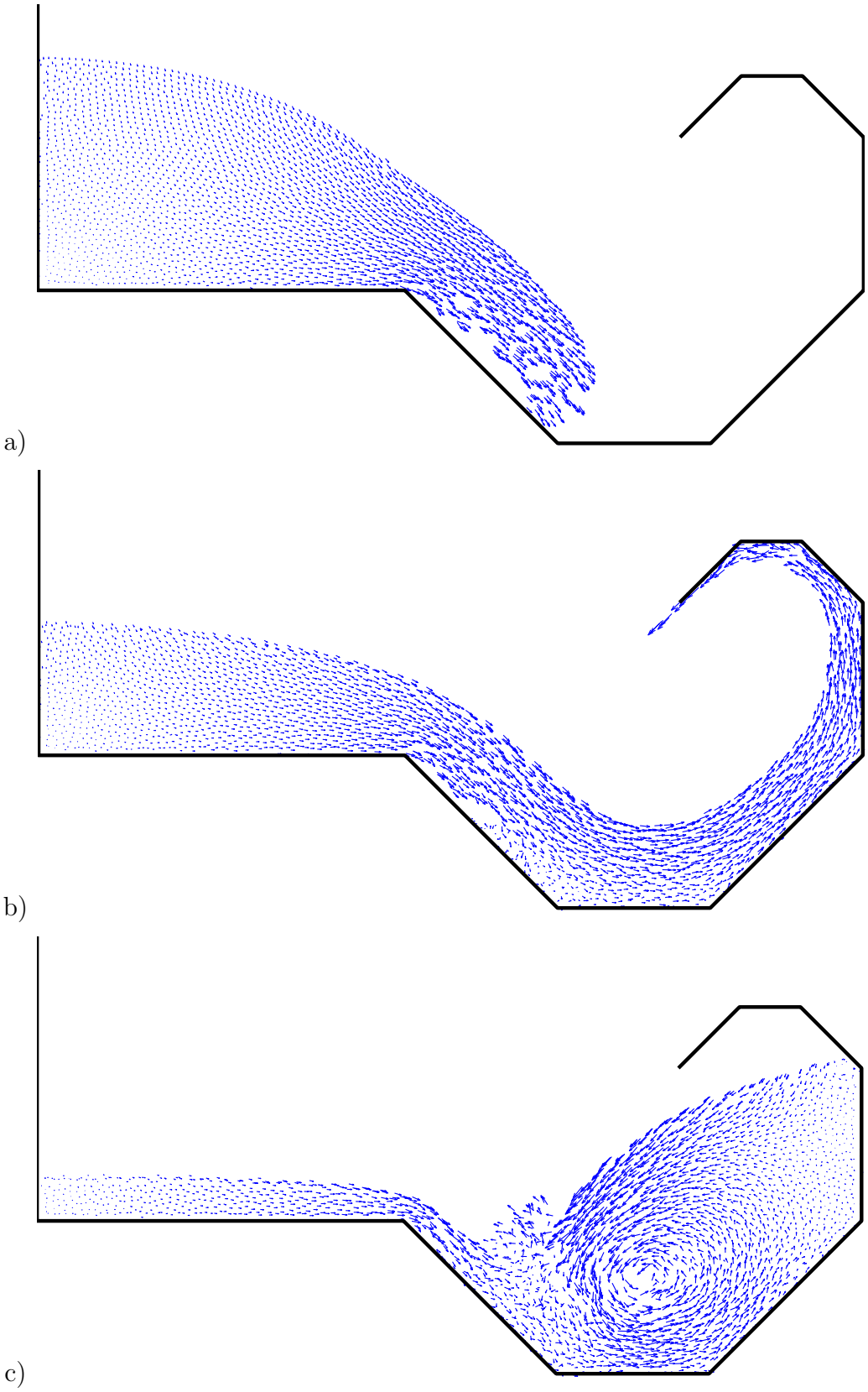
a) $t = 0.0$s b) $t = 0.062$s c) $t = 0.124$s d) $t = 0.186$s
e) $t = 0.248$s f) $t = 0.310$s

Figure 5.27: Flood defense velocities



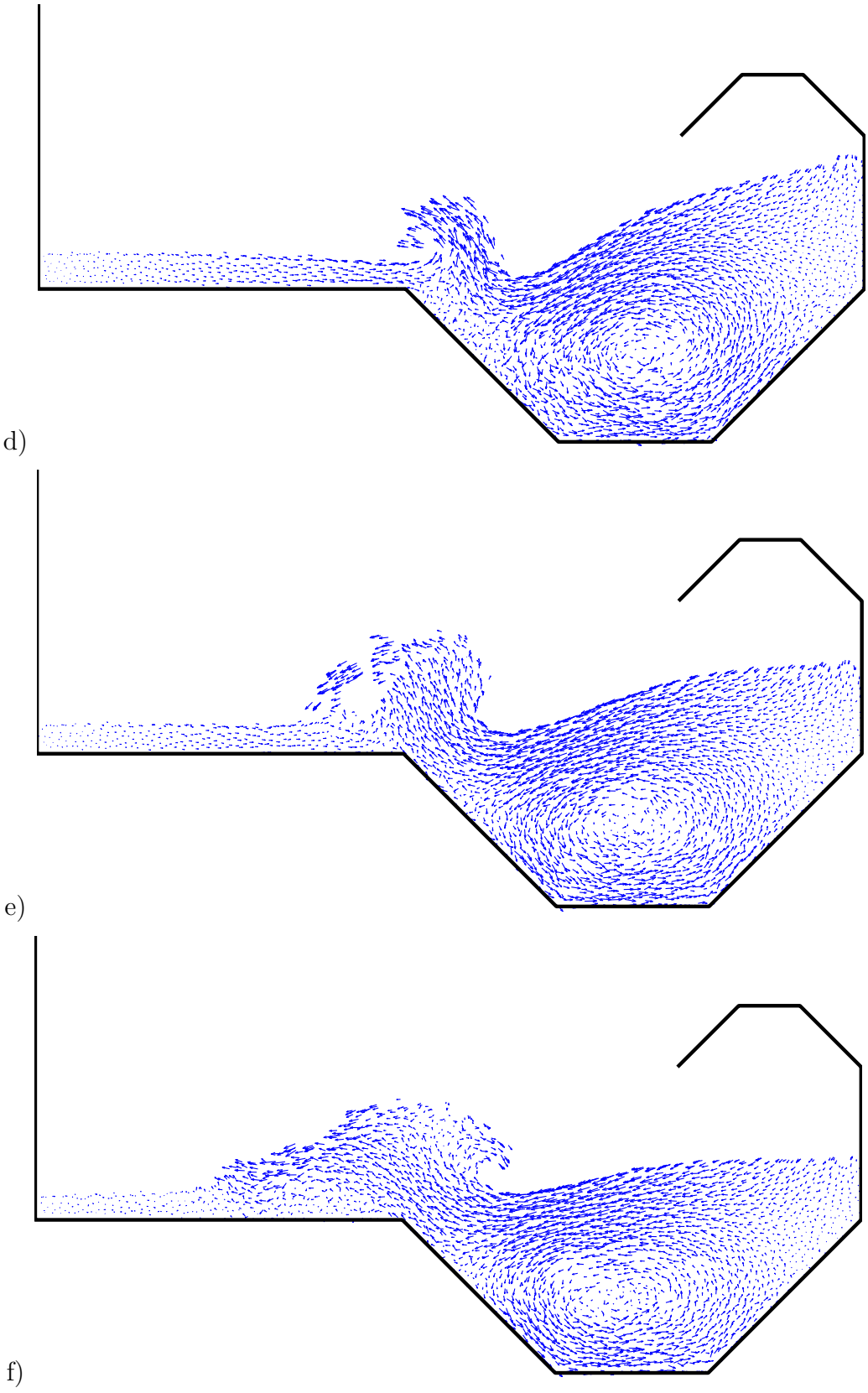
g) $t = 0.372$s h) $t = 0.434$s i) $t = 0.496$s j) $t = 0.558$s
k) $t = 0.620$s l) $t = 0.682$s

Figure 5.28: Flood defense velocities (cont.)



a) $t = 0.124$s b) $t = 0.248$s c) $t = 0.496$s

Figure 5.29: Flood defense velocity vector plot



d) $t = 0.558$s e) $t = 0.620$s f) $t = 0.682$s

Figure 5.30: Flood defense velocity vector plot (cont.)

5.4.3 Curved boundary example

The final example is of a water droplet falling into a semi–circular basin as shown in Figure 5.31. This demonstrates the ability of the new boundary contact force to model curved boundaries.

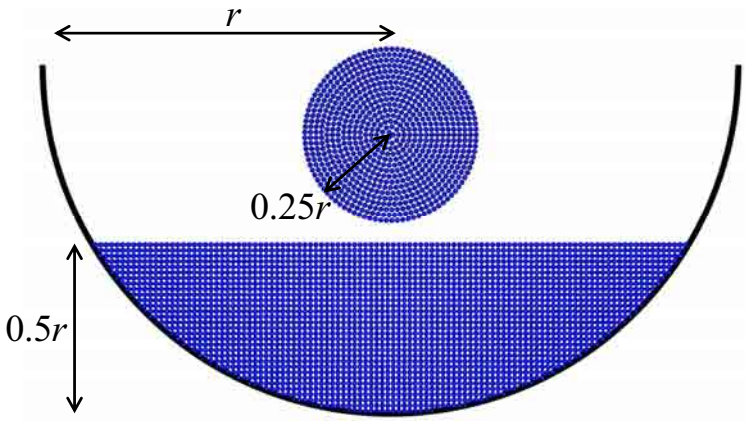
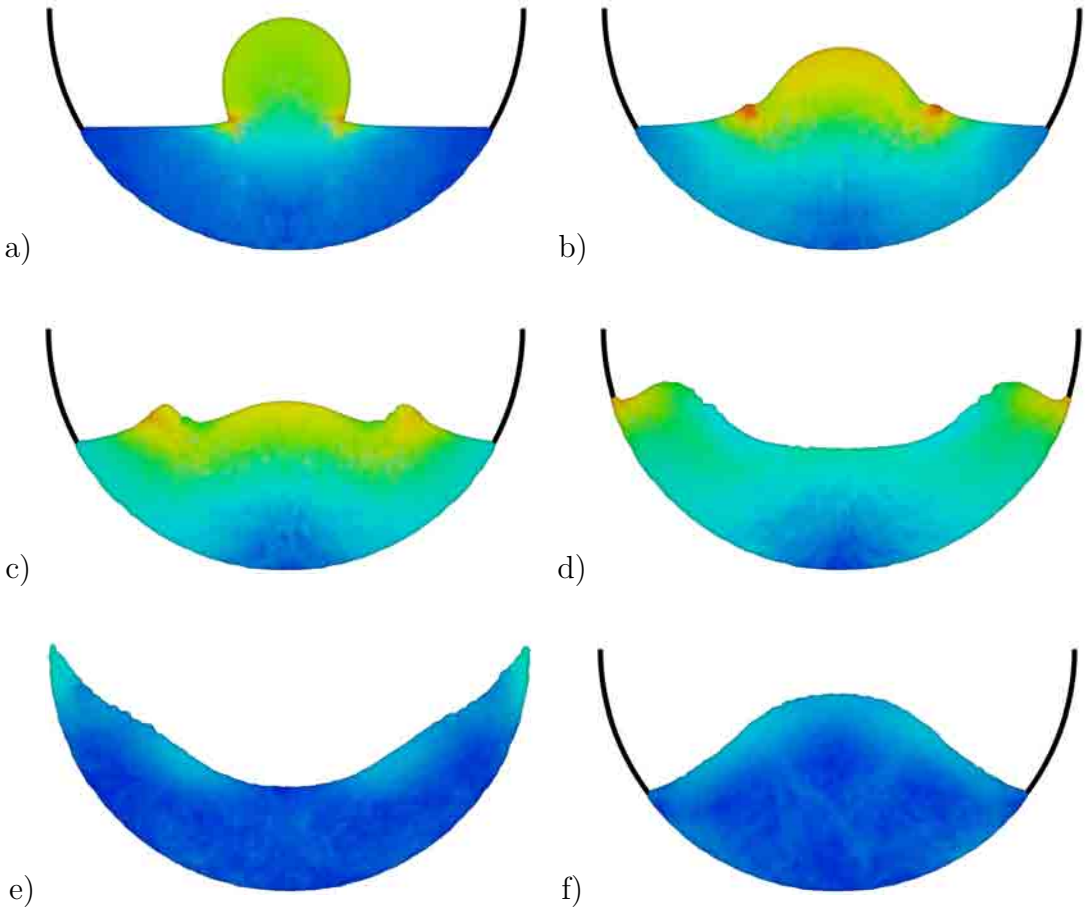


Figure 5.31: Water droplet initial configuration

The numerical model consists of a total of 3194 particles with material density of $\rho_0 = 1000\text{kgm}^{-3}$ and viscosity $\mu = 0.5\text{kgm}^{-1}\text{s}^{-1}$. The radius of the semi–circular basin is $r = 0.104\text{m}$ and is filled up to the halfway point with 2390 equally spaced fluid particles. The water droplet has radius $0.25r$ and consists of 804 fluid particles positioned in equally spaced concentric circles. A bulk modulus $P_0 = 25000\ \text{Nm}^{-2}$ is used to ensure density variations of less than 1%. A constant timestep of $0.25\times10^{-4}\text{s}$ is used throughout the simulation.

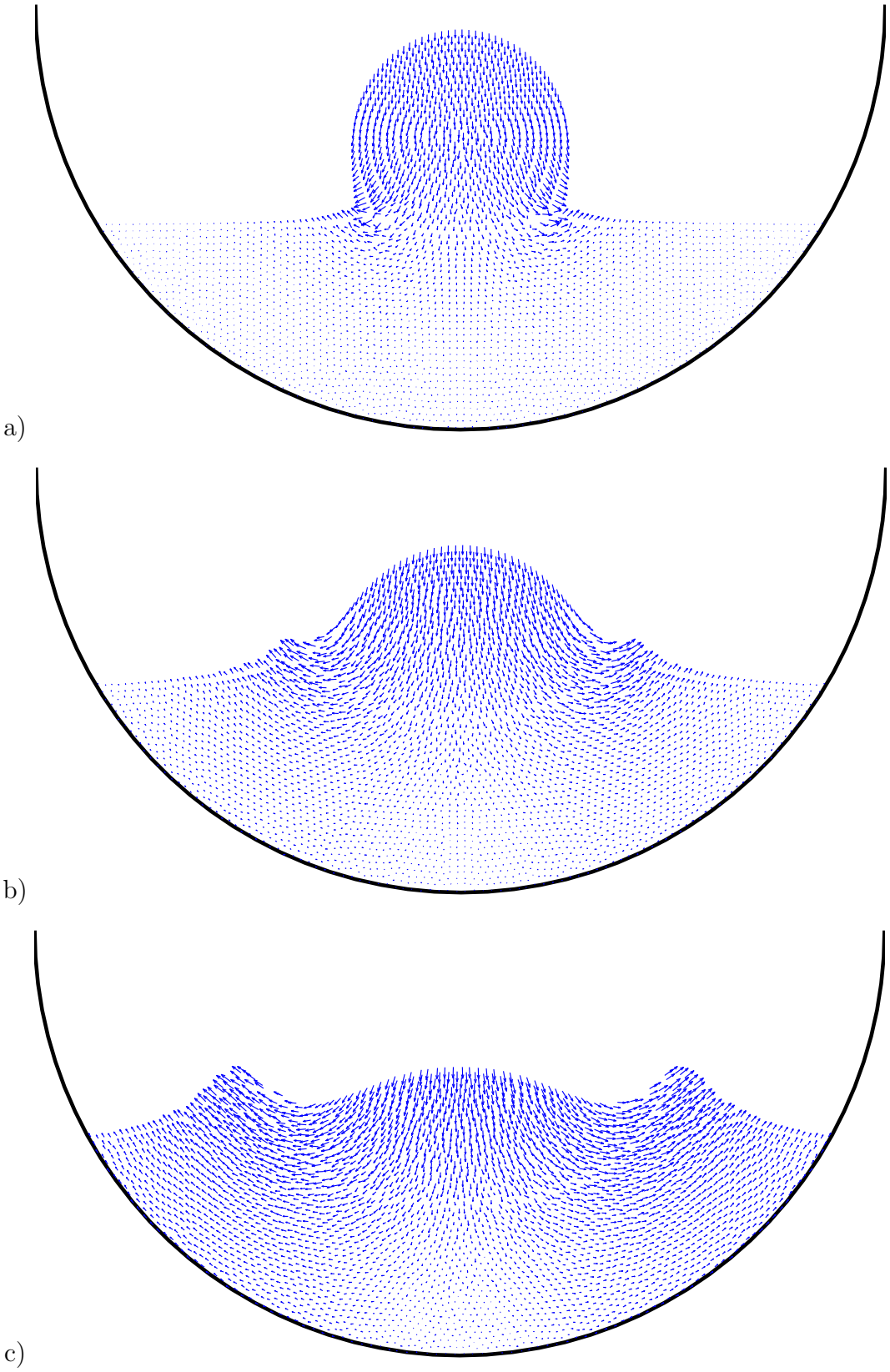
The semi–circular basin is defined by a single curved boundary segment as described in Section 5.3.2 and the boundary force is evaluated by approximating the boundary with straight line segments (see Figure 5.21).

Figure 5.32 shows the velocity of the water droplet and Figures 5.33 and 5.34 show velocity vector plots of the fluid at various instances of the simulation showing the new boundary implementation is capable of accurately modelling curved boundary segments.



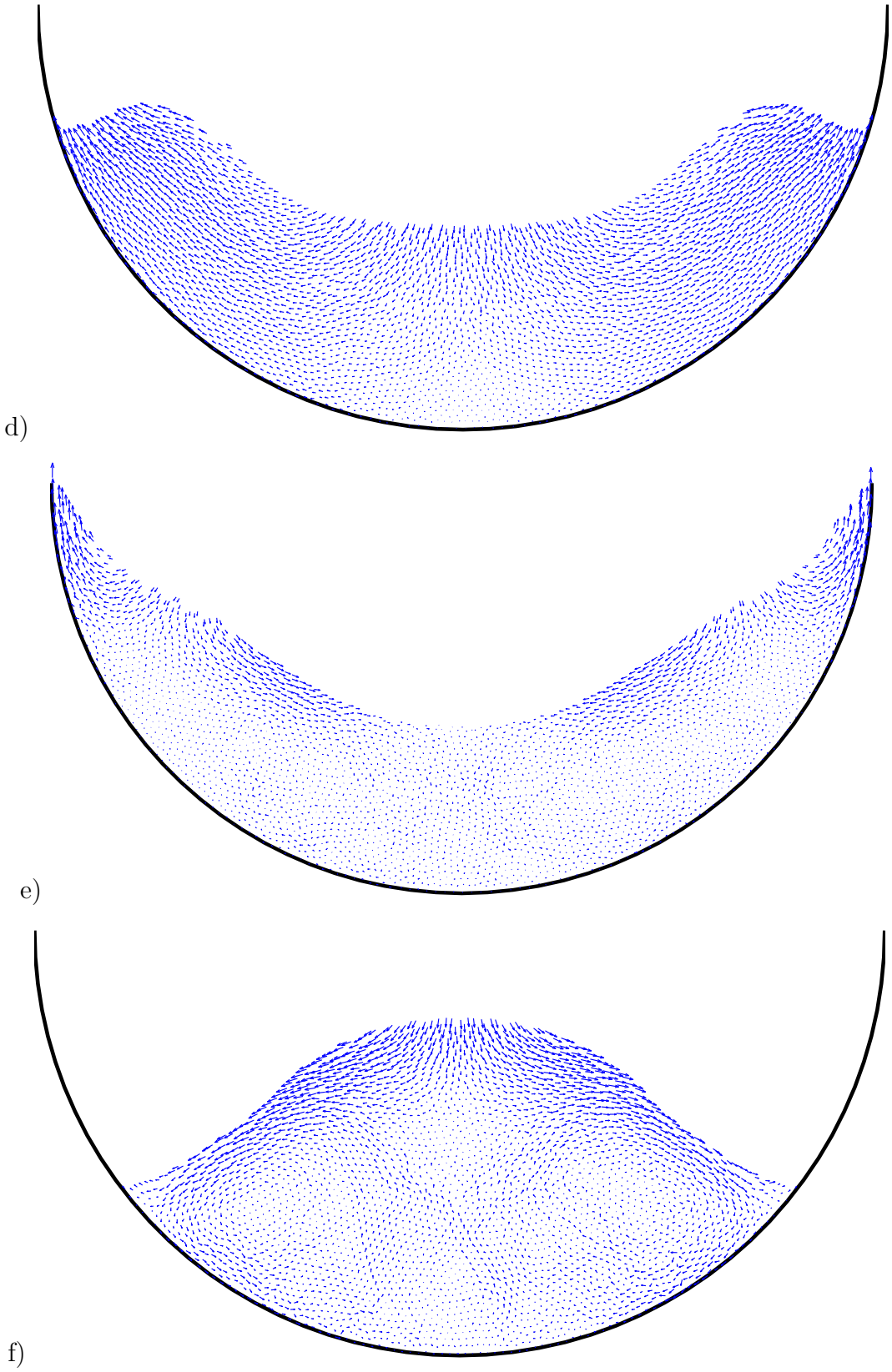
a) $t = 0.0475$s b) $t = 0.0725$s c) $t = 0.0975$s d) $t = 0.1477$s
e) $t = 0.21$s f) $t = 0.4225$s

Figure 5.32: Water droplet velocities



a) $t = 0.0475$s b) $t = 0.0725$s c) $t = 0.0975$s

Figure 5.33: Water droplet velocity vector plot



d) $t = 0.1477$s e) $t = 0.21$s f) $t = 0.4225$s

Figure 5.34: Water droplet velocity vector plot (cont.)

5.5 Concluding remarks

SPH was initially developed to solve problems in astrophysics where the domain of the simulations were infinite and notably absent of any solid boundaries. Indeed, the derivation for reproducing kernel approximation of the gradient of a function explicitly assumes the absence of solid boundaries and the terms coming from the boundary are neglected. Consequently, the SPH method does not naturally incorporate rigid or moving boundaries.

This chapter has presented several different methods which are commonly used to implement solid boundaries in SPH. These were the bounce back method, image particles and penalty methods. In addition the Lennard-Jones method has been described and a new formulation has been developed to avoid instabilities that have traditionally occurred when non-uniform boundary particle distributions are used.

A new expression for boundary contact forces has been developed from variational principles in Chapter 4. As such the resulting forces maintain the conservation properties of the underlying governing equations. A simple and inexpensive method for exactly evaluating these boundary forces for general boundaries in two dimensions has been presented and compared to the existing approximations.

The accuracy and flexibility of this formulation has been demonstrated by several free surface flow simulations. The flood defense and water droplet examples have shown the method is capable of modelling complex straight line and curved geometries in two dimensions.

This new variational boundary force implementation has been incorporated into the code developed for this thesis and will be used extensively in the refinement simulations that follow in Chapter 7.

Chapter 6

Adaptivity

6.1 Introduction to adaptivity in SPH

In the past most SPH derivations have been based on uniform distributions of particles of equal mass. This leads to large simulations with many particles and long run times. In other mesh based schemes it has become common place to use mesh adaptivity to improve numerical results and to reduce computation times. With a corresponding refinement strategy SPH can gain these same advantages.

In this chapter a refinement strategy based upon particle splitting is developed. Candidate particles are split into several 'daughter' particles according to a given refinement pattern centred about the original particle position. Through the solution of a non-linear minimisation problem the optimal mass distribution for the daughter particles is obtained so as to reduce the errors introduced into the underlying density field. This procedure necessarily conserves the mass of the system.

The density refinement errors for several daughter particle configurations in one, two and three dimensions are calculated and the optimal particle separations and smoothing lengths are obtained for each configuration.

Finally, conservation properties of particle splitting algorithms are discussed and it is proved that there is only one unique daughter particle velocity configuration that conserves both the kinetic energy and momentum of the system.

In order to implement dynamic refinement into a SPH framework two main considerations need to be dealt with:

- Firstly, to identify suitable *Criterion for Refinement* as a way to efficiently identify candidate particles for refinement and those particles for which refinement is not necessary.
- Secondly, to develop a *General Refinement Algorithm* whereby particles are refined into a number of corresponding *'daughter'* particles while ensuring that the basic properties of the system are conserved and that any errors introduced by the procedure are minimised.

Both stages should be applied automatically by an SPH code at runtime with no intervention required.

6.2 Criteria for refinement

Candidate particles for refinement could conceivably be identified by many different criteria depending on the type of problem to which refinement is being applied. It is not the aim of this thesis to study the criteria for refinement in detail, rather to develop an accurate and flexible refinement algorithm that is independent of the refinement criterion used.

Example refinement criteria

Refinement zones

A *refinement zone* is a region of a problem domain where particles automatically undergo refinement. As a particle passes into such a region the refinement algorithm replaces the original particle with its corresponding set of daughter particles. As the daughter particles leave the refinement zone they remain unchanged and continue to pass though the rest of the domain.

These regions can be placed in areas where more accuracy is desired or in regions where there are large differences in scale (see Figure 6.1). For example when water in a large tank flows into a narrow pipe refinement would allow a coarse distribution of particles in the tank with particles being refined as they travel through the

pipe. This reduces the total number of particles necessary for the simulation while maintaining the accuracy needed inside the narrow pipe.

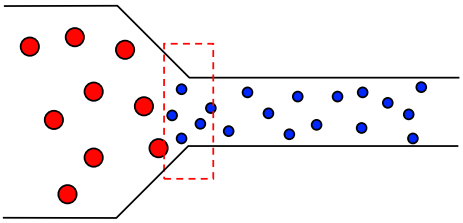


Figure 6.1: Example of a refinement zone through a narrow pipe

Number of neighbours

The number of neighbour particles is another obvious criterion to use for refinement. A particle with few neighbours is a suitable candidate for refinement since the SPH interpolation is likely to be poorer in its vicinity.

For example in problems with large dissipation the number of particle neighbours can vary considerably throughout the domain. When a given particle a has fewer neighbours than a prescribed minimum $n_a < n_{\min}$ then the particle could undergo refinement to maintain the local accuracy of the simulation as the distribution of particles thins out.

Problem specific criteria

Other physical quantities have already been used successfully as criterion for adaptivity in SPH. In the work of Lastiwka [70] particles are added in regions of high velocity gradient and removed in regions of low velocity gradient. In the one dimensional shock tube example they showed some improvement with adaptivity over the standard SPH method using a comparable number of particles.

Kitsionas [61, 62] has applied a particle splitting algorithm to an astrophysics problem concerned with the self-gravitating collapse of a region of gas. Here the refinement criterion is based on satisfying a physical requirement known as the 'Jeans Condition'. This ensured that the resolution of the particle distribution was sufficient to capture the physics of the problem.

6.3 A general refinement algorithm

Suppose that particle a with mass m_a, position $\mathbf{x}_a$, velocity $\mathbf{v}_a$ and smoothing length h_a has been identified as a candidate for refinement as shown in Figure 6.2. It remains to decide in some sense the 'best way' to introduce new particles into the simulation.

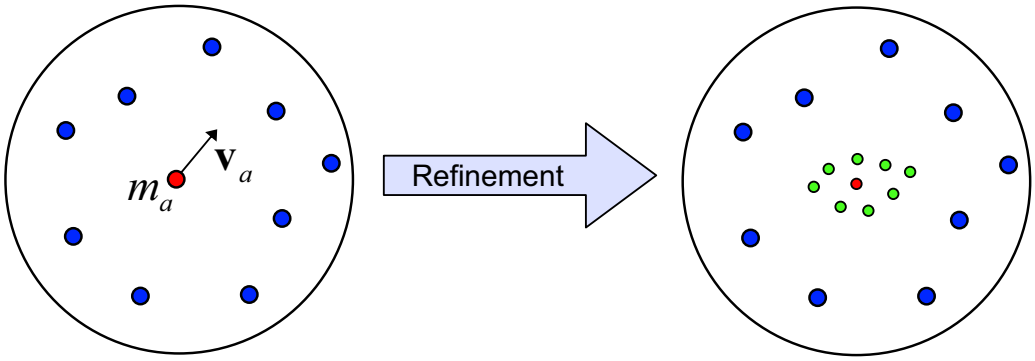


Figure 6.2: Particle refinement

There are several factors that need to be taken into account when devising a refinement algorithm:

- The addition of new particles will alter the local mass distribution and inevitably change the local density and velocity fields surrounding the refined particle. Any such change should be minimised by the refinement process.
- Regions where fine and coarse particles interact will be a consequence of the refinement process. Any proposed method must deal with such regions.
- Refined particles should have a reduced smoothing length corresponding to their smaller mass. This retains the local smoothing property of SPH and reduces the number of daughter particle neighbours.
- Where possible the refinement process should conserve the global properties such as the mass, kinetic energy, linear and angular momentum of the system.

6.3.1 Conservation

Suppose that the particle a is to be replaced by M daughter particles. Each daughter particle will have individual mass m_b, position $\mathbf{x}_b$, velocity $\mathbf{v}_b$ and smoothing length h_b to assign, where $b \in \{1, \cdots, M\}$.

$$m_b \ , \ \begin{pmatrix} x_b^1 \\ x_b^2 \\ x_b^3 \end{pmatrix} \ , \ \begin{pmatrix} v_b^1 \\ v_b^2 \\ v_b^3 \end{pmatrix} \text{ and } \ h_b.$$

This results in a total of eight degrees of freedom for each daughter particle in three dimensions. Ideally, these need to be chosen in such a way as to preserve the four global conservation properties given in Table 6.1.

	Before Refinement	After Refinement
(1)– Mass	m_a	$\sum_b m_b$
(2)– Kinetic Energy	$\frac{1}{2} m_a \mathbf{v}_a \cdot \mathbf{v}_a$	$\frac{1}{2} \sum_b m_b \mathbf{v}_b \cdot \mathbf{v}_b$
(3)– Linear Momentum	$m_a \mathbf{v}_a$	$\sum_b m_b \mathbf{v}_b$
(4)– Angular Momentum	$\mathbf{x}_a \times m_a \mathbf{v}_a$	$\sum_b \mathbf{x}_b \times m_b \mathbf{v}_b$

Table 6.1: Global Conservation Properties (1)–(4)

This chapter is concerned with assigning the daughter particles positions, velocities, masses, and smoothing lengths in such a way as to conserve the properties in Table 6.1 and to minimise the changes to the density and velocity fields that result from the refinement procedure.

It will be shown in Section 6.6 that there is only one possible refinement solution that simultaneously satisfies conservation properties (2)–(4). It is however possible to choose a mass distribution that conserves the global mass of the system (1) while minimising the error introduced to the local density field.

In the next section the positions of the refined daughter particles are given as a set of *refinement patterns* for several one, two and three dimensional cases. Each of these refinement patterns introduces an additional *separation parameter* $\varepsilon > 0$ that governs the spread of the daughter particles based around the original unrefined particle position. Each of these configurations will be studied in the remainder of this chapter or can be found in Appendix C.

In addition to fixing the separation of the daughter particles it is necessary to assume that their smoothing lengths are equal and proportional to the original smoothing length. This is given by $h_b = \alpha h$ for each particle b where the parameter $\alpha \in (0, 1]$ is known as the *smoothing ratio*.

Collectively the separation parameter ε and smoothing ratio α will be referred to as the *refinement parameter* (ε, α) of a given refinement pattern. Adopting these two simplifications means the number of degrees of freedom for a given refinement pattern are reduced to the choice of refinement parameter (ε, α), and the daughter particle masses m_b (in later analysis the daughter particle velocities $\mathbf{v}_b$ will also be considered).

For each of these refinement patterns optimal values for the separation parameter ε^* and daughter particle smoothing ratio α^* will be obtained with a corresponding set of daughter particle masses m_b^* such that the resulting changes to the density and velocity fields are minimised.

6.3.2 Refinement patterns

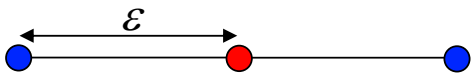
To proceed it is necessary to fix the positions of the new daughter particle with respect to the original particle position. In total five refinement patterns will be considered: two 1D and two 2D refinement patterns, and one 3D refinement pattern. In each case the separation of the particle configuration is governed by the additional separation parameter $\varepsilon > 0$.

Each refinement pattern is shown in Figure 6.3 through to Figure 6.7, the red point in each shows the original unrefined particle position. The coordinates of the daughter particles relative to this point are also included for the case where $\varepsilon = 1$ and particles are a unit distance apart from their nearest neighbour.

The orientation of the two and three dimensional refinement patterns do not necessarily have to be aligned with the coordinate axis but can be randomly rotated for each particle. However, for the current analysis and the simulations in Chapter 7 the orientation of the refinement patterns will remain fixed.

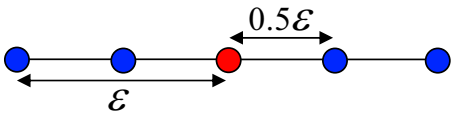
In the one dimensional cases the separation parameter ε denotes the maximum particle separation of the configuration with the daughter particles resulting from the five particle split positioned at distances 0.5ε and ε from the original particle.

1D refinement patterns



	x
1	0
2	-1
3	1

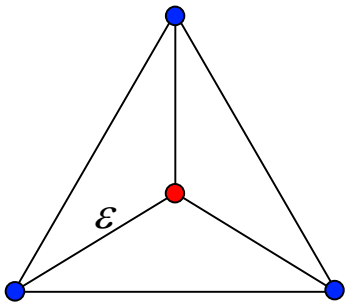
Figure 6.3: 1D 3–Particle refinement pattern with relative particle positions



	x
1	0
2	-1
3	$-\frac{1}{2}$
4	$\frac{1}{2}$
5	1

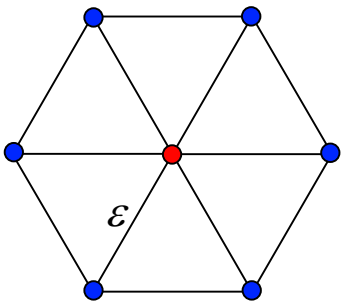
Figure 6.4: 1D 5–Particle refinement pattern with relative particle positions

2D refinement patterns



	x	y
1	0	0
2	0	1
3	$-\frac{\sqrt{3}}{2}$	$-\frac{1}{2}$
4	$\frac{\sqrt{3}}{2}$	$-\frac{1}{2}$

Figure 6.5: 2D triangular refinement pattern with relative particle positions



	x	y
1	0	0
2	1	0
3	-1	0
4	$\frac{1}{2}$	$\frac{\sqrt{3}}{2}$
5	$\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$
6	$-\frac{1}{2}$	$\frac{\sqrt{3}}{2}$
7	$-\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$

Figure 6.6: 2D hexagonal refinement pattern with relative particle positions

3D refinement pattern

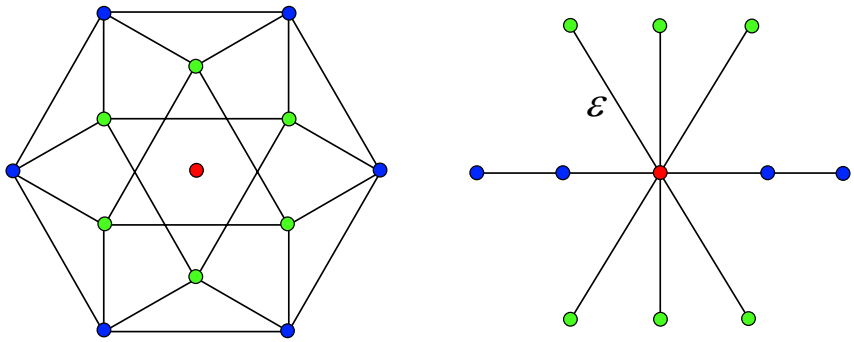


Figure 6.7: 3D hexagonal refinement pattern

	x	y	z
1	0	0	0
2	1	0	0
3	-1	0	0
4	$\frac{1}{2}$	$\frac{\sqrt{3}}{2}$	0
5	$\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$	0
6	$-\frac{1}{2}$	$\frac{\sqrt{3}}{2}$	0
7	$-\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$	0
8	0	$\frac{\sqrt{3}}{3}$	$-\sqrt{\frac{2}{3}}$
9	$-\frac{1}{2}$	$\frac{\sqrt{3}}{3} - \frac{\sqrt{3}}{2}$	$-\sqrt{\frac{2}{3}}$
10	$\frac{1}{2}$	$\frac{\sqrt{3}}{3} - \frac{\sqrt{3}}{2}$	$-\sqrt{\frac{2}{3}}$
11	0	$-\frac{\sqrt{3}}{3}$	$\sqrt{\frac{2}{3}}$
12	$-\frac{1}{2}$	$-\frac{\sqrt{3}}{3} + \frac{\sqrt{3}}{2}$	$\sqrt{\frac{2}{3}}$
13	$\frac{1}{2}$	$-\frac{\sqrt{3}}{3} + \frac{\sqrt{3}}{2}$	$\sqrt{\frac{2}{3}}$

Table 6.2: Relative particle positions for the 3D hexagonal refinement pattern

6.4 Density refinement error

In this section an error measure is defined from the change in local density field due to the refinement procedure. This error is shown to be independent of the initial particle mass and smoothing length. The optimum daughter particle masses are then calculated for a given distribution through the solution of a constrained minimisation problem.

The particles in SPH act as interpolation points for the underlying fields of interest and are not simply discrete particles. Any variable can be evaluated at any point in the domain (not only at the particle positions).

In particular, consider a collection of N particles and suppose that the N^{th} particle is to be refined into M daughter particles. Each of these M daughter particles having an as yet unspecified mass denoted by $m_b^* := m_{N_b}^*$ where $N_b \in \{1, \cdots, M\}$.

Before refinement the expression for the density at any point in the domain is given by

$$\rho(\mathbf{x}) = \sum_{a}^{N} m_a w_a(\mathbf{x}, h_a) \qquad \forall\ \mathbf{x} \in \Omega. \tag{6.1}$$

After refinement the local density distribution will inevitably change due to the redistribution of particles (see Figure 6.8) and is given by

$$\rho^*(\mathbf{x}) = \sum_{a=1}^{N-1} m_a w_a(\mathbf{x}, h_a) + \sum_{b=1}^{M} m_b^* w_b(\mathbf{x}, h_b) \qquad \forall\ \mathbf{x} \in \Omega. \tag{6.2}$$

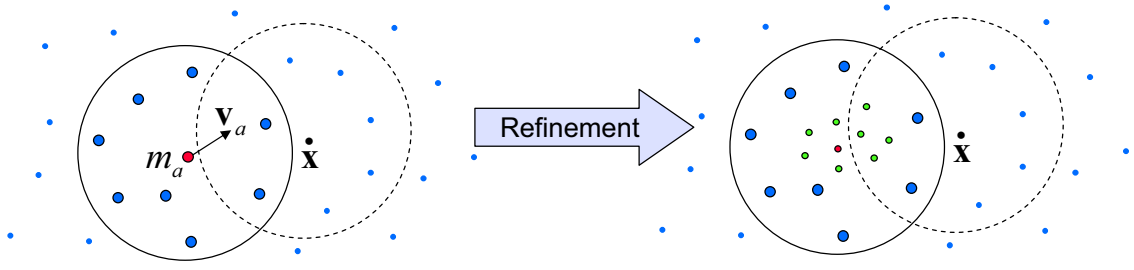


Figure 6.8: Change in contribution to density at $\mathbf{x}$ due to refinement of particle a

The *local density refinement error* at any point $\mathbf{x}$ can now be defined as the change in density interpolation due to the introduction of the new particles

$$e\left(\mathbf{x}\right) := \rho\left(\mathbf{x}\right) - \rho^{*}\left(\mathbf{x}\right) = m_N w_N\left(\mathbf{x}, h_N\right) - \sum_{b=1}^{M} m_b^{*} w_b\left(\mathbf{x}, h_b\right) \qquad \forall\ \mathbf{x} \in \Omega. \tag{6.3}$$

This change in local density field is entirely due to the ability of the refined particles to approximate the contribution of the original unrefined particle that they replace.

From the local error the *global density refinement error* can now be defined as

$$0 \leq \mathcal{E} := \int_{\Omega} e^2\left(\mathbf{x}\right)\ d\mathbf{x}. \tag{6.4}$$

Any adaptivity algorithm should be defined in such a way as to minimise this global density refinement error.

The above definitions for local and global density refinement errors are independent of the masses, positions, and smoothing lengths of the new daughter particles and so can be applied to any given refinement pattern.

Expanding the expression for $\mathcal{E}$ gives

$$\begin{aligned} \mathcal{E} &= \int_{\Omega} \left[m_N w_N\left(\mathbf{x}, h_N\right) - \sum_{b=1}^{M} m_b^{*} w_b\left(\mathbf{x}, h_b\right) \right]^2 d\mathbf{x}, \\ \mathcal{E} &= m_N^2 \int_{\Omega} w_N^2\left(\mathbf{x}, h_N\right) d\mathbf{x} - 2 m_N \sum_{b=1}^{M} m_b^{*} \int_{\Omega} w_N\left(\mathbf{x}, h_N\right) w_b\left(\mathbf{x}, h_b\right) d\mathbf{x} \\ &\qquad + \sum_{a,b=1}^{M} m_a^{*} m_b^{*} \int_{\Omega} w_a\left(\mathbf{x}, h_a\right) w_b\left(\mathbf{x}, h_b\right) d\mathbf{x}. \end{aligned} \tag{6.5}$$

By writing the masses of the daughter particles in terms of scalar parameters $\lambda_b > 0$ as $m_b^{*} = \lambda_b m_N$ for each b the global error is found to be proportional to the mass of the particle under refinement

$$\mathcal{E} = m_N^2 \left(\begin{array}{c} \displaystyle\int_{\Omega} w_N^2\left(\mathbf{x}, h_N\right) d\mathbf{x} - 2 \sum_{b=1}^{M} \lambda_b \int_{\Omega} w_N\left(\mathbf{x}, h_N\right) w_b\left(\mathbf{x}, h_b\right) d\mathbf{x} \\ \displaystyle + \sum_{a,b=1}^{M} \lambda_a \lambda_b \int_{\Omega} w_a\left(\mathbf{x}, h_a\right) w_b\left(\mathbf{x}, h_b\right) d\mathbf{x} \end{array} \right). \tag{6.6}$$

Conservation of mass is now enforced with the additional constraint on the λ_b parameters

$$\sum_{b=1}^{M} \lambda_b = 1. \tag{6.7}$$

Written more succinctly in vector form equation (6.6) is given by

$$0 \leqslant \mathcal{E} = m_N^2 \left(C - 2\boldsymbol{\lambda}^T \mathbf{b} + \boldsymbol{\lambda}^T \mathbf{A} \boldsymbol{\lambda} \right) \tag{6.8}$$

where $\boldsymbol{\lambda}$ is the vector of the unknown λ_a terms that determine the daughter particle masses by $m_b^* = \lambda_b m_N$.

The constant term C and components of the vector term $\mathbf{b}$ and matrix term $\mathbf{A}$ are shown diagrammatically in Figure 6.9 and are given by the following expressions

$$\begin{gathered} C = \int_{\Omega} w_N^2 \left(\mathbf{x}, h_N \right) d\mathbf{x} \ , \quad b_j = \int_{\Omega} w_N \left(\mathbf{x}, h_N \right) w_j \left(\mathbf{x}, h_j \right) d\mathbf{x} \ , \\ A_{ij} = \int_{\Omega} w_i \left(\mathbf{x}, h_i \right) w_j \left(\mathbf{x}, h_j \right) d\mathbf{x}. \end{gathered} \tag{6.9}$$

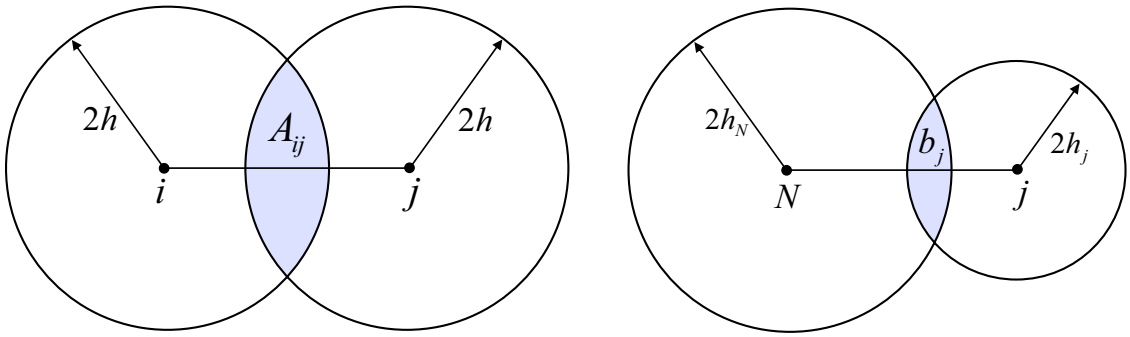


Figure 6.9: Graphical representation of A_{ij} and b_j terms

Equation (6.8) shows that the minimum of the global refinement error is independent of m_N. However, the magnitude of the error is found to be proportional to the unrefined mass m_N^2.

In the next section it will be shown that the density refinement error can also be made independent of the initial smoothing length h_N by writing the integral terms C, $\mathbf{b}$ and $\mathbf{A}$ in non–dimensional forms.

6.4.1 Derivation of the model problem

SPH kernel functions written in terms of the non–dimensional variable ξ are of the form

$$w(\xi) = \frac{1}{h^d} f(\xi) \qquad \text{where} \qquad \xi = \frac{r}{h} \in [0,2] \tag{6.10}$$

and d is the dimension of the problem.

Considering the integral expression for A_{ij}

$$A_{ij} = \int_{\Omega} w_i(\mathbf{x}, h_i)\, w_j(\mathbf{x}, h_j)\, d\mathbf{x} \tag{6.11}$$

and given that the daughter particles smoothing lengths are defined as $h_j = \alpha h_N$ where α is the fixed smoothing parameter, $\alpha \in (0,1]$ the expression for A_{ij} can be written as

$$A_{ij} = \frac{1}{\left(\alpha^d h_N^d\right)^2} \int_{\Omega} f_i(\xi)\, f_j(\xi)\, d\mathbf{x} \quad \text{where} \quad \xi = \frac{r}{\alpha h_N}\ . \tag{6.12}$$

Using the change of variable

$$d\hat{\mathbf{x}} = \frac{1}{(\alpha h_N)^d}\, d\mathbf{x} \ \Rightarrow\ d\mathbf{x} = (\alpha h_N)^d\, d\hat{\mathbf{x}} \tag{6.13}$$

where $d\hat{\mathbf{x}}$ is now a non–dimensional volume element A_{ij} can be written as

$$\begin{aligned} A_{ij} &= \frac{1}{\left(\alpha^d h_N^d\right)^2} \int_{\Omega} f_i\left(\hat{\xi}\right) f_j\left(\hat{\xi}\right) \alpha^d h_N^d d\hat{\mathbf{x}} \\ &= \frac{1}{\alpha^d h_N^d} \int_{\Omega} f_i\left(\hat{\xi}\right) f_j\left(\hat{\xi}\right) d\hat{\mathbf{x}} \end{aligned} \tag{6.14}$$

where

$$\hat{\xi} = \frac{\alpha h_N}{\alpha h_N}\, \hat{r} = \hat{r} \in [0,2]\,. \tag{6.15}$$

Therefore for any h

$$\int_{\Omega} w_i(\mathbf{x}, h)\, w_j(\mathbf{x}, h)\, d\mathbf{x} = \frac{1}{h^d} \int_{\Omega} w_i(\hat{\mathbf{x}}, 1)\, w_j(\hat{\mathbf{x}}, 1)\, d\hat{\mathbf{x}}. \tag{6.16}$$

This has the affect of scaling the integral down to unit smoothing length $h = 1$ as shown in Figure 6.10.

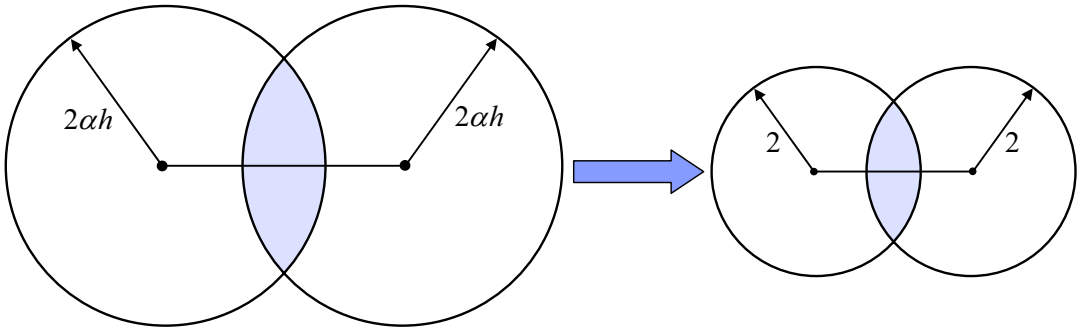


Figure 6.10: Scaling integrals into non–dimensional form

In the same way the b_j terms can be written in a non–dimensional form. Substituting equation (6.10) for both kernel functions gives

$$\begin{gathered}\forall\ h \ \text{ and } \ \alpha \in (0,1] \\ b_j = \int_\Omega w_N(\mathbf{x},h)\, w_j(\mathbf{x},\alpha h)\, d\mathbf{x} = \frac{1}{h^d}\ \frac{1}{\alpha^d h^d} \int_\Omega f_N(\xi_1)\, f_j(\xi_2)\, d\mathbf{x} \\ \text{where} \quad \xi_1 = \frac{r_1}{h} \quad \text{and} \quad \xi_2 = \frac{r_2}{\alpha h}\ .\end{gathered} \tag{6.17}$$

As before a change of variable yields

$$\begin{gathered} d\hat{\mathbf{x}} = \frac{1}{h^d} d\mathbf{x} \ \Rightarrow\ d\mathbf{x} = h^d d\hat{\mathbf{x}} \\ \text{where} \quad \hat{\xi}_1 = \frac{h}{h}\, r_1' = r_1' \in [0,2] \quad \text{and} \quad \hat{\xi}_2 = \frac{h\ r_2'}{\alpha h} = \frac{r_2'}{\alpha} \in [0,2]\,. \end{gathered} \tag{6.18}$$

With these substitutions b_i is given by

$$b_j = \frac{1}{h^d}\ \frac{1}{\alpha^d h^d} \int_\Omega f_N\left(\hat{\xi}_1\right) f_j\left(\hat{\xi}_2\right) h^d d\hat{\mathbf{x}} = \frac{1}{h^d} \left(\frac{1}{\alpha^d} \int_\Omega f_N\left(\hat{\xi}_1\right) f_j\left(\hat{\xi}_2\right) h^d d\hat{\mathbf{x}} \right). \tag{6.19}$$

Therefore for any smoothing length h and smoothing ratio α

$$\int_\Omega w_N(\mathbf{x},h)\, w_j(\mathbf{x},\alpha h)\, d\mathbf{x} = \frac{1}{h^d} \left(\int_\Omega w_N(\hat{\mathbf{x}},1)\, w_j(\hat{\mathbf{x}},\alpha)\, d\hat{\mathbf{x}} \right). \tag{6.20}$$

Proceeding in this way the global refinement error can written independently of both the smoothing length and the mass of the original particle by using the non-dimensional form of the integrals

$$0 \leqslant \mathcal{E}[\varepsilon,\alpha](\boldsymbol{\lambda}) = \frac{m_N^2}{h_N^d} \left(\bar{C} - 2\ \boldsymbol{\lambda}^T \bar{\mathbf{b}} + \boldsymbol{\lambda}^T \bar{\mathbf{A}}\ \boldsymbol{\lambda} \right) \tag{6.21}$$

where the coefficients are now obtained as

$$\bar{C} = \int_{\Omega} w_N^2 \left(\hat{\mathbf{x}}, 1\right) d\hat{\mathbf{x}} \ , \quad \bar{b}_j = \int_{\Omega} w_N \left(\hat{\mathbf{x}}, 1\right) w_j \left(\hat{\mathbf{x}}, \alpha\right) d\hat{\mathbf{x}} \ ,$$
$$\bar{A}_{ij} = \frac{1}{\alpha^d} \int_{\Omega} w_i \left(\hat{\mathbf{x}}, \alpha\right) w_j \left(\hat{\mathbf{x}}, \alpha\right) d\hat{\mathbf{x}}. \tag{6.22}$$

The notation $\mathcal{E}\left[\varepsilon, \alpha\right](\boldsymbol{\lambda})$ emphasises the dependence of $\mathcal{E}$ on the choice of refinement parameter (ε, α). In addition, the minimum error obtained from an optimal mass distribution $\boldsymbol{\lambda}^*$ is independent of the initial mass m_N and smoothing length h_N of the particle under refinement. However, the magnitude of the error is proportional to

$$\mathcal{E}\left[\varepsilon, \alpha\right](\boldsymbol{\lambda}) \propto \frac{m_N^2}{h_N^d} \ . \tag{6.23}$$

Therefore, the initial mass and smoothing length of the particle under refinement can be ignored while minimising the density refinement error and the optimal mass distribution can be calculated for any SPH simulation via the solution of the Model Problem.

The model problem

> *Given a particle of unit mass and unit smoothing length and a given refinement pattern that splits it into M daughter particles with refinement parameter (ε, α) find $\lambda_j^* \geq 0$ for $j = 1, \cdots, M$ such that*

$$\mathcal{E}^*\left[\varepsilon, \alpha\right] = \min_{\boldsymbol{\lambda}} \mathcal{E}\left[\varepsilon, \alpha\right](\boldsymbol{\lambda}) = \bar{C} - 2\ \boldsymbol{\lambda}^{*T} \bar{\mathbf{b}} + \boldsymbol{\lambda}^{*T} \bar{\mathbf{A}}\ \boldsymbol{\lambda}^* \quad \textit{where} \quad \sum\nolimits_{j=1}^{M} \lambda_j^* = 1.$$

The coefficients $\bar{C}$, $\bar{\mathbf{b}}$ and $\bar{\mathbf{A}}$ are calculated numerically and the above minimisation problem can be solved easily by standard non-linear programming methods. The solution to this problem is guaranteed since $\mathcal{E}$ can be shown to be a convex function due to the fact that $\bar{\mathbf{A}}$ is symmetric positive definite (SPD).

The actual refinement error introduced when a particle of mass m_N with smoothing ratio h_N is refined is then obtained from the model problem solution by

$$\mathcal{E} = \frac{m_N^2}{h_N^d}\ \mathcal{E}^*. \tag{6.24}$$

6.5 Density refinement results

The magnitude of the refinement error $\mathcal{E}[\varepsilon, \alpha]$ is still dependent upon the separation of the daughter particles and their smoothing lengths (see Figure 6.11). Some values of (ε, α) will result in much smaller refinement errors than others. By solving the model problem over a range of these parameters certain pairs $(\varepsilon^*, \alpha^*)$ can be identified as admissible choices that result in sufficiently small refinement errors.

The selection of these parameters also has practical implications to SPH simulations. A large value of smoothing ratio means that daughter particles will have a large number of neighbour particles at a greater computational expense than if the smoothing lengths we chosen more appropriately. Daughter particle separation should also be chosen to avoid a clumped or spaced out distribution of daughter particles.

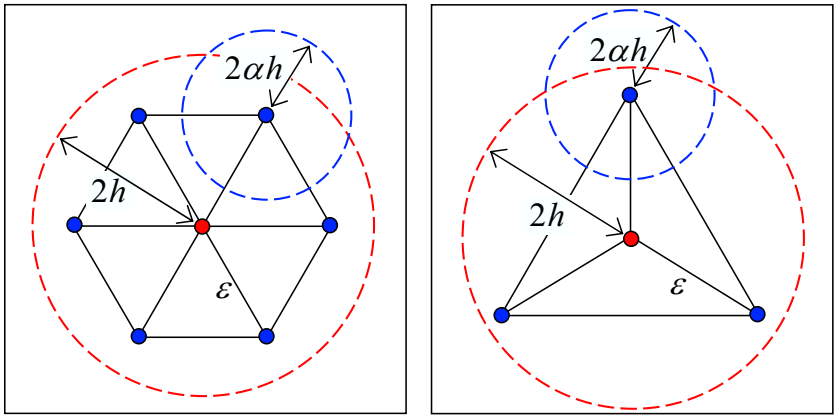


Figure 6.11: 2D refinement examples for given (ε, α)

This section presents the results obtained from solving the model problem for density refinement. For each of the refinement patterns given in Section 6.3.2 the model problem is solved for values of smoothing ratio and separation parameter (ε, α) ranging from $0.1 \rightarrow 0.9$. With each choice of (ε, α) the solution to the model problem results in the optimal daughter particle mass distribution with the corresponding minimised density refinement error $\mathcal{E}^*[\varepsilon, \alpha]$.

The results for the one dimensional refinement patterns are presented in detail below. In this case the particle is split into three or five particles respectively, centred about the original particle position as shown in Figure 6.12. The results for the two and three dimensional refinement patterns can be found in Appendix C.

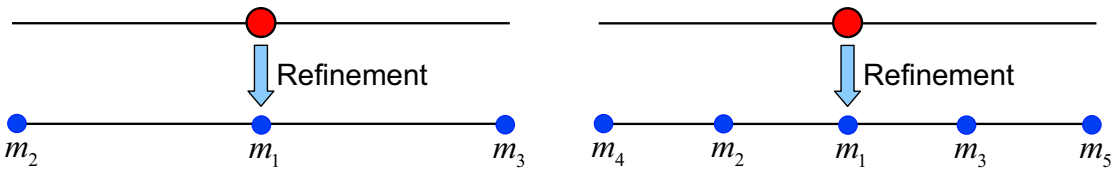


Figure 6.12: 1D refinement into 3 and 5 particles respectively

The pairs $(\varepsilon^*, \alpha^*) = (0.4, 0.4)$ and $(0.6, 0.6)$ are taken as example values for discussion. Any other pairs could have been picked as examples. $(0.4, 0.4)$ was chosen simply because it has a moderately large minimum refinement error while $(0.6, 0.6)$ has a suitably low minimum refinement error. With these values the results can easily be analysed (at least in one and two dimensions) by comparing the refined particles approximation to the original particle (the model problem particle has unit mass and smoothing length: $w(\mathbf{x}, 1)$).

In previous implementations of refinement in SPH the mass of the particle under refinement was often uniformly split amongst the daughter particles. However, the optimal mass distributions obtained from the solution of the model problem will show that often this will not minimise the density refinement error. In general non–uniform mass distributions yield the optimal mass distribution. In addition the solution to the model problem shows that refinement improves as the particle is split into a greater number of daughter particles.

For each choice of refinement parameter (ε, α) solving the model problem gives the minimum refinement error $\mathcal{E}^*[\varepsilon, \alpha]$ with the corresponding mass distribution. This is shown in Figure 6.13 and Figure 6.14 where the density refinement errors less than 0.05 are plotted against ε and α. The example refinement parameters $(\varepsilon^*, \alpha^*) = (0.4, 0.4)$ and $(0.6, 0.6)$ are identified by crosses.

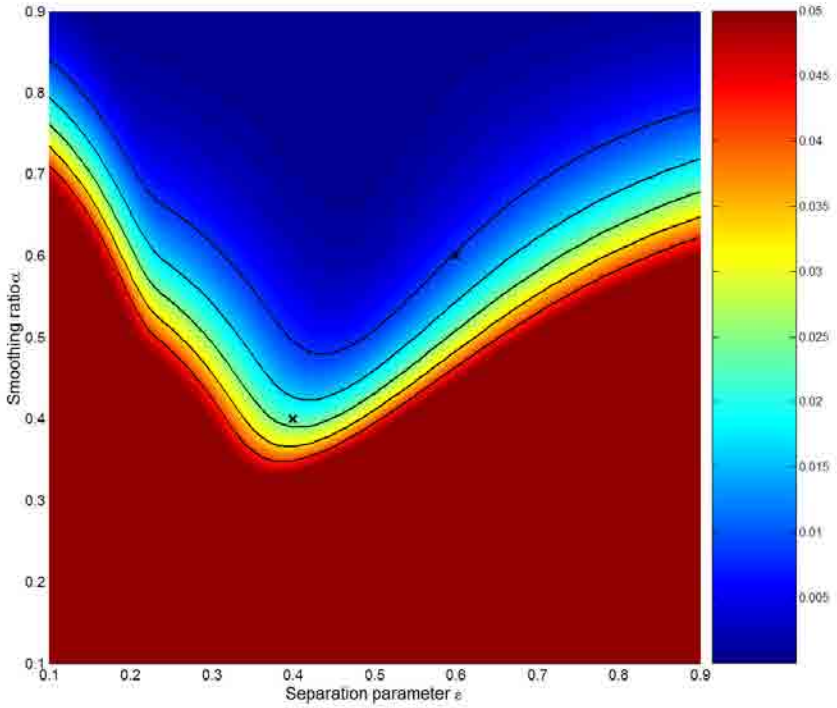


Figure 6.13: Graph of density refinement error less than 0.05 for 1D 3–particle refinement showing $(\varepsilon, \alpha) = (0.4, 0.4)$ and $(0.6, 0.6)$

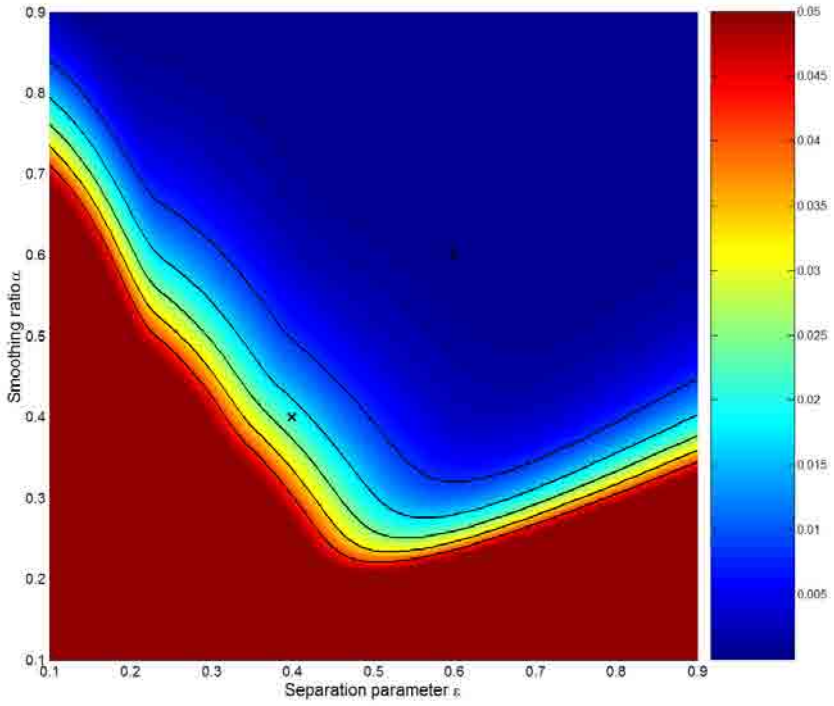


Figure 6.14: Graph of density refinement error less than 0.05 for 1D 5–particle refinement showing $(\varepsilon, \alpha) = (0.4, 0.4)$ and $(0.6, 0.6)$

From these graphs it can be seen that careful choice of refinement parameter (ε, α) is essential if the minimum density refinement error is to be sufficiently small. The benefit of refining a particle into five particles rather than three can be identified by the larger choice of refinement parameters (ε, α) resulting in sufficiently small minimum refinement errors (shown as the blue region in the graphs). Similar graphs for all the refinement patterns can be found in Appendix C including plots of the log of the refinement errors and the optimal mass distributions.

Tables 6.3 and 6.4 show the refinement errors and the corresponding optimal mass distributions calculated in one dimension for the example values of the refinement parameter. Immediately it can be seen that the optimal mass distributions are not uniform and that refinement into five particles rather than just three gives better results. In both these cases the refinement parameter $(0.6, 0.6)$ introduces smaller refinement errors into the density field compared to $(0.4, 0.4)$.

ε	α	Refinement Error	m_1	m_2, m_3
0.4	0.4	0.0223	0.443	0.2785
0.6	0.6	0.0088	0.6477	0.17615

Table 6.3: Refinement errors and optimal mass distributions for 1D 3–particle refinement

ε	α	Refinement Error	m_1	m_2, m_3	m_4, m_5
0.4	0.4	0.02077	0.353	0.078	0.244
0.6	0.6	0.000273	0.433	0.187	0.095

Table 6.4: Refinement errors and optimal mass distributions for 1D 5–particle refinement

Figures 6.15–6.18 show the resulting density profiles before and after refinement in one dimension. The effect the refinement parameter (ε, α) has on the optimal solution can be clearly seen. When the original particle is split into five daughter particles with refinement parameter $(0.6, 0.6)$ there is no significant change in density profile after refinement and the replacement daughter particle configuration is best able to approximate the contribution of the original unrefined particle.

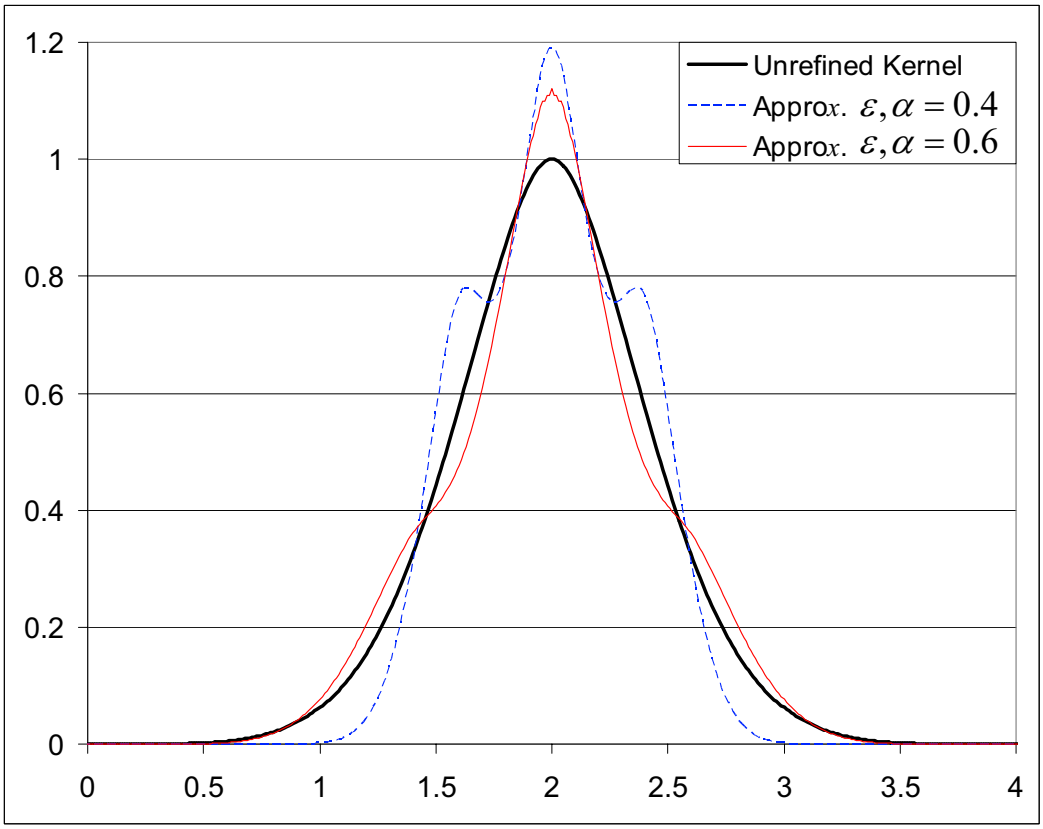


Figure 6.15: Refined particles approximation to original kernel for 1D 3–particle refinement

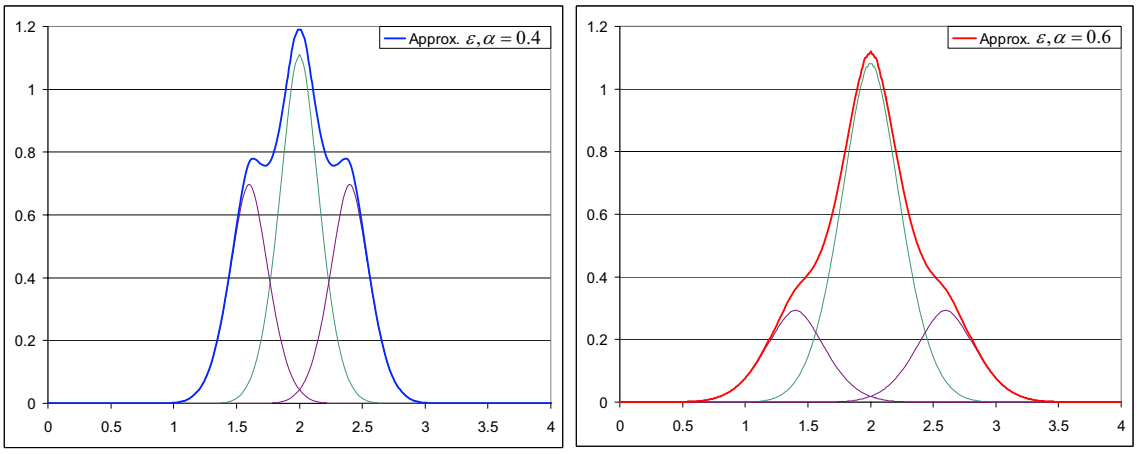


Figure 6.16: Construction of kernel approximations for 1D 3–particle refinement

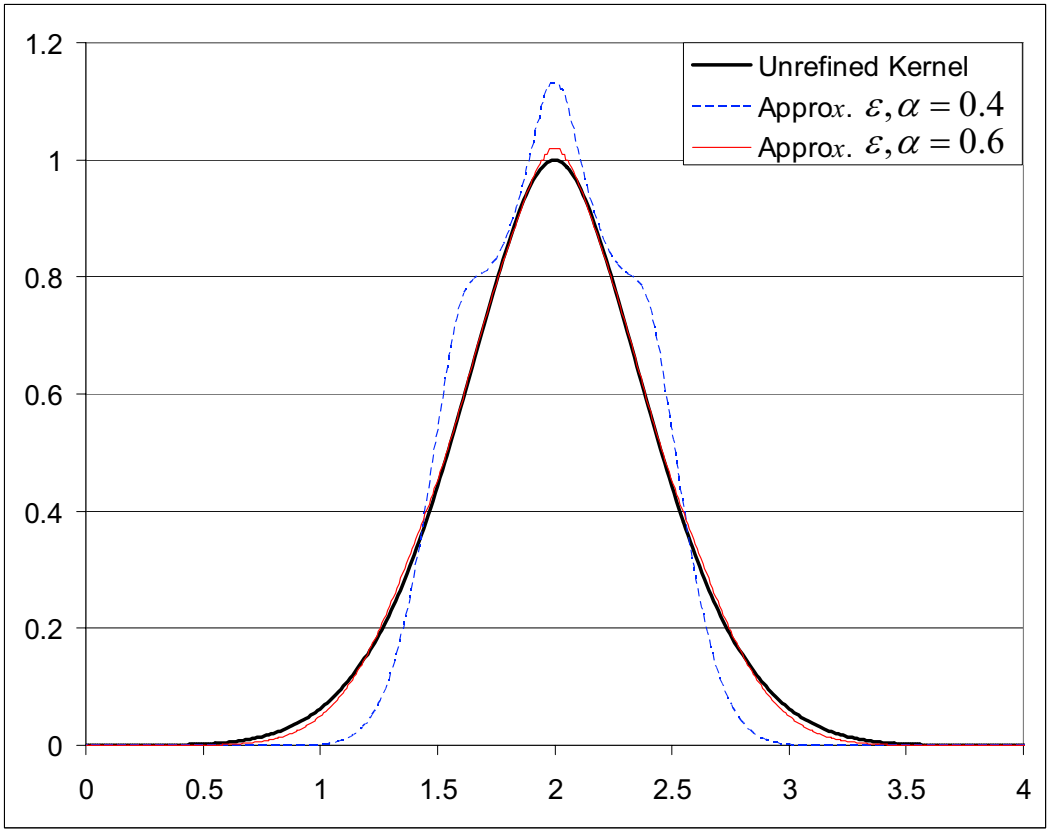


Figure 6.17: Refined particles approximation to original kernel for 1D 5–particle refinement

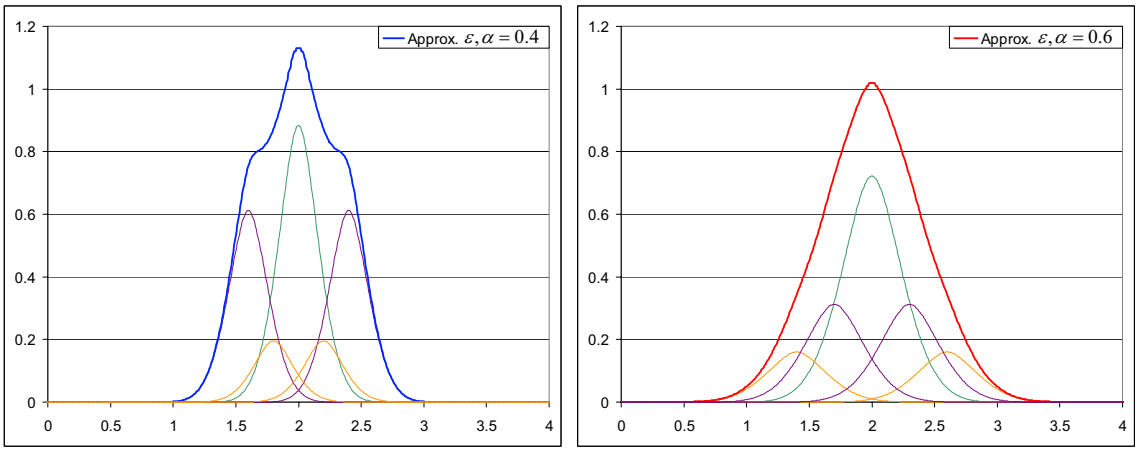


Figure 6.18: Construction of kernel approximations for 1D 5–particle refinement

6.5.1 Global refinement results

The importance of the correct choice of refinement parameter (ε, α) is highlighted when refining a group of particles. In this section groups of particles in one and two dimensions are refined and the resulting density profiles and refinement errors are compared to the initial particle configuration.

In one dimension the global density profile of a group of seven particles is calculated then the central three particles are refined into five corresponding daughter particles as shown in Figure 6.19.

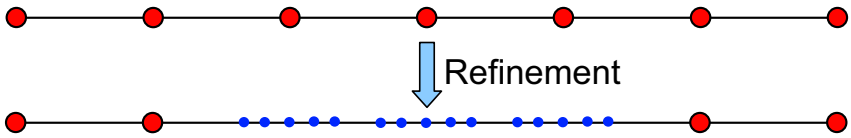


Figure 6.19: 1D global refinement example

While, in two dimensions the global density profile of a group of 11×11 particles is calculated then the central 5×5 region of particles is refined using the hexagonal refinement pattern as shown in Figure 6.20.

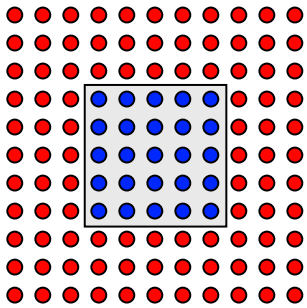


Figure 6.20: 2D global refinement example

Figure 6.21 and Table 6.5 shows the resulting global density profiles and the global refinement errors for the one dimensional example. With refinement parameter $(0.6, 0.6)$ the refinement procedure does not significantly alter the density field but when using $(0.4, 0.4)$ there is a noticeable fluctuation in density profile in the regions where the refined and unrefined particles meet.

Figure 6.22 and Figure 6.23 show the refined particles contribution to the resulting global density distribution for both choices of refinement parameter.

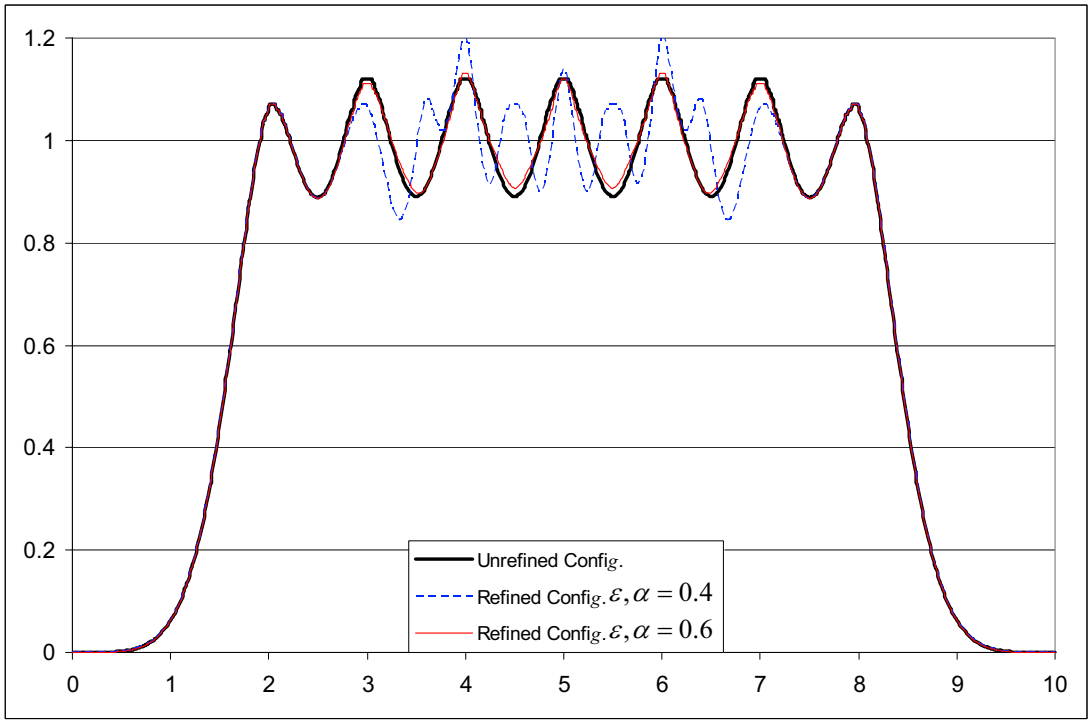


Figure 6.21: Global approximations to original density distribution

ε	α	Global Refinement Error
0.4	0.4	3.7525×10^{-2}
0.6	0.6	$6.5656 \times 10{-}4$

Table 6.5: Global refinement errors for 1D 5–particle refinement

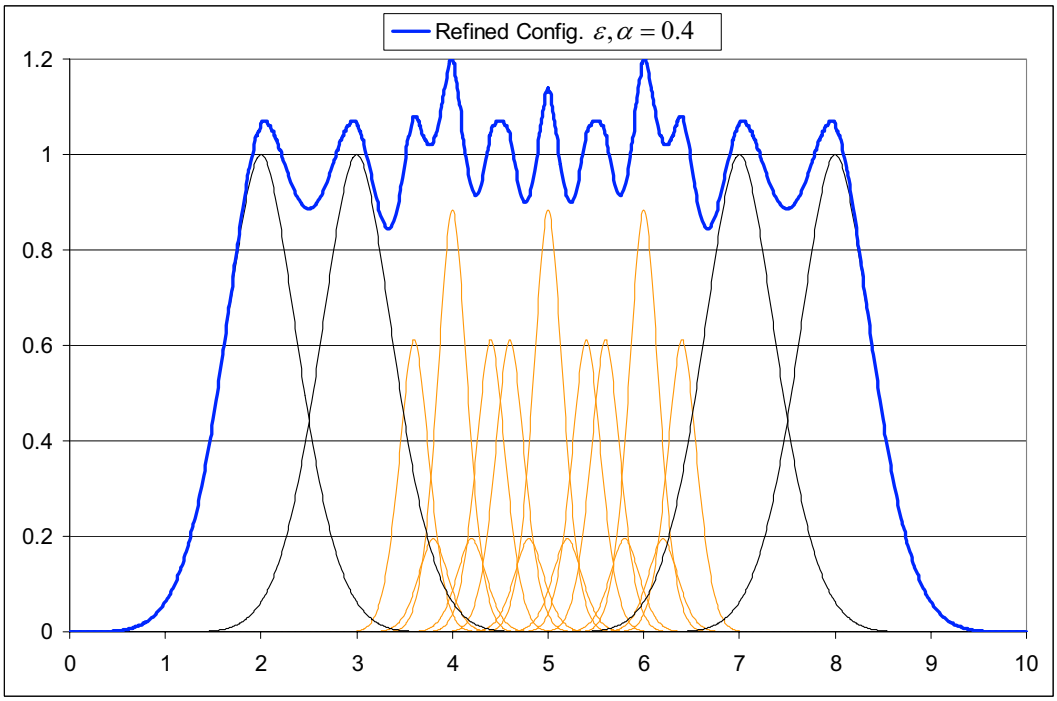


Figure 6.22: Construction of global approximation for $(\varepsilon, \alpha) = (0.4, 0.4)$

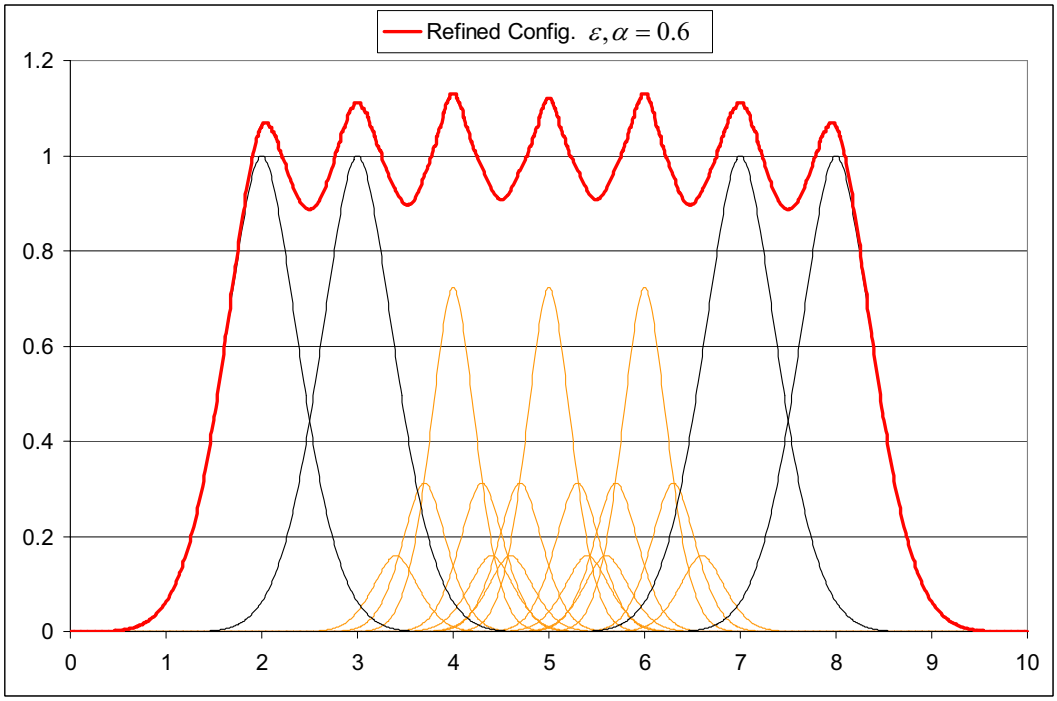


Figure 6.23: Construction of global approximation for $(\varepsilon, \alpha) = (0.6, 0.6)$

The refinement errors for the two dimensional example were found to produce similar results. These are summarised in Figure 6.24 and Table 6.6.

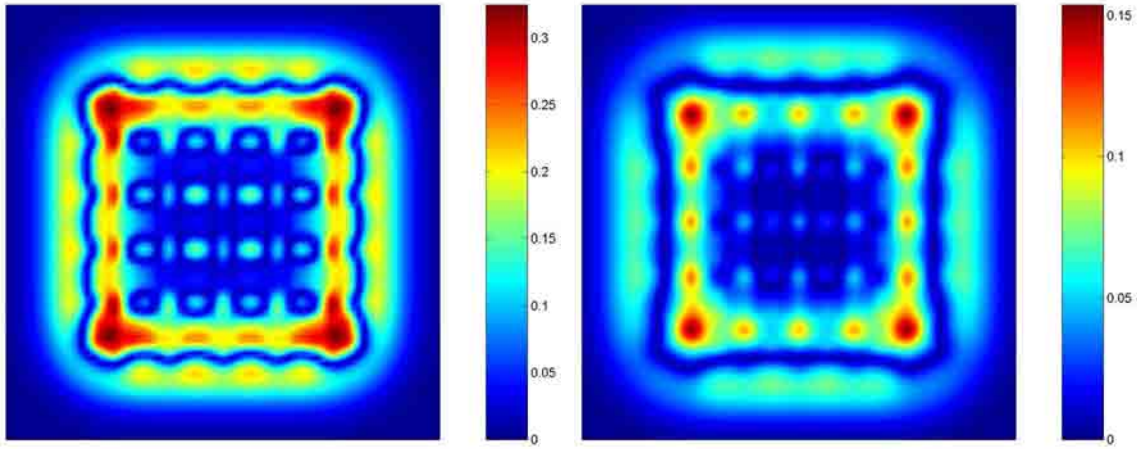


Figure 6.24: Global density refinement errors with 2D hexagonal refinement for $(\varepsilon, \alpha) = (0.4, 0.4)$ and $(0.6, 0.6)$ respectively

ε	α	Global Refinement Error
0.4	0.4	0.946
0.6	0.6	0.0911

Table 6.6: Global refinement errors for 2D hexagonal refinement

6.6 Particle velocities and conservation

Now that the density refinement errors have been minimised with conservation of mass ensured it remains to discuss how to assign the daughter particle velocities. It is not clear yet whether it is possible to minimise the error introduced into the velocity field while simultaneously conserving the kinetic energy and the linear and angular momentum of the system as given by conditions (2)–(4) in Table 6.7.

	Before Refinement	After Refinement
(1)– Mass	m_N	$\sum_b m_b$
(2)– Kinetic Energy	$\frac{1}{2} m_N \mathbf{v}_N \cdot \mathbf{v}_N$	$\frac{1}{2} \sum_b m_b \mathbf{v}_b \cdot \mathbf{v}_b$
(3)– Linear Momentum	$m_N \mathbf{v}_N$	$\sum_b m_b \mathbf{v}_b$
(4)– Angular Momentum	$\mathbf{x}_N \times m_N \mathbf{v}_N$	$\sum_b \mathbf{x}_b \times m_b \mathbf{v}_b$

Table 6.7: Global Conservation Properties (1)–(4)

A given set of constraints may not even be consistent in the sense that there may only be one velocity configuration (or worse still no velocity configuration) that satisfies them all simultaneously. Then in both cases attempting to find an optimum solution would be in vein since there would be no 'best' solution to find.

Traditionally, new particles are given the interpolated velocity from the original unrefined particle configuration (see Figure 6.25). Namely,

$$\mathbf{v}_b = \frac{\sum_i m_i \mathbf{v}_i w_i \left(\mathbf{x}_b, h_i\right)}{\sum_i m_i w_i \left(\mathbf{x}_b, h_i\right)} \tag{6.25}$$

where the summation over i is only taken over the neighbours of particle b from the unrefined particle configuration. This has the advantage that the resulting interpolated velocities follow that of the underlying flow. However, they will not in general preserve the kinetic energy, nor the linear and angular momentum of the system.

One such choice of daughter particle velocities that does conserve these properties is to assign all daughter particles with the original particle velocity $\mathbf{v}_N$ that they replace. In other words setting $\mathbf{v}_b = \mathbf{v}_N$ for each daughter particle b.

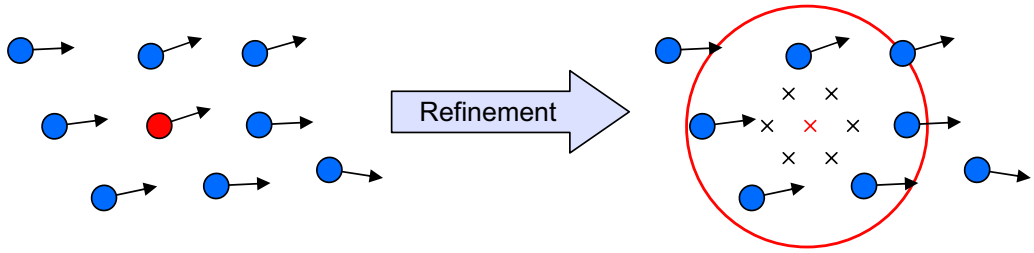


Figure 6.25: Interpolated velocity refinement using initial particle distribution

The question remains, is there a way in which a more representative set of velocities can be chosen such that these global properties are conserved?

In Section 6.6.1 it is shown that the answer to this question is in fact no and this is the unique conservative daughter particle velocity configuration. In Section 6.6.2 the corresponding velocity refinement errors associated to this velocity configuration are calculated.

6.6.1 Inconsistent velocity constraints

In order to see that the daughter particle velocity configuration defined by setting $\mathbf{v}_b = \mathbf{v}_N$ for each daughter particle b is the unique conservative solution satisfying (2)–(4) in Table 6.7 it is necessary to consider only the particle equations for conservation of kinetic energy and linear momentum given by

$$\mathbf{v}_N \cdot \mathbf{v}_N = \sum_{b=1}^{M} \lambda_b \mathbf{v}_b \cdot \mathbf{v}_b \quad \text{and} \quad \mathbf{v}_N = \sum_{b=1}^{M} \lambda_b \mathbf{v}_b. \tag{6.26}$$

Here the λ_i's are given from the analysis in previous section which satisfy $\sum_{b=1}^{M} \lambda_b = 1$. Suppose for now that the daughter particles are constrained to move in the same direction as the original particle but with varying magnitude. In this case

$$\mathbf{v}_i = \mu_i \mathbf{v}_N \quad \text{where} \quad \mu_i > 0 \quad \text{for each } i. \tag{6.27}$$

Substituting these velocities into equation (6.26) gives two scalar equations in μ

$$f(\boldsymbol{\mu}) = \sum_{b=1}^{M} \lambda_b \mu_b^2 - 1 \quad \text{and} \quad g(\boldsymbol{\mu}) = \sum_{b=1}^{M} \lambda_b \mu_b - 1. \tag{6.28}$$

The first of which defines an ellipsoid in the M-dimensional space $\boldsymbol{\mu}$, while the second equation defines a plane in the same space as shown in Figure 6.26.

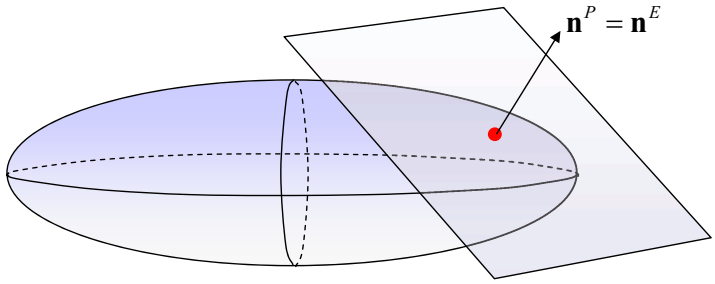


Figure 6.26: The unique solution $\boldsymbol{\mu}^*$ satisfying $f(\boldsymbol{\mu}) = 0$ and $g(\boldsymbol{\mu}) = 0$

As noted above $\boldsymbol{\mu}^* = [1, \cdots, 1]^T$ satisfies both $f(\boldsymbol{\mu}^*) = 0$ and $g(\boldsymbol{\mu}^*) = 0$. However, it still needs to be verified that there is in fact more than one solution satisfying the constraint equations above.

The normal to the plane is given by $\mathbf{n}_p = [\lambda_1, \cdots, \lambda_M]^T$. While the normal to the ellipsoid at a point $\boldsymbol{\mu}$ is given by

$$\mathbf{n}_e = [\text{grad}(f)](\boldsymbol{\mu}) = \begin{pmatrix} 2\lambda_1\mu_1 \\ \vdots \\ 2\lambda_M\mu_M \end{pmatrix}. \tag{6.29}$$

Therefore, at $\boldsymbol{\mu}^* = [1, \cdots, 1]^T$ the normal is given by $\mathbf{n}_e = [2\lambda_1, \cdots, 2\lambda_M]^T$ and the plane is tangential to the ellipsoid at $\boldsymbol{\mu}^*$. Consequently, $\boldsymbol{\mu}^*$ is the unique solution satisfying equation (6.28).

What has been shown above is that the only daughter particle velocity configuration (of the form $\mathbf{v}_i = \mu_i \mathbf{v}_N$) that conserves both kinetic energy and linear momentum is for all the daughter particles to move with the same velocity as the original particle $\mathbf{v}_b = \mathbf{v}_N$. In addition, when the refinement pattern is symmetric about the original particle position the system will also conserve the angular momentum of the system.

The above result can now be generalised to prove that this is in fact true when any set of daughter particle velocities $\mathbf{v}_b$ is considered.

Without loss of generality suppose that in addition to $\mathbf{v}_N \neq \mathbf{0}$ that each component v_N^i of $\mathbf{v}_N$ is non-zero in other words that $v_N^i \neq 0$. If this is not the case one (or two) components of $\mathbf{v}_N$ is equal to zero when written in terms of the original basis

$\{\mathbf{a}_1,\ \mathbf{a}_2,\ \mathbf{a}_3\}$. However the choice of basis is arbitrary and it is a simple matter to select an alternative basis $\{\mathbf{b}_1,\ \mathbf{b}_2,\ \mathbf{b}_3\}$ such that $v_N^i \neq 0$ for each i.

With a suitable choice of basis the refined velocities can be defined in terms of the original particle velocity $\mathbf{v}_N$ by

$$\mathbf{v}_i = \mathbf{\Gamma}_i \mathbf{v}_N \qquad \text{where} \qquad \mathbf{\Gamma}_i = \begin{bmatrix} \alpha_i & 0 & 0 \\ 0 & \beta_i & 0 \\ 0 & 0 & \gamma_i \end{bmatrix} \quad \alpha_i, \beta_i, \gamma_i \in \mathbb{R}. \tag{6.30}$$

and $\alpha_i,\ \beta_i,\ \gamma_i$ are now the independent variables controlling the daughter particle velocities. As before substituting these terms into the conservation equations of kinetic energy and linear momentum gives

$$\begin{aligned} \mathbf{v}_N^T \mathbf{v}_N = \sum_{b=1}^{M} \lambda_b \mathbf{v}_N^T \mathbf{\Gamma}_b^T \mathbf{\Gamma}_b \mathbf{v}_N = \mathbf{v}_N^T \left(\sum_{b=1}^{M} \lambda_b \mathbf{\Gamma}_b^T \mathbf{\Gamma}_b \right) \mathbf{v}_N, \\ \mathbf{v}_N = \left(\sum_{b=1}^{M} \lambda_b \mathbf{\Gamma}_b \right) \mathbf{v}_N. \end{aligned} \tag{6.31}$$

Equating the coefficients and noting that $\mathbf{\Gamma}^T = \mathbf{\Gamma}$ yields the following pair of matrix conditions

$$\mathbf{I}_{3\times 3} = \sum_{b=1}^{M} \lambda_b \mathbf{\Gamma}_b^2 \qquad \text{and} \qquad \mathbf{I}_{3\times 3} = \sum_{b=1}^{M} \lambda_b \mathbf{\Gamma}_b. \tag{6.32}$$

Since $\mathbf{I}_{3\times 3}$, $\mathbf{\Gamma}_b$, and $\mathbf{\Gamma}_b^2$ terms are symmetric matrices the expressions given above represent three independent pairs of equations in α, β and γ respectively. Each of these pairs is identical to the pair given in equation (6.28) and so share the same unique solution.

In conclusion, it has been shown that there is only one possible fully conservative velocity refinement strategy and that is to set all the daughter particle velocities equal to that of the unrefined particle they replace.

The decision is then between using the interpolated daughter particle velocities and so preserving the topological properties of the flow but loosing conservation. Or to move all refined particles with the unrefined particle velocity to fully enforce conservation, while perhaps loosing some of the topology of the velocity field.

6.6.2 Velocity refinement error

Using the SPH approximations for linear momentum and density shown respectively below

$$\langle \rho_a \mathbf{v}_a \rangle = \sum_b V_b \left(\rho_b \mathbf{v}_b \right) w_b \left(\mathbf{x}_a, h_b \right) \quad \text{and} \quad \langle \rho_a \rangle = \sum_b m_b w_b \left(\mathbf{x}_a, h_b \right) \tag{6.33}$$

and using the approximation $\langle f \rangle \langle g \rangle = \langle fg \rangle$ the SPH form for the velocity is typically given by

$$\langle \mathbf{v}_a \rangle = \frac{\sum\limits_b m_b \mathbf{v}_b w_b \left(\mathbf{x}_a, h_b \right)}{\sum\limits_b m_b w_b \left(\mathbf{x}_a, h_b \right)} \ . \tag{6.34}$$

As before suppose that the N^{th} particle is refined into M daughter particles and that their masses $m_b = \lambda_b m_N$ are assumed to be those obtained from Section 6.4 which minimise the density refinement error.

The velocity field before refinement is given by

$$\mathbf{v} \left(\mathbf{x} \right) = \frac{\sum\limits_{a=1}^{N} m_a \mathbf{v}_a w_a \left(\mathbf{x}, h_a \right)}{\sum\limits_{a=1}^{N} m_a w_a \left(\mathbf{x}, h_a \right)} \tag{6.35}$$

and the velocity field after refinement is given by

$$\mathbf{v}^* \left(\mathbf{x} \right) = \frac{\sum\limits_{a=1}^{N-1} m_a \mathbf{v}_a w_a \left(\mathbf{x}, h_a \right) + \sum\limits_{b=1}^{M} m_b \mathbf{v}_b w_b \left(\mathbf{x}, h_b \right)}{\sum\limits_{a=1}^{N-1} m_a w_a \left(\mathbf{x}, h_a \right) + \sum\limits_{b=1}^{M} m_b w_b \left(\mathbf{x}, h_b \right)}. \tag{6.36}$$

Note that the difference between the denominators of $\mathbf{v}$ and $\mathbf{v}^*$ has already been minimised in the previous section when finding the optimal daughter particle mass distribution. Therefore, the problem can be simplified by only minimising the difference between the numerators $\mathbf{N} \left(\mathbf{x} \right)$ and $\mathbf{N}^* \left(\mathbf{x} \right)$ of equations above

$$\mathbf{N} \left(\mathbf{x} \right) = \sum_{a=1}^{N} m_a \mathbf{v}_a w_a \left(\mathbf{x}, h_a \right) \tag{6.37}$$

and

$$\mathbf{N}^* \left(\mathbf{x} \right) = \sum_{a=1}^{N-1} m_a \mathbf{v}_a w_a \left(\mathbf{x}, h_a \right) + \sum_{b=1}^{M} m_b \mathbf{v}_b w_b \left(\mathbf{x}, h_b \right). \tag{6.38}$$

The *local velocity refinement error* at any point $\mathbf{x}$ is then defined to be the function $L : \mathbb{R}^{3M} \times \mathbb{R}^3 \to C(\mathbb{R}^3)$,

$$\begin{aligned} L(\mathbf{v}_1, \cdots, \mathbf{v}_M)(\mathbf{x}) &= (\mathbf{N}(\mathbf{x}) - \mathbf{N}^*(\mathbf{x})) \cdot (\mathbf{N}(\mathbf{x}) - \mathbf{N}^*(\mathbf{x})) \\ &= \|\mathbf{N}(\mathbf{x}) - \mathbf{N}^*(\mathbf{x})\|^2 \qquad \forall \mathbf{x} \in \Omega. \end{aligned} \tag{6.39}$$

From the local velocity error the *global velocity refinement error* is defined to be the function $\mathcal{L} : \mathbb{R}^{3M} \to \mathbb{R}^+$,

$$0 \leqslant \mathcal{L}(\mathbf{v}_1, \cdots, \mathbf{v}_M) = \int_\Omega L(\mathbf{v}_1, \cdots, \mathbf{v}_M)(\mathbf{x})\, d\mathbf{x}. \tag{6.40}$$

Expanding $\mathcal{L}(\mathbf{v}_1, \cdots, \mathbf{v}_M)$ as before gives

$$\mathcal{L}(\mathbf{v}_1, \cdots, \mathbf{v}_M) = m_N^2 \left[\begin{array}{c} (\mathbf{v}_N \cdot \mathbf{v}_N) \displaystyle\int_\Omega w_N^2(\mathbf{x}, h_N)\, d\mathbf{x} \\ -2 \displaystyle\sum_{a=1}^{M} \lambda_a (\mathbf{v}_a \cdot \mathbf{v}_N) \int_\Omega w_N(\mathbf{x}, h_N)\, w_a(\mathbf{x}, h_a)\, d\mathbf{x} \\ + \displaystyle\sum_{a,b=1}^{M} \lambda_a \lambda_b (\mathbf{v}_a \cdot \mathbf{v}_b) \int_\Omega w_a(\mathbf{x}, h_a)\, w_b(\mathbf{x}, h_b)\, d\mathbf{x} \end{array} \right] \tag{6.41}$$

where the λ_a, $\lambda_a \lambda_b$ terms are now constants taken from the solution to the previous density refinement calculations and the independent variables are now the velocities of the daughter particles.

The global velocity refinement error $\mathcal{L}$ can be written more succinctly in vector notation where $\mathbf{V} = [\mathbf{v}_1, \cdots, \mathbf{v}_M]^T$ is the vector of daughter particle velocities

$$\mathcal{L}(\mathbf{v}_1, \cdots, \mathbf{v}_M) = \frac{m_N^2}{h_N^3} \left[\tilde{C} - 2\mathbf{V}^T \tilde{\mathbf{B}} + \mathbf{V}^T \tilde{\mathbf{A}} \mathbf{V} \right]. \tag{6.42}$$

The constant term $\tilde{C}$, and components of vector term $\tilde{\mathbf{B}} \in \mathbb{R}^{3M}$ and matrix term $\tilde{\mathbf{A}} \in \mathbb{R}^{3M} \times \mathbb{R}^{3M}$ have been written in non–dimensional form and are given by

$$\begin{aligned} \tilde{C} &= (\mathbf{v}_N \cdot \mathbf{v}_N) \int_\Omega w_N^2(\hat{\mathbf{x}}, 1)\, d\hat{\mathbf{x}}, \\ \tilde{B}_i &= \left(\lambda_i \int_\Omega w_N(\hat{\mathbf{x}}, 1)\, w_i(\hat{\mathbf{x}}, \alpha)\, d\hat{\mathbf{x}} \right) \mathbf{v}_N, \qquad \tilde{\mathbf{A}} = \left[\begin{array}{ccc} \tilde{\mathbf{A}}_{11} & \cdots & \tilde{\mathbf{A}}_{1M} \\ \vdots & \ddots & \vdots \\ \tilde{\mathbf{A}}_{M1} & \cdots & \tilde{\mathbf{A}}_{MM} \end{array} \right] \\ \tilde{\mathbf{A}}_{ij} &= \left[\frac{\lambda_i \lambda_j}{\alpha^3} \int_\Omega w_i(\hat{\mathbf{x}}, 1)\, w_j(\hat{\mathbf{x}}, 1)\, d\hat{\mathbf{x}} \right] \mathbf{I}_{3\times 3}, \end{aligned} \tag{6.43}$$

It should be noted that vector term $\tilde{\mathbf{B}}$ and the constant term $\tilde{C}$ when in this form are still written in terms of the initial unrefined particle velocity $\mathbf{v}_N$. In order to derive a model problem for minimising the velocity refinement error $\mathcal{L}$ it would first need to be written independently of the initial unrefined velocity $\mathbf{v}_N$.

However, it is simple to calculate the resulting velocity refinement error for the unique conservative velocity configuration calculated in Section 6.6.1. In this case $\mathbf{v}_b = \mathbf{v}_N$ for each daughter particle b and equation (6.41) can be written in non–dimensional form as

$$\mathcal{L} = (\mathbf{v}_N \cdot \mathbf{v}_N) \frac{m_N^2}{h_N^3} \left(\begin{array}{c} \int\limits_\Omega w_N^2 (\hat{\mathbf{x}}, 1)\, d\hat{\mathbf{x}} - 2 \sum\limits_{a=1}^{M} \lambda_a \int\limits_\Omega w_N (\hat{\mathbf{x}}, 1)\, w_a (\hat{\mathbf{x}}, \alpha)\, d\hat{\mathbf{x}} \\ + \sum\limits_{a,b=1}^{M} \frac{\lambda_a \lambda_b}{\alpha^3} \int\limits_\Omega w_a (\hat{\mathbf{x}}, 1)\, w_b (\hat{\mathbf{x}}, 1)\, d\hat{\mathbf{x}} \end{array} \right).$$

Written in vector form and in terms of the daughter particle masses $\boldsymbol{\lambda}$ gives

$$\begin{aligned} \mathcal{L} &= (\mathbf{v}_N \cdot \mathbf{v}_N) \frac{m_N^2}{h_N^3} \left(\bar{C} - 2\, \boldsymbol{\lambda}^T \bar{\mathbf{b}} + \boldsymbol{\lambda}^T \bar{\mathbf{A}}\, \boldsymbol{\lambda} \right) \\ &= \|\mathbf{v}_N\|^2\ \mathcal{E}\left[\varepsilon, \alpha\right](\boldsymbol{\lambda}) \\ &= \|\mathbf{v}_N\|^2\ \frac{m_N^2}{h_N^3}\ \mathcal{E}^* \end{aligned} \tag{6.44}$$

where $\mathcal{E}^*$ is the solution of the previous model problem for minimising the density refinement error with corresponding refinement parameter (ε, α).

Therefore, in the case of the fully conservative velocity distribution the global velocity refinement error can be calculated and is found to be proportional to global density refinement error calculated in Section 6.4.

6.7 Concluding remarks

In this chapter a general refinement strategy for SPH simulations has been proposed. The algorithm is simple to implement in any number of dimensions and can be applied irrespective of the refinement criteria used to identify the candidate particles.

The global density refinement error has been defined as a measure of the error introduced to the density field caused by the particle refinement. For a given refinement pattern the resulting density refinement errors can be studied and minimised over a variety of refinement parameters (ε, α). This is achieved through the solution of the model problem from which the optimal daughter particle masses are obtained. In general these optimal mass distributions for refinement are non-uniform and each daughter particle has its own distinct mass.

Candidate refinement parameters $(\varepsilon^*, \alpha^*)$ that introduce sufficiently small density refinement errors can then be identified for use in subsequent SPH simulations. This process conserves the mass of the system. The global density refinement errors for several one, two and three dimensional refinement patterns have been calculated, candidate refinement parameters $(\varepsilon^*, \alpha^*)$ identified and the resulting refined density profiles plotted. These optimal parameters will be applied to several fluid flow examples in Chapter 7.

Conservation of kinetic energy and linear and angular momentum are harder to establish for general refinement algorithms. It has been shown that the unique fully conservative velocity distribution for the daughter particles is to move them with the velocity of the unrefined particle that they replace. Consequently, in all the examples in Chapter 7 daughter particles will be assigned the velocity of the particle that they replace.

The global velocity refinement error has also been defined as a measure of the error introduced to the velocity field caused by the particle refinement. In general this has been shown to be dependent upon the velocity of the initial unrefined particle. However, this expression simplifies for the unique conservative velocity configuration and the global velocity refinement error is found to be proportional to global density refinement error obtained from the solution of the model problem.

Chapter 7

Refinement simulations

7.1 Introduction

In this chapter the dynamic particle refinement strategy presented in Chapter 6 is successfully implemented into the existing SPH framework.

The essential ingredients of the kernel and gradient corrections from Chapter 3, the variational SPH formulation for particles with non-uniform smoothing lengths and masses as given in Chapter 4 and the new variational boundary contact force from Chapter 5 are all required to create a single flexible SPH code with which to model free surface flow problems which incorporate dynamic particle refinement.

In the previous chapter the global density and velocity refinement errors have been defined and studied. However, it has not been ascertained whether such measures of error can be reliably used as a guide to identify optimal refinement parameters which will result in accurate and stable variable resolution simulations in SPH.

In order to validate the refinement procedure and boundary contact force implementation four simple flow simulations in two dimensions are presented. The hexagonal refinement pattern, in a fixed orientation, with the corresponding refinement parameter (ε, α) is used to distribute the daughter particles in all of the examples.

The accuracy of the refinement procedure is first investigated using the Couette and Poiseuille flows for which analytic solutions are available. The simulations are run both with and without refinement. In the refined simulations the particles in the central region of the flows are refined according to the preceding theory. The ve-

locity profiles of the refined flows are then compared to those of the unrefined flows and to their corresponding analytic solutions.

The refined Couette and Poiseuille flows are examples of static refinement tests. These are simulations whereby the particles under refinement are either initially stationary or moving with a constant velocity. Consequently, no errors are introduced due to velocity refinement and the only error comes from the redistribution of particle masses.

It should be noted that the refined Couette and Poiseuille flows are challenging simulations for a variable resolution SPH code. This is because particles will remain on the boundary between the refined and unrefined regions throughout the duration of the simulations. Consequently, any errors generated by the refinement procedure are more likely to propagate through the domain.

The second set of simulations consist of two more complex flows used to test the dynamic refinement implementation in conjunction with the new boundary forces. The first example models the flow separation through a funnel while the second models an emptying tank with two small outlets on the side wall.

In these examples designated refinement zones are used as the particle refinement criteria. Upon entering these regions a particle is replaced with its corresponding set of daughter particles. In these dynamic refinement examples the daughter particle velocities are chosen to be equal to the velocity of the particle they replace and as such the refinement procedure is fully conservative.

7.2 Static refinement simulations

7.2.1 Couette Flow

The Couette flow models the motion of a fluid between two parallel, infinitely long horizontal plates. The fluid and plates are initially stationary. Instantaneously the top plate is moved with a fixed velocity and the fluid motion is generated.

Computational model

The problem domain is a rectangular region 0.0005m × 0.001m modelled with 20 × 40 = 800 fluid particles as shown in Figure 7.1. The upper plate velocity is $\|\mathbf{v}_U\| = 2.5 \times 10^{-5}\text{ms}^{-1}$ in the horizontal direction corresponding to a Reynolds Number of 2.5×10^{-2}. The material density is $\rho_0 = 1000\text{kgm}^{-3}$, the coefficient of kinetic viscosity is $\nu = 10^{-6}\text{m}^2\text{s}^{-1}$ and a constant timestep of 10^{-4}s is used throughout the simulation.

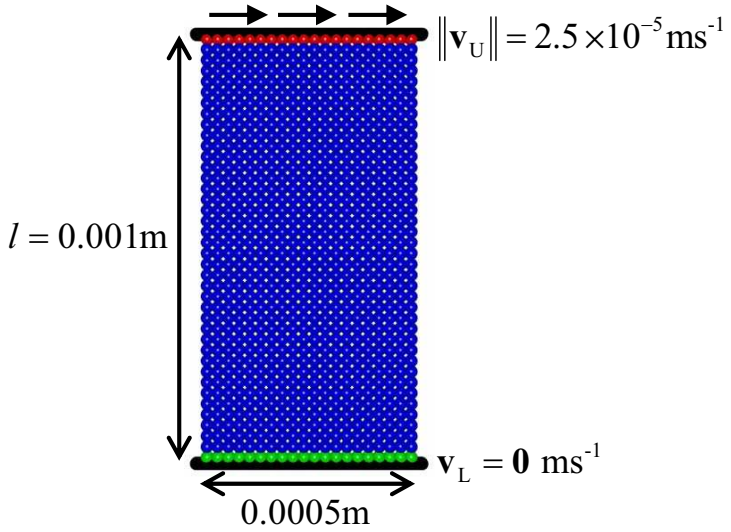


Figure 7.1: Initial particle configuration for the Couette flow

The artificial bulk modulus for equation of state (2.81) is given by equation (2.83) as

$$P_0 = \frac{\left(\frac{\|\mathbf{v}_{\max}\|}{\text{M}}\right)^2 \rho}{\gamma} = \frac{1000\left(\frac{2.5\times10^{-5}}{0.1}\right)^2}{7} = 8.93 \times 10^{-6} \tag{7.1}$$

For a Mach number equal to 0.1, this gives the sound speed for a given particle a as

$$c_a = \sqrt{\frac{\gamma P_0}{\rho_a}} = \sqrt{\frac{7 \times 8.93 \times 10^{-6}}{\rho_a}}. \tag{7.2}$$

The essential boundary conditions of the upper and lower plates are enforced by first initialising the velocity of the top row of particles to $2.5 \times 10^{-5}\text{ms}^{-1}$ and the velocity of the bottom row of particles to zero. These particles are then moved with a constant velocity by setting their accelerations to zero at the end of each timestep. The remaining particles making up the flow are initially stationary with their accelerations calculated as normal.

Analytic solution

The series solution to the Couette flow is give by the equation [101]

$$v_x(y,t) = \frac{|\mathbf{v}_U|}{l} y + \sum_{n=1}^{\infty} \frac{2|\mathbf{v}_U|}{n\pi} (-1)^n \sin\left(\frac{n\pi}{l} y\right) \exp\left(-\upsilon \frac{n^2\pi^2}{l^2} t\right) \tag{7.3}$$

$v_x(y,t)$ is the horizontal velocity at a given distance y across the flow and time t.

Periodic boundaries

A periodic boundary condition in the direction of the flow is used to model the infinitely long channel. Particle neighbours are modified to reflect the periodic boundary with particles at one periodic boundary contributing to particles at the other periodic boundary as shown in Figure 7.2.

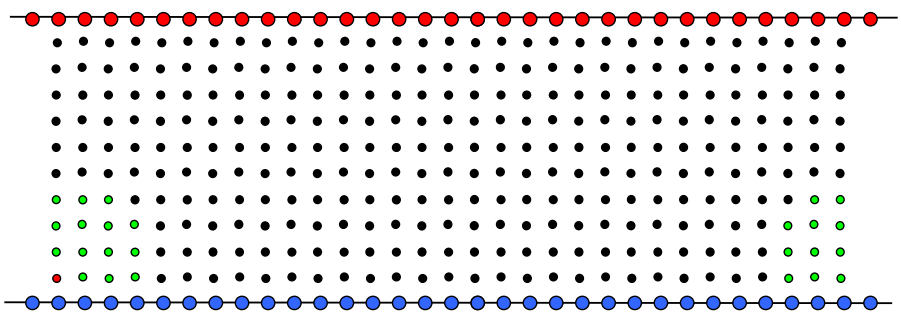


Figure 7.2: Neighbour search with periodic boundary

7.2.2 Poiseuille flow

The Poiseuille flow models the motion of a fluid between two stationary infinitely long parallel plates. From rest the fluid is subject to a constant driving force F (which may be attributed to a pressure difference or simply an external force), and the fluid flow between the two plates is generated.

Computational model

The geometry of the computational model is exactly the same as for the Couette flow except that the essential boundary conditions for the Poiseuille flow dictate that the upper and lower rows of particles are stationary as shown in Figure 7.3.

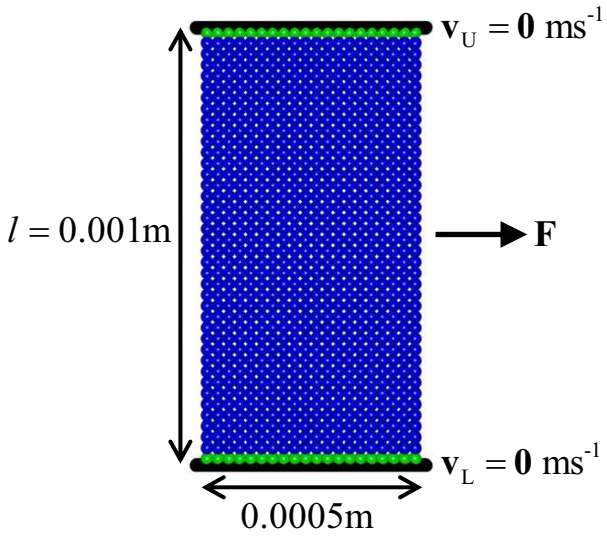


Figure 7.3: Initial particle configuration for the Poiseuille flow

All particles are subject to a constant driving force of $F = 2 \times 10^{-4}$ms^{-2} in the horizontal direction. This corresponds to a peak velocity of 2.5×10^{-5}ms^{-1} and a Reynolds number of 2.5×10^{-2}.

Analytic solution

The series solution to the Poiseuille flow is given by the equation [101]

$$v_x(y,t) = \frac{F}{2\upsilon} y(y-l) + \sum_{n=0}^{\infty} \frac{4Fl^2}{\upsilon\pi^3(2n+1)^3} \sin\left(\frac{\pi y}{l}(2n+1)\right) \exp\left(-\upsilon\frac{(2n+1)^2\pi^2}{l^2}t\right) \tag{7.4}$$

$v_x(y,t)$ is the horizontal velocity at a given distance y across the flow and time t.

7.2.3 Unrefined Couette and Poiseuille results

Figure 7.4 and Figure 7.6 show the velocity of the unrefined Couette and Poiseuille flows at several stages of the simulation. Figure 7.5 and Figure 7.7 plot the resulting velocity profiles of the unrefined Couette and Poiseuille flows. The solid lines are the velocity profiles obtained from the series solutions given by equation (7.3) and equation (7.4) respectively. First order consistency of the gradient terms for both the Couette and Poiseuille flows were enforced using the mixed kernel and gradient corrections. Both flows were also run with the additional higher order Hessian term for the gradient of the kernel function as given by equation (3.60). However, in both cases the improvement to the velocity profile was found to be negligible.

The unrefined Couette flow results are in good agreement with the analytical solution and accurately predict the resulting linear steady state flow. On average the error in the steady state velocity profile was less than 1%. However, the Poiseuille flow results are not quite as accurate. The early stages of the flow are in good agreement with the analytical solution but gradually the flow velocities fall short of the maximum steady state velocity of $2.5 \times 10^{-5}\text{ms}^{-1}$ as the simulation progresses. On average the error in the steady state velocity profile was 4%.

Several factors can be identified as possible causes for the failure of the Poiseuille flow to reach the correct steady state. The boundary conditions implemented were very simplistic and no addition image particles were used to improve the interpolation in the vicinity of the boundary. The steady state of the Poiseuille flow is parabolic and the gradients used were only linearly corrected. The loss of energy in the flow is most likely to be due to the artificial compressibility of the formulation and an insufficient particle resolution across the flow. In the next section it is shown that the Poiseuille flow simulation improves with the addition of more particles across the flow.

More accurate results have been obtained for the Poiseuille flow by Morris [101] and Sigalotti [117] who use SPH to model the dynamic pressure $p_d = p_t - p_h$, where p_t is total pressure and p_h hydrostatic pressure. This generates an additional body force in the expression for the pressure gradient and can improve the accuracy for problems where the local variation in the pressure gradients are small.

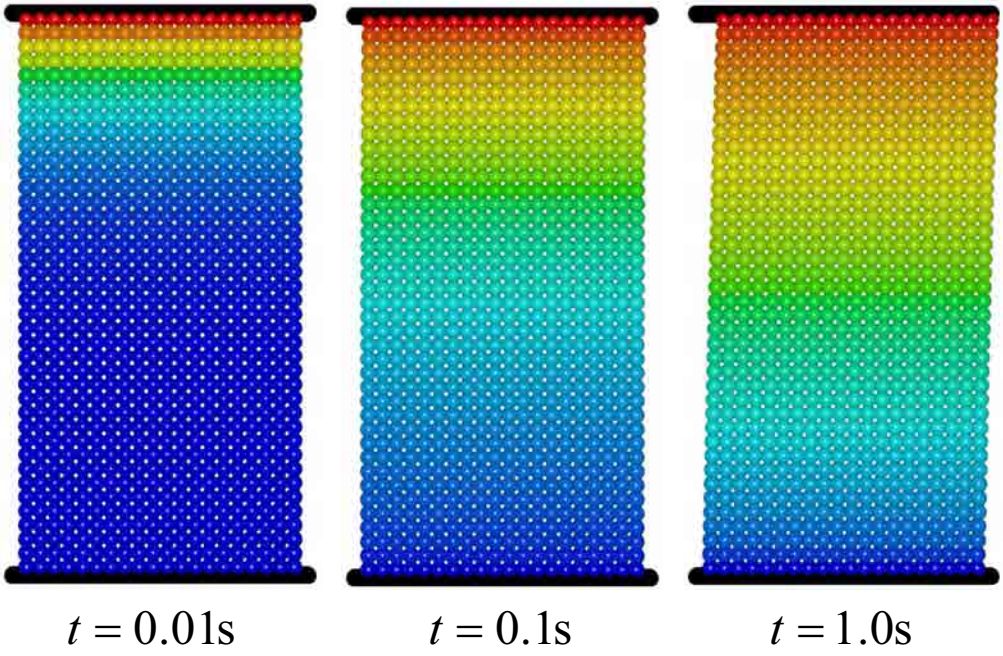


Figure 7.4: Couette flow velocities without refinement

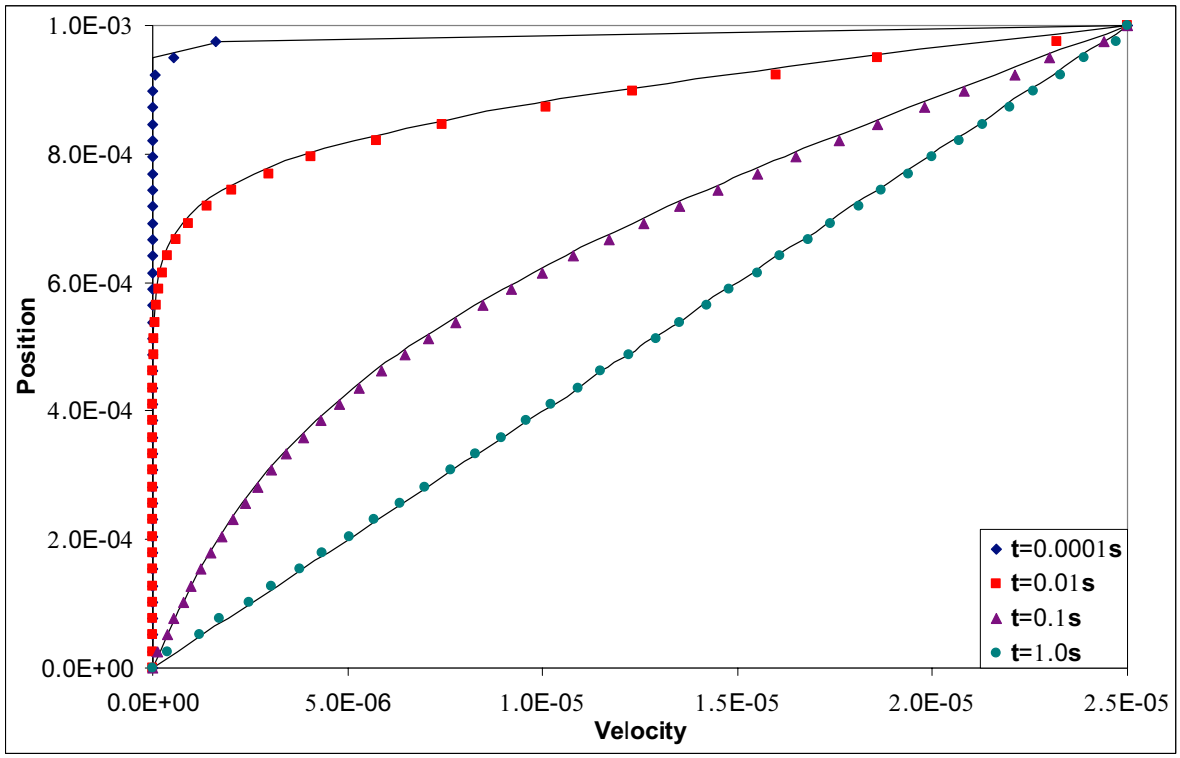


Figure 7.5: Velocity profiles of the Couette flow without refinement

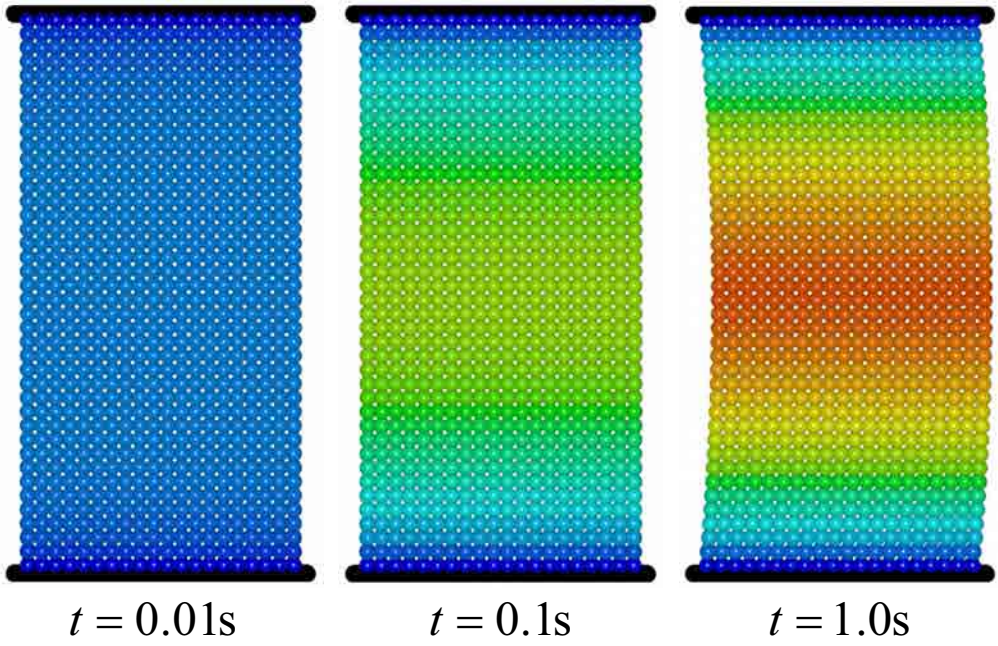


Figure 7.6: Poiseuille flow velocities without refinement

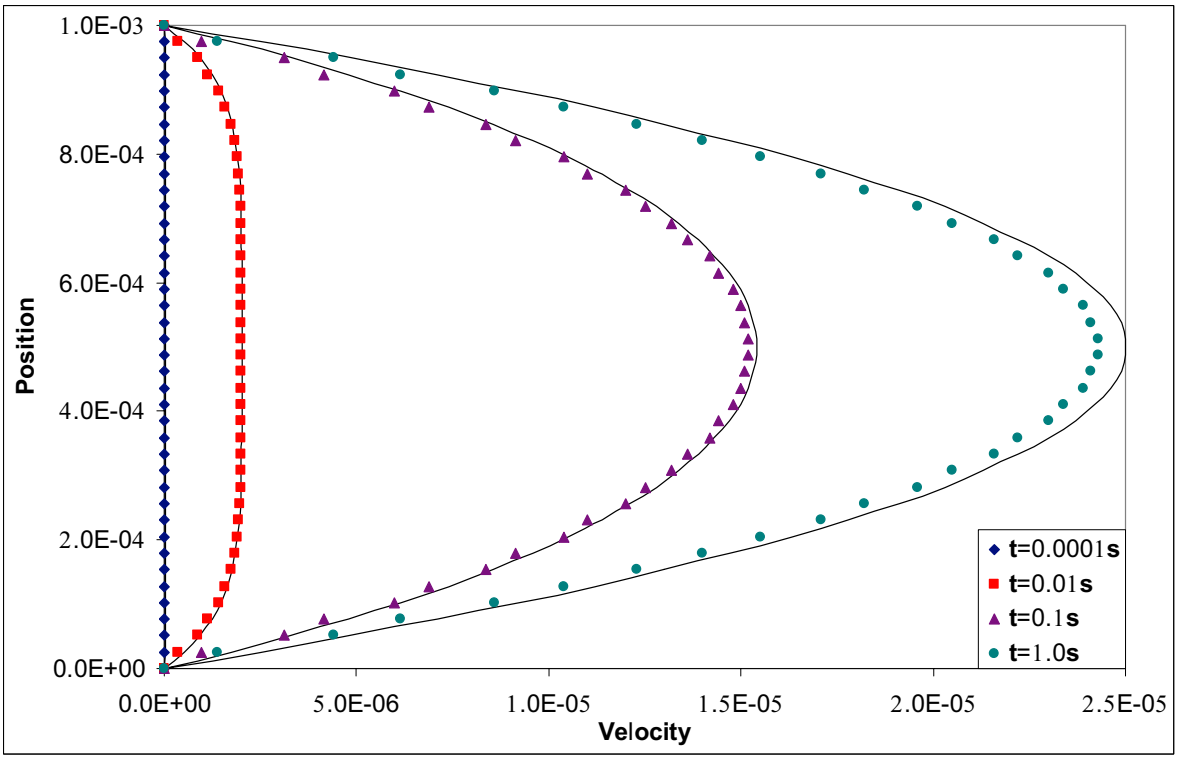


Figure 7.7: Velocity profiles of the Poiseuille flow without refinement

7.2.4 Refined Couette and Poiseuille results

The two dimensional hexagonal refinement pattern is now used to refine the central 16 rows of the Couette and Poiseuille flows as shown in Figure 7.8. This results in the refinement of 320 particles and a simulation consisting of 2720 particles in total.

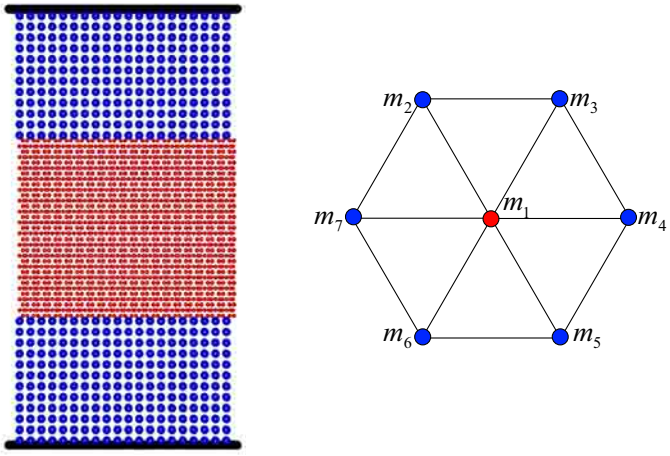


Figure 7.8: Refined region for the Couette and Poiseuille flows

The Couette and Poiseuille examples are relatively simple flows but with the introduction of the refined central region they become challenging simulation since refined particles will remain in the central region of the flow throughout the duration of the simulation. Therefore, any errors introduced at the boundary between the refined and unrefined regions are likely to be able to propagate throughout the whole domain.

Both the refined Couette and Poiseuille flows were found to be unstable without the additional higher order Hessian correction term in the expression for the gradient of the kernel function as derived in Section 3.5.

Three example values for the refinement parameter $(\varepsilon, \alpha) = (0.4, 0.4), (0.4, 0.6)$ and $(0.6, 0.6)$ are used to investigate the performance of the refined SPH simulations. The corresponding density refinement errors and optimal mass distributions for the two dimensional hexagonal refinement pattern are given in Table 7.1.

ε	α	Refinement Error	m_1	$m_2 \rightarrow m_7$
0.4	0.4	0.081334	0.139394	0.143434
0.4	0.6	0.078857	0.281854	0.119691
0.6	0.6	0.004180	0.386914	0.102181

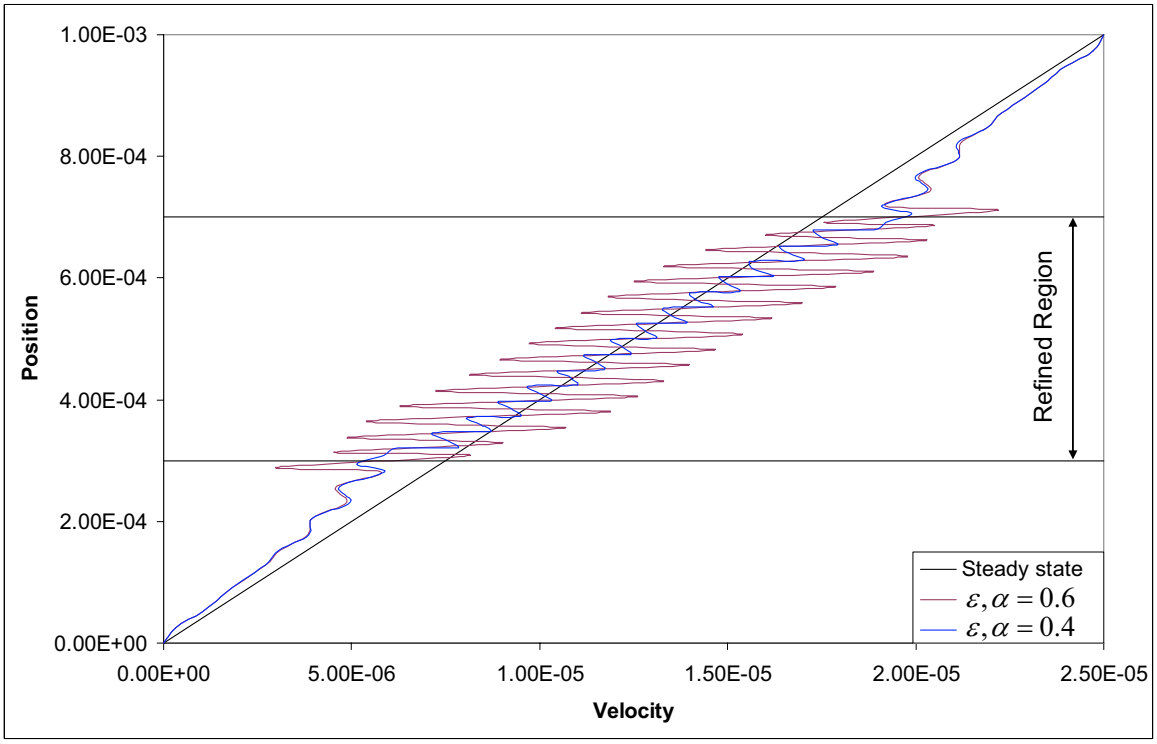
Table 7.1: Refinement errors and optimal mass distributions for 2D hexagonal refinement

To see the affect of the refinement without Hessian stabilization the refined flows were run from their corresponding steady state solutions for a total of 5 timesteps. The resulting perturbations to the velocity profiles are plotted in Figure 7.9 and Figure 7.10. It can be seen that for both the Couette and Poiseuille flows the refinement parameter $(\varepsilon, \alpha) = (0.4, 0.6)$ introduces the smallest perturbations to the steady state solution. This is despite the fact that the refinement errors suggest that $(\varepsilon, \alpha) = (0.6, 0.6)$ should perform the best.

The resulting refined Couette and Poiseuille flows with Hessian stabilization using the refinement parameter $(\varepsilon, \alpha) = (0.4, 0.6)$ are given in Figures 7.11–7.14. Figure 7.11 and Figure 7.13 show the velocity of the refined Couette and Poiseuille flows at several stages of the simulation. Figure 7.12 and Figure 7.14 plot the resulting velocity profiles of the refined Couette and Poiseuille flows.

Figure 7.12 shows the velocity profile of the refined Couette flow at several instances in the simulation. The velocity in both the refined and unrefined regions are in good agreement with the analytic solution. This confirms that with correct choice of refinement parameter and the necessary kernel corrections the refinement procedure does not adversely affect the accuracy of the refined Couette flow simulation.

Figure 7.14 shows the velocity profile of the refined Poiseuille flow at several instances in the simulation. In this case there is a noticeable improvement in the flow when compared to the unrefined simulation given in Figure 7.7. The refined simulation more accurately predicts the maximum steady state velocity and the resulting average error in the steady state velocity profile is reduced from 4% to 1.5%. This shows that the accuracy of SPH simulations can be improved through careful use of particle refinement.



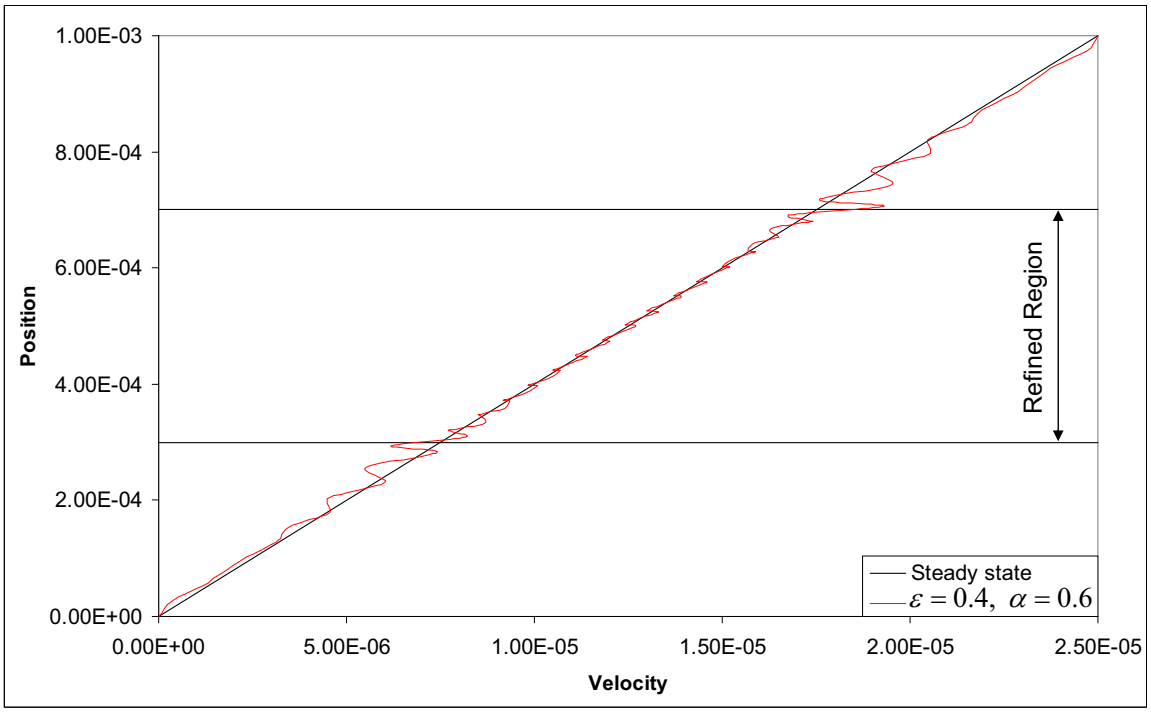
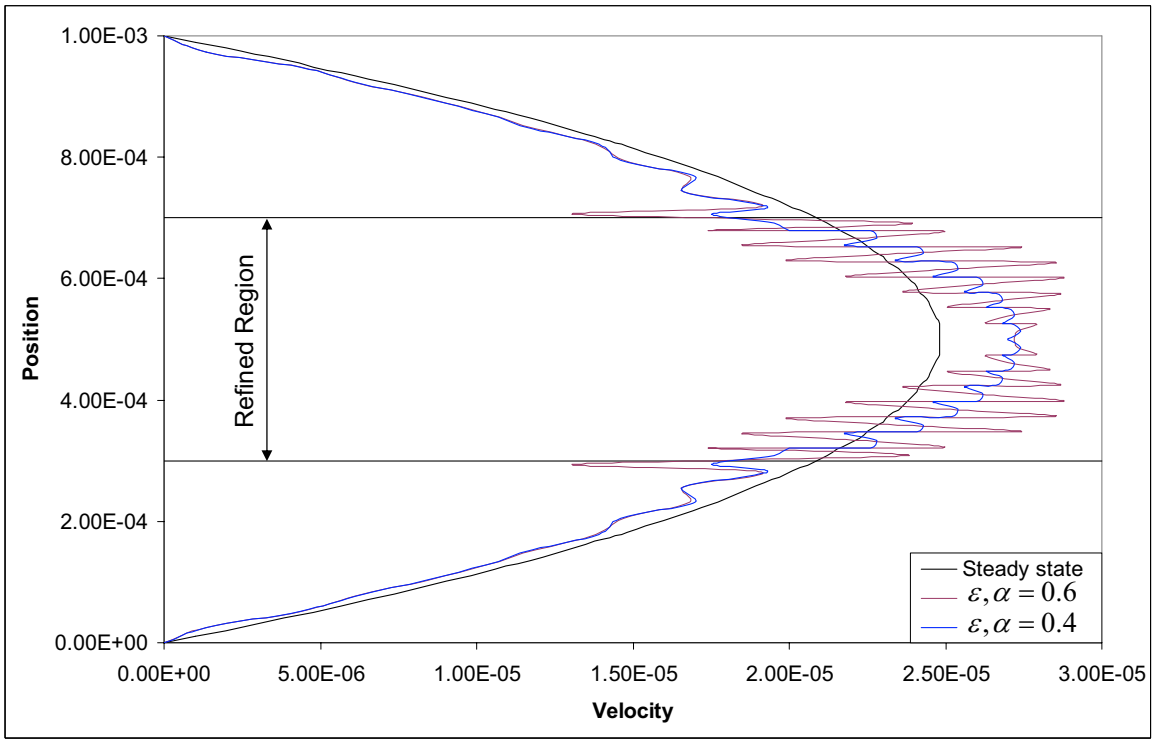


Figure 7.9: Couette velocity profiles after 5 timesteps from the steady state solution without Hessian stabilization



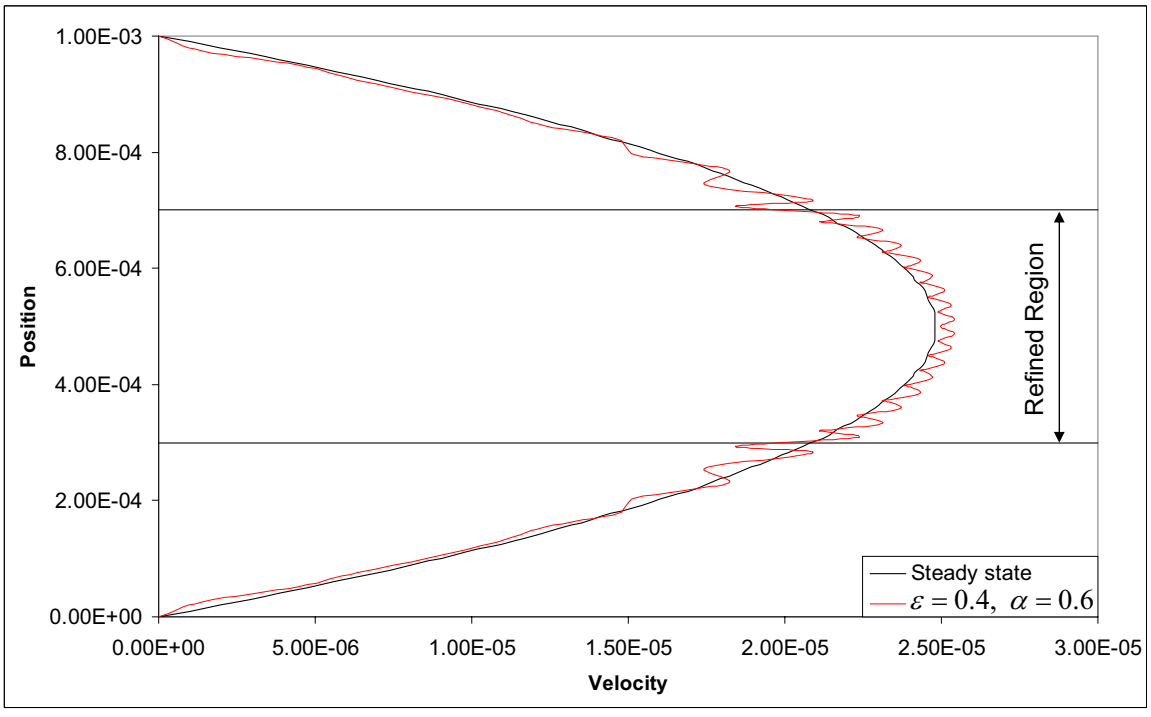


Figure 7.10: Poiseuille velocity profiles after 5 timesteps from the steady state solution without Hessian stabilization

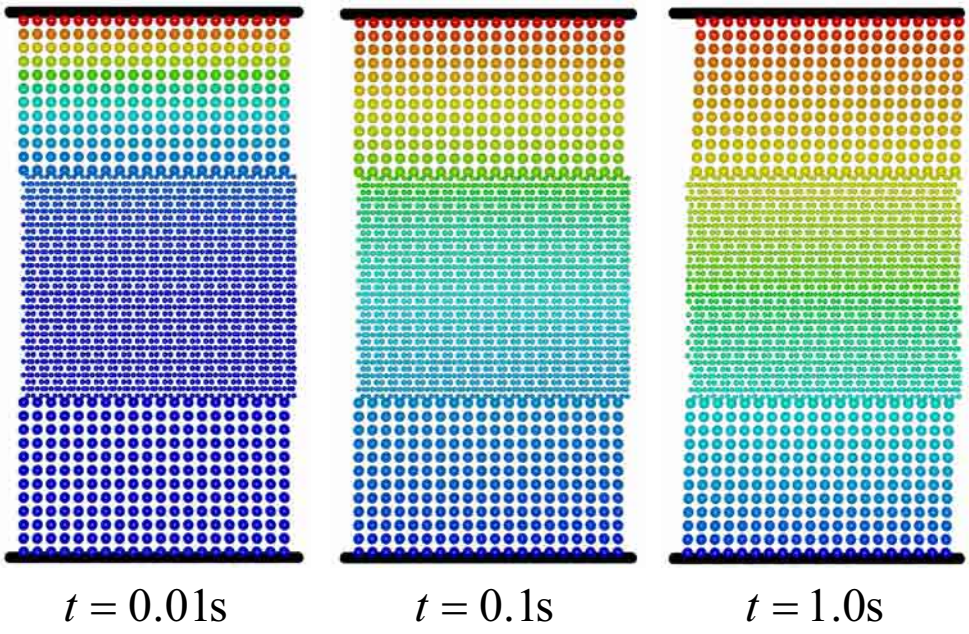


Figure 7.11: Refined Couette flow velocities with Hessian correction

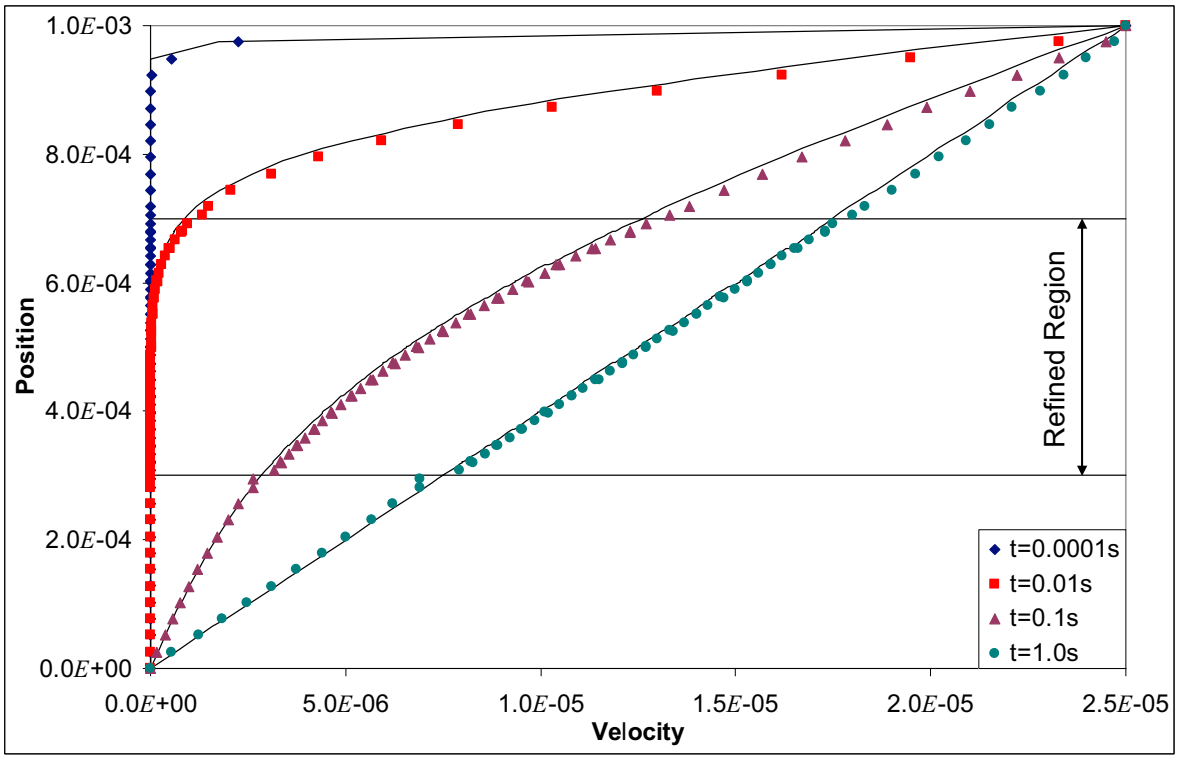


Figure 7.12: Velocity profiles of the refined Couette flow with Hessian correction

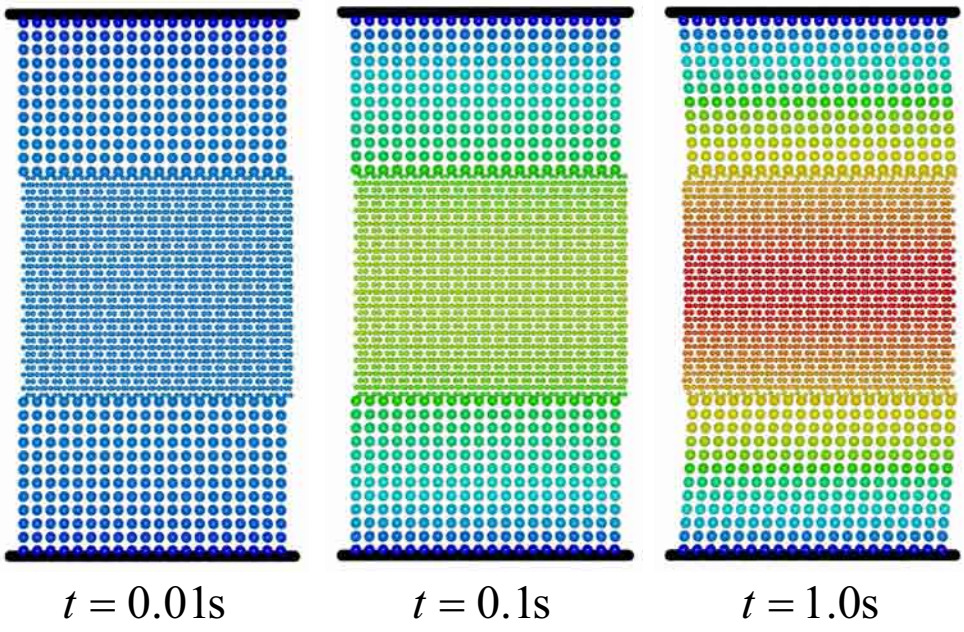


Figure 7.13: Refined Poiseuille flow velocities with Hessian correction

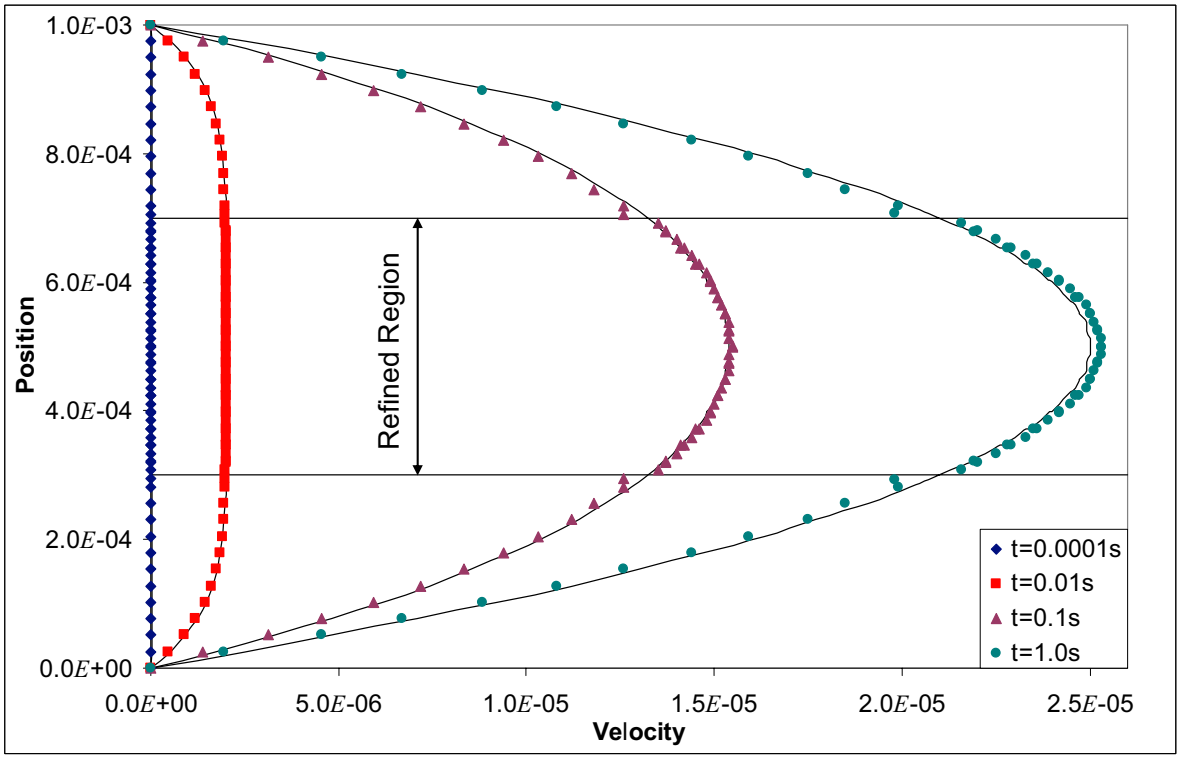


Figure 7.14: Velocity profiles of the refined Poiseuille flow with Hessian correction

7.3 Dynamic refinement simulations

7.3.1 Flow separation through funnel

The tank the fluid flows from is 0.2m wide and the initial height of the fluid is 0.05m. At the start of the simulation this fluid is represented by $61 \times 15 = 915$ unrefined particles as shown in Figure 7.15.

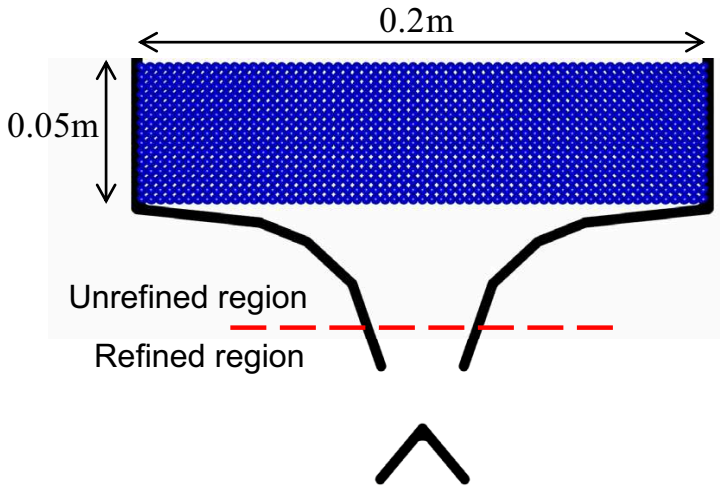


Figure 7.15: Initial particle configuration for the funnel example

The material density is $\rho_0 = 1000\text{kgm}^{-3}$ and viscosity $\mu = 0.5\text{kgm}^{-1}\text{s}^{-1}$. The artificial bulk modulus is taken to be $P_0 = 15000\text{Nm}^{-2}$ and a constant timestep of 0.125×10^{-4}s is used throughout the simulation.

The curved boundary of the funnel is represented by five line segments on each side of the tank with a further two representing the central wedge. The coordinates of which are given in Table 7.2. The boundary contact forces are calculated using the variational formulation given in Section 5.3.

As particles flow through the funnel they are refined into their corresponding daughter particles (when x–coordinate is less than $-4.0 \times^{-2}$ m). The daughter particle velocities are set to be equal to the velocity of the particle they replace and as such the refinement procedure is fully conservative.

The refinement parameter used is $(\varepsilon, \alpha) = (0.4, 0.6)$ which corresponds to the best parameter found for the refined Couette and Poiseuille simulations.

x	y
-0.1017	0.1
-0.1017	-1.67×10^{-3}
-5.79×10^{-2}	$-6.5\times^{-3}$
-4.11×10^{-2}	$-1.33\times^{-2}$
-2.5×10^{-2}	-2.82×10^{-2}
-1.47×10^{-2}	-5.78×10^{-2}

x	y
-1.5×10^{-2}	-9.8×10^{-2}
0.0	-8×10^{-2}
1.5×10^{-2}	-9.8×10^{-2}

Table 7.2: Coordinates of the left boundary and the central wedge (in metres)

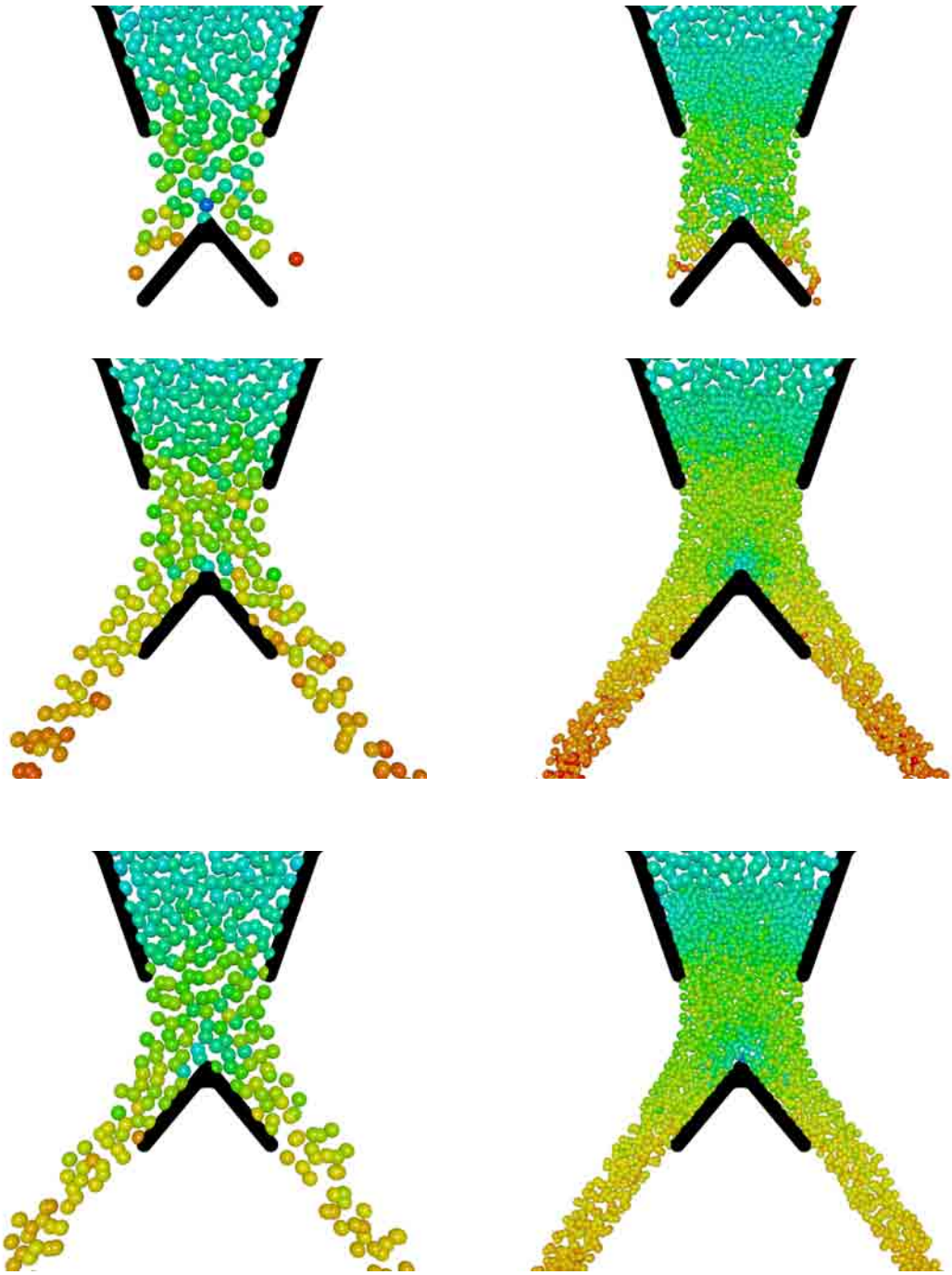
The code was terminated after 0.21 seconds simulation time at which point there was a combined total of 3518 refined and unrefined particles.

Figure 7.16 and Figure 7.17 show the resulting flow both with and without refinement. Without refinement the flow becomes fragmented and breaks up into small groups of particles. When this occurs the particle neighbours decrease and the SPH interpolations become inaccurate. However, with refinement the flow remains largely continuous and the number of neighbouring particles remain approximately uniform.

Figure 7.16 shows the velocities of the particles in the vicinity of the refinement zone clearly showing the transition between unrefined and refined particles. Not only are the velocities continuous over this region but the velocity profile is much more detailed.

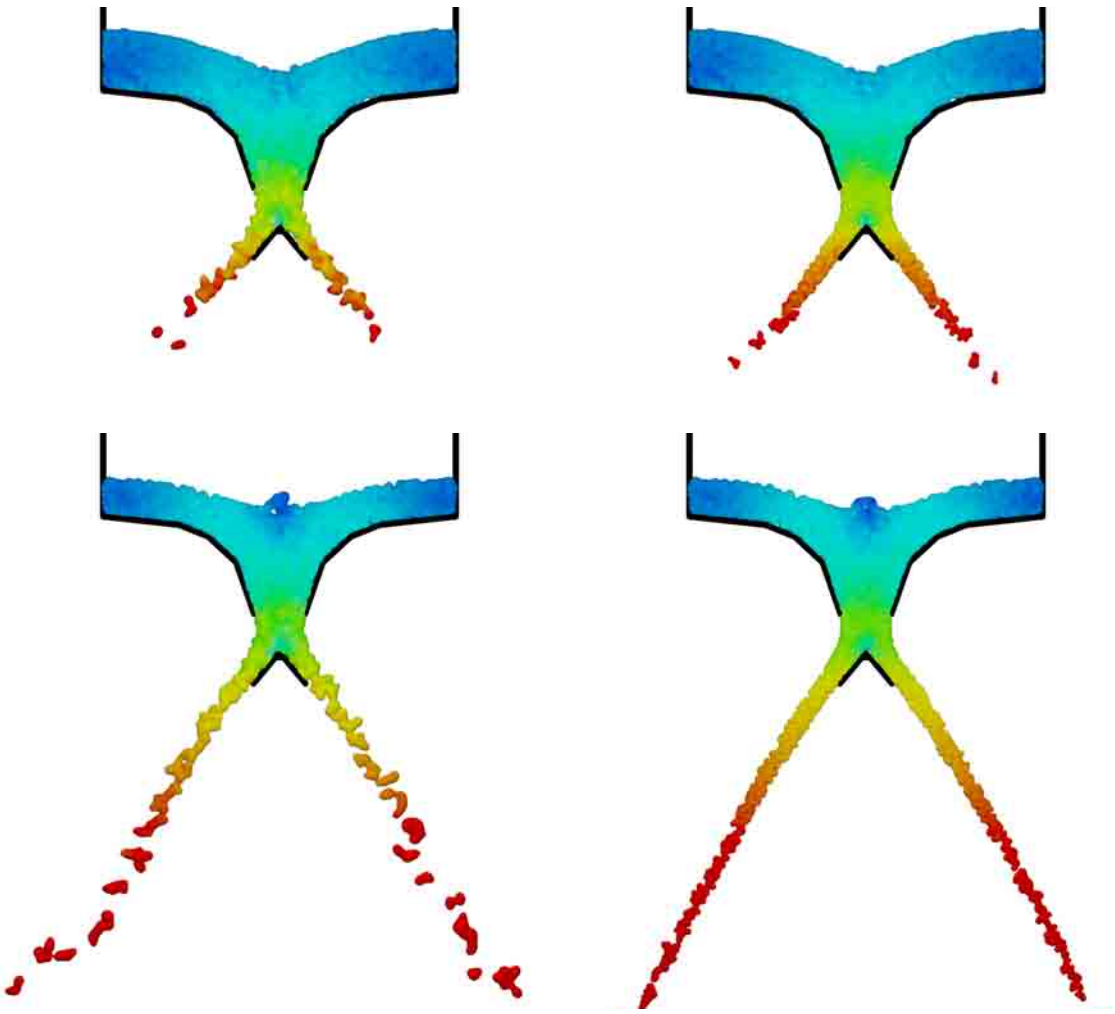
Figure 7.18 plots the particle densities in the vicinity of the central wedge at $t = 0.21$s. Due to the nature of the problem and the artificial incompressibility of the formulation the average particle density variation around the central wedge was between 2%–5%. The densities across the refinement transition and away from the central wedge were largely uniform and within the expected range ($< 1\%$).

This flow separation example has demonstrated that refinement can more accurately capture the physical behaviour of a simulation in addition to improving its accuracy.



Without refinement With refinement

Figure 7.16: Closeup of particle velocities at the funnel refinement zone (at $t = 0.1$s, 0.155s, and 0.21s)



Without refinement With refinement

Figure 7.17: Particle velocities through funnel (at $t = 0.13$s and $t = 0.21$s)

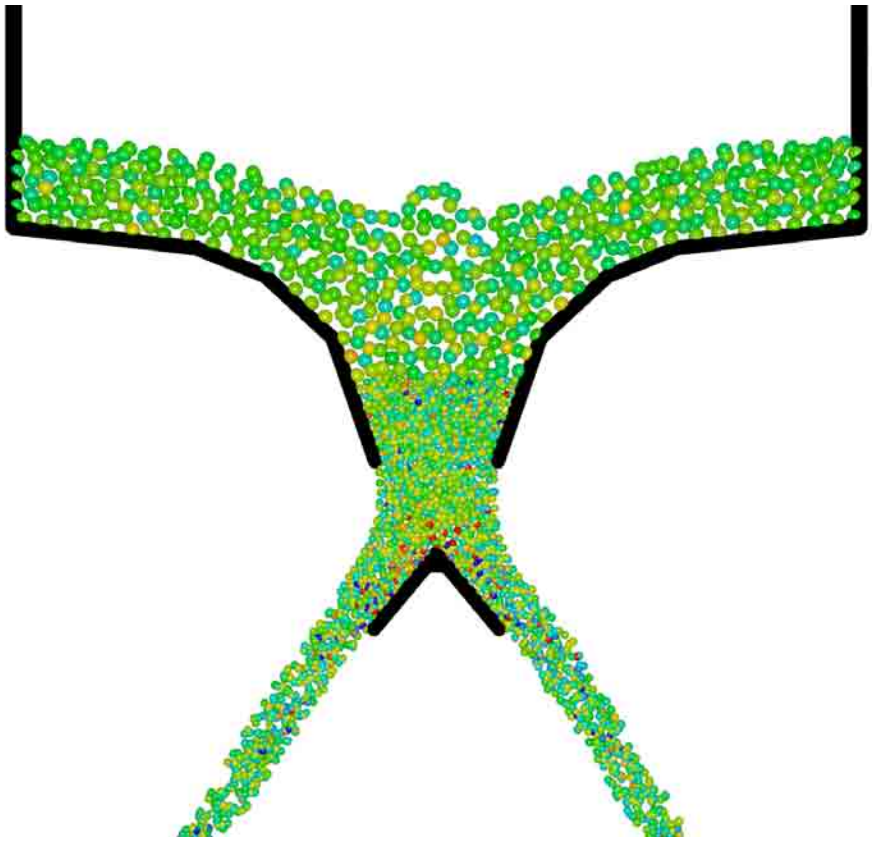


Figure 7.18: Particle densities through funnel at $t = 0.21$s $\left(\rho = 980 - 1020 \text{ kgm}^{-3}\right)$

7.3.2 Emptying tank

In this final example the tank is 0.01m wide and the initial height of the fluid is 0.06m. At the start of the simulation this fluid is represented by $21 \times 120 = 2520$ unrefined particles as shown in Figure 7.19.

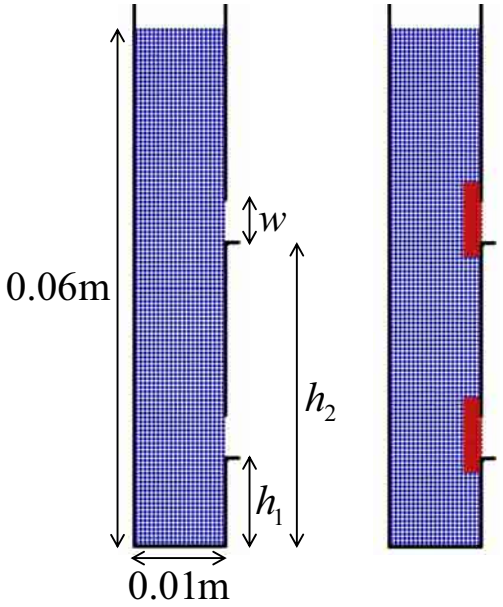


Figure 7.19: Initial particle configuration for emptying tank example

In this simulation two values of viscosity are used, a viscous fluid with $\mu = 0.5\text{kgm}^{-1}\text{s}^{-1}$ and the case of water with corresponding viscosity $\mu = 0.0011\text{kgm}^{-1}\text{s}^{-1}$ with material density $\rho_0 = 1000\text{kgm}^{-3}$. The artificial bulk modulus is taken to be $P_0 = 16741\text{Nm}^{-2}$ and a constant timestep of 10^{-5}s is used throughout these simulations.

As before the boundary is represented by seven line segments and boundary forces are calculated using the variational formulation given in Section 5.3. The outlets are $w = 0.005$m wide and their bases are positioned $h_1 = 0.01$m and $h_2 = 0.035$m from the bottom of the tank.

The refinement zones are positioned over the outlets and particles inside these regions are refined into their corresponding daughter particles using the refinement parameter $(\varepsilon, \alpha) = (0.4, 0.4)$. This results in a total of 3336 particles at the end of the first timestep. The code was terminated after 0.3 seconds simulation time at which point there was a combined total of 13,505 refined and unrefined particles.

Figure 7.20 plots the change in height of the fluid as the tanks empty both with and without particle refinement. The water with viscosity $\mu = 0.0011\text{kgm}^{-1}\text{s}^{-1}$ is found to flow completely from the tank in approximately 0.25s while the viscous fluid with viscosity $\mu = 0.5\text{kgm}^{-1}\text{s}^{-1}$ takes approximately 0.37s to empty completely. It can be seen from the graph that the rate of discharge from the tank has not been affected by the particle refinement.

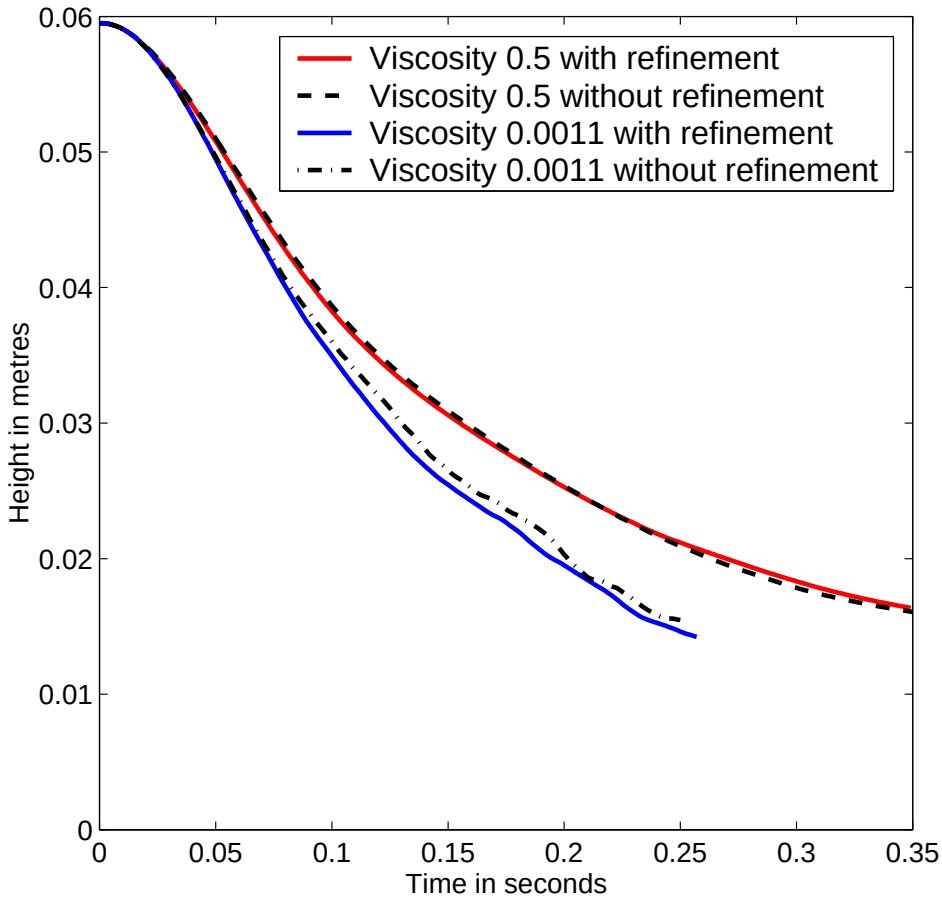


Figure 7.20: Change in height of the fluid in the tank over time

Figure 7.21 shows the emptying tank velocities for the viscous fluid at various instances of the simulation both with and without particle refinement. As the simulation progresses and the tank empties the regions of refined particles are seen to mix with unrefined particles. Across these regions the velocity and density fields are found to be continuous and the variable mass SPH formulation remains accurate and stable.

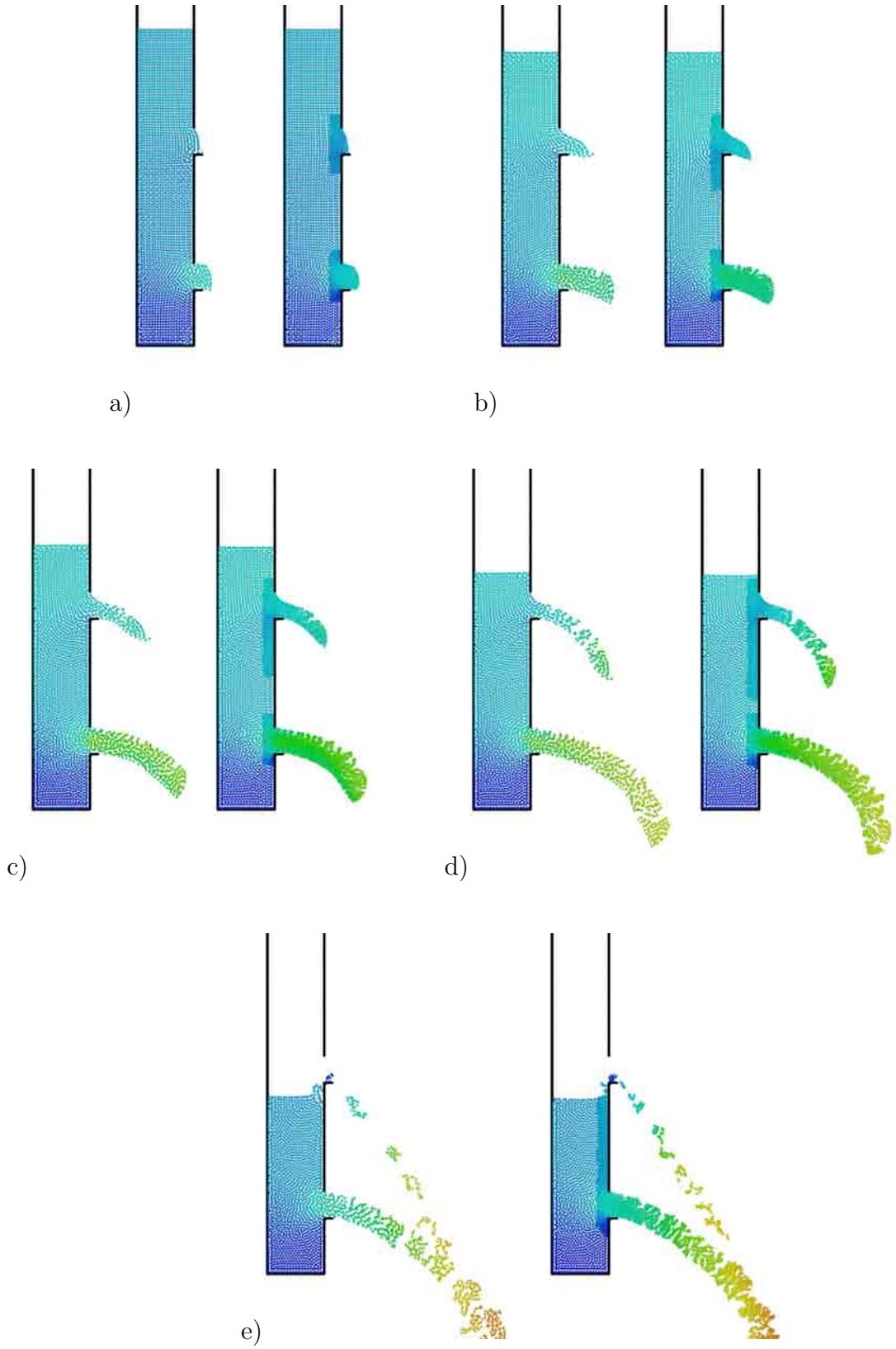


Figure 7.21: Emptying tank velocities for $\mu = 0.5$ with/without refinement

7.4 Concluding remarks

This chapter has presented several variable resolution SPH simulations utilising the dynamic particle refinement strategy developed in Chapter 6.

Static refinement examples

The refined Couette and Poiseuille flows were particularly challenging flows for the variable resolution code to simulate and were used to investigate the affect the refinement parameter (ε, α) has on the accuracy and stability of adaptive SPH simulations.

In these examples the flow is in the horizontal direction only and as such the refined particles remain fixed in their initial hexagonal distributions as the flows evolve. The refined and unrefined regions of the flow are well defined and constant in these simulations. Therefore, particles will remain on the boundary between these regions throughout the duration of the simulations and any errors generated by the refinement procedure at the boundary are more likely to propagate through the domain.

The results from refined Couette and Poiseuille flows demonstrate that the density refinement errors do not not necessarily provide a reliable measure with which to ascertain whether a given refinement parameter (ε, α) will result in accurate or stable variable resolution simulations.

Additional higher order Hessian correction terms were required to stabilize the refined Couette and Poiseuille simulations for any choice of refinement parameter. With this stabilization, the refinement parameter $(\varepsilon, \alpha) = (0.4, 0.6)$ produced the most accurate results. This is despite the fact that the density refinement error suggests that $(\varepsilon, \alpha) = (0.6, 0.6)$ should perform the best.

Nevertheless, the refined Couette flow has shown that with care and the necessary kernel corrections the refinement procedure does not adversely affect the accuracy of the SPH method. The refined Poiseuille flow has shown that refinement can also be used to improve the accuracy of SPH simulations.

Dynamic refinement examples

In the more dynamic simulations of the flow separation through a funnel and the emptying tank the refined particles could quickly mix and redistribute as they moved with the flow and as such the regular lattice formations found in the refined Couette and Poiseuille flows were soon smoothed out. The earlier instabilities did not occur and the additional Hessian corrections were unnecessary.

These examples have demonstrated that dynamic refinement can help to more accurately capture the physical behaviour of a simulation in addition to improving the local accuracy of the method.

The emptying tank example has shown that the variable resolution SPH formulation can easily cope with mixed regions of refined and unrefined particles. In the transitional regions the velocity and density fields were found to be continuous and the variable resolution SPH formulation remains accurate and stable.

Chapter 8

Summary and future developments

The main objective of this research was to provide a rigourous theoretical framework for the analysis of general particle refinement algorithms in SPH.

This objective has been successfully accomplished and refinement has been incorporated into a new general purpose variable resolution SPH code in the form of particle splitting algorithms. The resulting formulation has been applied to several examples of incompressible fluid flows and the accuracy of the refinement process has been verified by comparison to several analytic solutions.

8.1 General remarks

The thesis started with an introduction to the development of meshless methods including a detailed literature review covering the state of the art SPH method, applications to the simulation of incompressible fluid flows and the current implementations of adaptivity and variable resolutions in SPH simulations. The weaknesses of the previous research on refinement in SPH were identified as the lack of any quantitative study into the errors introduced by particle refinement and the need to prove whether it is possible to derive a fully conservative refinement algorithm. This research has addressed both of these fundamental issues.

In Chapter 2 the formulation of the SPH method was presented for the solution of incompressible Newtonian fluid flows. The traditional form for the discrete govern-

ing equations were derived for the continuity, momentum and energy equations in conjunction with the corresponding equation of state. These equations were then combined with suitable time integration schemes and particle search algorithms to obtain a simple and flexible algorithm for the numerical simulation of a wide range of fluid dynamics problems.

Several techniques were then introduced in Chapter 3 which considerably improve the accuracy and stability of the traditional SPH equations. The addition of kernel and gradient corrections enforce constant and linear consistency of the discrete SPH expressions. The SPH method was stabilized further by including an extra kernel Hessian term in the SPH expression for the gradient of a function. This additional higher order term results in the most accurate, linearly consistent and stabilized SPH formulation.

The first step towards the implementation of variable resolution SPH simulations was the development of a set of governing equations which could cope with particles of varying mass and varying smoothing length. Such a formulation was derived from variational principles in Chapter 4. An additional boundary correction term was introduced in this formulation in order to improve the density evaluation in the vicinity of solid boundaries. This lead to an additional internal force which can be identified as a variationally consistent boundary contact force. In the absence of any solid boundaries the resulting expression for the internal forces was found to take the same form as those derived using the traditional approach. When kernel and gradient corrections were adopted it was shown that the resulting variational SPH algorithm conserved linear and angular momentum for general materials described by stress tensors with both isotropic and deviatoric components.

Traditionally, it has been difficult to enforce essential boundary conditions in SPH simulations. In Chapter 5 several different approaches were described which have been developed specifically to model solid or moving boundaries in meshless methods. In particular a new formulation using Lennard-Jones boundary particles was presented. This new approach was motivated by the kernel corrections of Chapter 3 and avoids the instabilities often found when irregular boundary particle distributions are used to model complex geometries.

The remainder of Chapter 5 was concerned with the derivation of a simple and efficient method to exactly calculate the variational boundary contact force (given in Chapter 4) for general boundaries in two dimensions. The resulting contact forces were compared to the approximations that had previously been used in the literature and the accuracy of the new approach was compared to the previous methods via the breaking dam example. The flood defense and water droplet examples then demonstrated the method was capable of modelling complex straight line and curved geometries in two dimensions. In all the simulations in this thesis the boundaries are described by straight or curved line segments only and no boundary particles are required.

In Chapter 6 a general particle refinement strategy based upon particle splitting was developed. Whereby, candidate particles are split into several daughter particles according to a given refinement pattern centred about the original particle position. The separation and smoothing lengths of the daughter particles are controlled by an additional refinement parameter (ε, α) consisting of the separation parameter ε and the smoothing ratio α.

Specific particle refinement criteria were not studied in detail since the resulting refinement algorithm can be applied independently of the refinement criteria used. In all the refinement simulations in this thesis refinement zones are used as the particle refinement criteria. Upon entering these regions particles are replaced with their corresponding set of daughter particles.

The global density refinement error was defined as a measure of the error introduced to the density field caused by the particle refinement procedure. For a given refinement pattern and refinement parameter (ε, α) the refinement error was minimised by solving the model problem to obtain the optimal daughter particle mass distribution. In previous literature the mass of the particle under refinement was usually split equally between the refined particles. However, in general the daughter particle masses obtained from the model problem were not necessarily all equal and the optimal daughter particle mass distributions were often found to be non-uniform.

It was also proved in Chapter 6 that there is a unique fully conservative daughter particle velocity distribution which preserves all the global properties of the system. Namely, when all the daughter particles move with the velocity of the unrefined

particle that they replace then the total kinetic energy and linear and angular momentum will be conserved. This result had previously been missing from the existing literature.

To conclude Chapter 6 the global velocity refinement error was defined as a measure of the error introduced to the velocity field caused by the particle refinement procedure. For the fully conservative velocity configuration this is found to be proportional to the global density refinement error obtained from the solution of the model problem.

Chapter 7 combined all the theory developed in the previous chapters and successfully implemented dynamic particle refinement into the existing SPH framework. The aim was to ascertain whether the global density and velocity refinement errors could be used to identify optimal refinement parameters which result in accurate variable resolution simulations.

The refined Couette and Poiseuille flows showed that this is not necessarily the case and that refinement parameters which generate small refinement errors do not necessarily guarantee stable simulations. However, with care and the addition of extra kernel Hessian corrections the refinement procedure was implemented without sacrificing the accuracy of the method and in the case of the Poiseuille flow the results improved with additional particle refinement.

In the flow separation through a funnel and the emptying tank examples these instabilities did not occur and Hessian corrections were found to be unnecessary. These simulations demonstrated that the variable resolution formulation deals naturally with mixtures of refined and unrefined particles and that the accuracy and physical behaviour of more complex simulations can be improved with the addition of particle refinement.

The contributions made by this thesis are briefly summarised as follows:

- A new Lennard-Jones formulation has been developed to avoid instabilities which occur with irregularly spaced boundary particle.

- A simple and efficient method for exactly calculating the variational boundary contact force for general boundaries in two dimensions has been derived. This approach avoids the need for additional boundary particles and can model

both straight line and curved geometries.

- Dynamic particle refinement has been successfully incorporated into a new general purpose variable resolution SPH code for the simulation of incompressible free surface fluid flows. The accuracy of the formulation has been verified by comparisons with analytic solutions.

- The errors introduced into the density field caused by particle refinement have been studied and controlled. It has been shown that the mass distributions which minimise these errors are not necessarily uniform and that refined particles often have differing masses.

- The unique, fully conservative, daughter particle velocity configuration has been identified for refined particles in SPH simulations.

8.2 Future research

There are several topics covered in this thesis which would benefit from further research. These are briefly summarised as follows:

- It should be possible to evaluate the variational boundary contact force for general boundaries in three dimensions. The equivalent expression for the exact contact force in the vicinity of a single plane has already been derived and implemented successfully for simple examples. However, similar expressions for general boundaries in three dimensions have yet to be derived and approximations are still needed to model corner regions.

- The refinement examples in Chapter 7 should be extended to three dimensions and the relative performance of other refinement patterns should be studied in more detail.

- In the refinement examples in Chapter 7 all the daughter particle smoothing lengths were equal and fixed. The choice of daughter particle smoothing length was found to affect the stability of the refined Couette and Poiseuille flows. The possibility of designing a refinement algorithm which incorporates variable daughter particle smoothing lengths may result in smaller density refinement errors and more stable simulations.

- The implementation of multi-resolution algorithms in the SPH code would allow even greater variation in the particle distribution across the computational domain. With more than one level of refinement even larger problems could be modelled without the need for excessive numbers of particles.

- General particle refinement criteria should be investigated in more detail. In this research particle refinement was based simply on geometric considerations. In particular, a refinement criterion based on an error measure is required in order to locally refine particles in regions identified as having large errors.

- The implementation of particle amalgamation algorithms in the SPH code will be essential to control particle distributions across the computational domain and help reduce the number of particles in regions where higher accuracy is not required. Such algorithms should gather together refined particles and generate coarser distributions of larger amalgamated particles.

- The maximum stable timestep for simulations with particle refinement should be studied in more detail. The timestep will always reduce in simulations with particle refinement because of the reduced daughter particle smoothing lengths. Simulations with particle refinement should always be implemented in such a way that they will be more efficient than the corresponding high resolution simulation.

Appendices

Appendix A

An introduction to fluid dynamics

Consider a fluid contained in a Volume Ω. A point in the fluid $\mathbf{x}$ relative to standard Euclidean coordinates is written as $\mathbf{x} = (x, y, z)$. At t_0 a fluid particle has initial position $\mathbf{x}_0$.

Write $\boldsymbol{\phi}(t, \mathbf{x})$ to denote the trajectory followed by the particle that is at point $\mathbf{x}$ at time $t = 0$ where $\boldsymbol{\phi}$ is assumed to be suitably smooth.

$\boldsymbol{\phi}_t$ is used to denote the map $\mathbf{x} \mapsto \boldsymbol{\phi}(t, \mathbf{x})$ that for fixed time t maps all fluid particles from their position at time $t = 0$ to their position at time t. If V is a region of fluid in Ω with boundary S then $V_t := \boldsymbol{\phi}_t(V)$ is the deformed region at time t with surface $S_t := \boldsymbol{\phi}_t(S)$ moving with the fluid.

Suppose each particle of fluid is initially at point $\mathbf{x}_0$ at time $t = 0$. At time t the position of the particle is given by $\mathbf{x} = \boldsymbol{\phi}(t, \mathbf{x}_0)$ such that $\boldsymbol{\phi}(0, \mathbf{x}_0) = \mathbf{x}_0$
and $\boldsymbol{\phi}(t + \tau, \mathbf{x}_0) = \boldsymbol{\phi}(t, \boldsymbol{\phi}(\tau, \mathbf{x}_0))$ where $\boldsymbol{\phi}$ is a bijective.

The velocity of the flow is given by

$$\mathbf{v}(t, \mathbf{x}) = \frac{d\mathbf{x}}{dt} = \frac{d}{dt}\boldsymbol{\phi}(t, \mathbf{x}_0). \tag{A.1}$$

Let $f(t, \mathbf{x})$ be any scalar field of interest over the volume V_t. The total 'amount' of property f in V_t denoted by F is given by the integral

$$F = \int_{V_t} f(t, \mathbf{x})\, d\mathbf{x}. \tag{A.2}$$

The time rate of change of f over the fluid volume V_t is simply defined as

$$\frac{dF}{dt} = \frac{d}{dt}\int\limits_{V_t} f\left(t,\mathbf{x}\right) d\mathbf{x}. \tag{A.3}$$

The differentiation in equation (A.3) cannot be taken inside the integration since the limit V_t is itself function of time.

The transport theorem

For any $V_t \subset \Omega$, $f\left(t,\mathbf{x}\right)$ a scalar field on Ω with sufficiently smooth flow $\boldsymbol{\phi}\left(t,\mathbf{x}\right)$ then

$$\frac{d}{dt}\int\limits_{V_t} f\left(t,\mathbf{x}\right) d\mathbf{x} = \int\limits_{V_t}\left[\frac{\partial f}{\partial t} + \nabla\cdot\left(f\mathbf{v}\right)\right] d\mathbf{x}. \tag{A.4}$$

Proof

$$\frac{d}{dt}\int\limits_{V_t} f\left(t,\mathbf{x}\right) d\mathbf{x} = \frac{d}{dt}\int\limits_{V_0} f\left(t,\boldsymbol{\phi}\left(t,\mathbf{x}_0\right)\right) J\left(t,\mathbf{x}_0\right) d\mathbf{x}_0 \tag{A.5}$$

where $J\left(t,\mathbf{x}\right)$ is the *Jacobian* of $\boldsymbol{\phi}\left(t,\mathbf{x}\right)$ satisfying

$$J = \left|\frac{\partial\phi^i}{\partial x^j}\right| > 0 \qquad \text{noting that} \qquad \frac{\partial}{\partial t}J\left(t,\mathbf{x}\right) = J\left(t,\mathbf{x}\right)\nabla\cdot\mathbf{v}\left(t,\mathbf{x}\right). \tag{A.6}$$

Since V_0 is independent of time, the differentiation may be taken inside the integral to give

$$\int\limits_{V_0}\left(\frac{\partial}{\partial t}f\left(t,\boldsymbol{\phi}\left(t,\mathbf{x}_0\right)\right)\right) J\left(t,\mathbf{x}_0\right) + f\left(t,\boldsymbol{\phi}\left(t,\mathbf{x}_0\right)\right)\frac{\partial J}{\partial t} d\mathbf{x}_0. \tag{A.7}$$

By writing the integrals back in terms of the deformed volumes the desired result is obtained

$$\int\limits_{V_t}\frac{\partial f}{\partial t} + \left(\nabla f\cdot\mathbf{v} + f\nabla\cdot\mathbf{v}\right) d\mathbf{x} = \int\limits_{V_t}\frac{\partial f}{\partial t} + \nabla\cdot\left(f\mathbf{v}\right) d\mathbf{x}, \tag{A.8}$$

$$\frac{d}{dt}\int\limits_{V_t} f\left(t,\mathbf{x}\right) d\mathbf{x} = \int\limits_{V_t}\left[\frac{\partial f}{\partial t} + \nabla\cdot\left(f\mathbf{v}\right)\right] d\mathbf{x}. \tag{A.9}$$

Material derivative

The time rate of change of a variable f as one follows a given fluid particle is named the *Material Derivative* and is given by

$$\frac{Df}{Dt} := \frac{\partial}{\partial t} f\left(t, \boldsymbol{\phi}\left(t, \mathbf{x}_0\right)\right) = \frac{\partial f}{\partial t} + (\nabla f) \cdot \mathbf{v} \ . \tag{A.10}$$

It is important to note that Df/Dt and $\partial f/\partial t$ are both numerically *and* physically different. $\partial/\partial t$ is the local rate of change of a variable at a fixed point, while D/Dt is the rate of change of a variable of a point of fluid moving through the domain. $(\nabla f) \cdot \mathbf{v}$ is known as the *convective derivative*.

The governing equations of fluid dynamics

Continuity equation

Conservation of mass dictates that the amount of fluid in a control volume V_t at time t is conserved

$$\frac{d}{dt} \int\limits_{V_t} \rho dV = 0. \tag{A.11}$$

Applying the Transport Theorem to the above conservation equation yields

$$\int\limits_{V_t} \left[\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) \right] dV = 0. \tag{A.12}$$

Since this holds for any arbitrary control volume V_t the continuity equation is given as

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \tag{A.13}$$

or written in terms of the material derivative as

$$\frac{D\rho}{Dt} = -\rho \ \nabla \cdot \mathbf{v}. \tag{A.14}$$

For an incompressible fluid this simplifies to

$$\nabla \cdot \mathbf{v} = 0. \tag{A.15}$$

Momentum equation

Newton's Second Law states that the sum of the resultant forces on a body is equal its rate of change of momentum. The momentum in a control volume V_t at time t is given by

$$\iiint_{V_t} \rho \mathbf{v} dV. \tag{A.16}$$

External forces acting on a mass of the fluid may be classified as either *body forces* such as gravity or electromagnetic forces, or *surface forces*, such as pressure or viscous stresses. If $\mathbf{F}$ represents the resultant body force per unit mass while $\mathbf{P}$ represents the resultant surface force per unit area. Then the sum of these forces acting over the fluid is given by

$$\iiint_{V_t} \rho \mathbf{F} dV + \iint_{S_t} \mathbf{P} dS. \tag{A.17}$$

In component form Newton's 2$^{\text{nd}}$ Law applied to equations (A.16) and (A.17) results in the equation

$$\frac{d}{dt} \iiint_{V_t} \rho v^i dV = \iiint_{V_t} \rho F^i dV + \iint_{S_t} P^i dS. \tag{A.18}$$

Writing $\mathbf{P} = \boldsymbol{\sigma}\mathbf{n}$ in terms of the stress tensor $\boldsymbol{\sigma}$ where σ^{ij} is the i^{th} component of the force per unit area across the surface element perpendicular to the j^{th} direction, and $\mathbf{n}$ is the unit normal on surface gives

$$\frac{d}{dt} \iiint_{V_t} \rho v^i dV = \iiint_{V_t} \rho F^i dV + \iint_{S_t} \sigma^{ij} n^j dS. \tag{A.19}$$

Applying the Transport theorem to the left-hand side, and the Divergence Theorem to the surface integral gives

$$\iiint_{V_t} \left[\frac{\partial}{\partial t} \left(\rho v^i \right) + \nabla \cdot \left(\rho v^i \mathbf{v} \right) \right] dV = \iiint_{V_t} \rho F^i dV + \iiint_{V_t} \sigma^{ij,j} dV, \tag{A.20}$$

$$\iiint_{V_t} \left[\frac{\partial}{\partial t} \left(\rho v^i \right) + \nabla \cdot \left(\rho v^i \mathbf{v} \right) - \rho F^i - \sigma^{ij,j} \right] dV = 0. \tag{A.21}$$

Since this holds for any arbitrary control volume V_t the momentum equation can be written in component form as

$$\frac{\partial}{\partial t} \left(\rho v^i \right) + \nabla \cdot \left(\rho v^i \mathbf{v} \right) - \rho F^i - \sigma^{ij,j} = 0. \tag{A.22}$$

Expanding the first two terms of the left-hand side

$$\rho\left(\frac{\partial v^i}{\partial t}+\nabla v^i\cdot\mathbf{v}\right)+v^i\left(\frac{\partial \rho}{\partial t}+\nabla\cdot(\rho\mathbf{v})\right)-\rho F^i-\sigma^{ij,j}=0. \tag{A.23}$$

The terms in the first bracket represent the material derivative of v^i while the term in the second bracket are the continuity equation and so vanish.

$$\frac{Dv^i}{Dt}=\frac{1}{\rho}\sigma^{ij,j}+F^i. \tag{A.24}$$

Written in vector form the momentum equation is therefore given by

$$\frac{D\mathbf{v}}{Dt}=\frac{1}{\rho}\nabla\cdot\boldsymbol{\sigma}+\mathbf{F}. \tag{A.25}$$

Energy equation

The change in kinetic and internal energy of the fluid is equal to the work done by the fluid plus the heat added. Applying this to control volume V_t gives the rate of change of energy of the fluid as

$$\frac{d}{dt}\iiint_{V_t}\rho\left(e+\frac{1}{2}\mathbf{v}\cdot\mathbf{v}\right)dV=W+Q. \tag{A.26}$$

The *total energy* per unit mass is given by

$$E=e+\frac{1}{2}\mathbf{v}\cdot\mathbf{v} \tag{A.27}$$

where e is the *internal energy* per unit mass and $\frac{1}{2}\mathbf{v}\cdot\mathbf{v}$ is the *kinetic energy* per unit mass. W is the rate of work done by the surroundings onto V_t and Q is the rate of heat addition to the fluid.

Let T be the temperature and k be the *thermal conductivity* of the fluid. Using Fourier's Law of Heat Conduction the heat conducted across surface dS in the normal direction $\mathbf{n}$ is at the rate

$$-k\frac{\partial T}{\partial n}. \tag{A.28}$$

Therefore rate at which heat is added to V_t across S by conduction by use of the Divergence Theorem is

$$\iint_S(k\nabla T)\cdot\mathbf{n}dS=\iiint_{V_t}\nabla\cdot(k\nabla T)\,dV. \tag{A.29}$$

Let q be the rate of heat added to the fluid per unit mass giving Q

$$Q = \iiint_{V_t} \rho q + \nabla \cdot (k \nabla T)\, dV. \tag{A.30}$$

The rate of work done by the body force and surface force is given by

$$W = \iiint_{V_t} \rho \mathbf{v} \cdot \mathbf{F} dV + \iint_{S_t} \rho \mathbf{v} \cdot \mathbf{P} dS. \tag{A.31}$$

By the relationship $\mathbf{P} = \boldsymbol{\sigma}\mathbf{n}$ and use of the Divergence Theorem the rate of work done by the fluid is given by

$$W = \iiint_{V_t} \rho \mathbf{v} \cdot \mathbf{F} + \nabla \cdot (\boldsymbol{\sigma}\mathbf{v})\, dV. \tag{A.32}$$

Applying the Transport Theorem to the left-hand side of equation (A.26) and substituting equations (A.30) and (A.32) into the right-hand side gives

$$\iiint_{V_t} \frac{\partial}{\partial t}(\rho E) + \nabla \cdot (\rho E \mathbf{v})\, dV = \iiint_{V_t} \left[\rho q + \rho \mathbf{v} \cdot \mathbf{F} + \nabla \cdot (k \nabla T) + \nabla \cdot (\boldsymbol{\sigma}\mathbf{v})\right] dV.$$

Since this holds for any arbitrary control volume V_t the Energy Equation can be written as

$$\frac{\partial}{\partial t}(\rho E) + \nabla \cdot (\rho E \mathbf{v}) = \rho q + \rho \mathbf{v} \cdot \mathbf{F} + \nabla \cdot (k \nabla T) + \nabla \cdot (\boldsymbol{\sigma}\mathbf{v}). \tag{A.33}$$

Simplify the above equation by ignoring contributions due to heat conduction and heat sources. By using the Continuity equation (A.13) gives

$$\frac{DE}{Dt} = \frac{1}{\rho} \nabla \cdot (\boldsymbol{\sigma}\mathbf{v}) + \mathbf{F} \cdot \mathbf{v}. \tag{A.34}$$

Writing the above equation in terms of *internal energy* and in expanded component form allows further simplification

$$\rho \frac{De}{Dt} + v^i \left(\rho \frac{\partial v^i}{\partial t} + \rho v^k \frac{\partial v^i}{\partial x^k} \right) = \sigma^{ij} \frac{\partial v^i}{\partial x^j} + v^i \left(\frac{\partial \sigma^{ij}}{\partial x^j} + \rho f^i \right). \tag{A.35}$$

The terms in the brackets above are simply the Momentum equation multiplied by $\mathbf{v}$ and hence cancel. Giving a simple form of the Energy equation

$$\frac{De}{Dt} = \frac{1}{\rho}\, \boldsymbol{\sigma} : \nabla \mathbf{v} \qquad \text{where} \qquad \mathbf{A} : \mathbf{B} := A^{ij} B^{ij}. \tag{A.36}$$

Appendix B

Kernel function primitives

Quintic kernel primitives

Primitive equation for gamma in the vicinity of a single boundary

• For $0 < r \leqslant h$:

$$\begin{aligned} F(x,y) &= \frac{a}{48h^5}\left(33x^4yr + 26y^3rx^2 + 8y^5r + 15x^6\log(y+r)\right) \\ &+\frac{b}{h^4}\left(1/5\ y^5 + 2/3\ x^2y^3 + x^4y\right) + \frac{c}{8h^3}\left(5x^2yr + 2y^3r + 3x^4\log(y+r)\right) \\ &+\frac{d}{h^2}\left(x^2y + 1/3\ y^3\right) + ey + \frac{f}{x}h^2\tan^{-1}(y/x) \end{aligned}$$

$$\boxed{\begin{array}{c} a = \frac{15}{7}, \quad b = -\frac{35}{3}, \quad c = 24, \quad d = -20, \quad e = 8 \\ f = -\frac{8}{3}, \quad r = \sqrt{x^2+y^2} \end{array}}$$

• For $h \leqslant r \leqslant 2h$:

$$\begin{aligned} F(x,y) &= \frac{a}{48h^5}\left(33x^4yr + 26y^3rx^2 + 8y^5r + 15x^6\log(y+r)\right) \\ &+\frac{b}{h^4}\left(1/5\ y^5 + 2/3\ x^2y^3 + x^4y\right) + \frac{c}{8h^3}\left(5x^2yr + 2y^3r + 3x^4\log(y+r)\right) \\ &+\frac{d}{h^2}\left(x^2y + 1/3\ y^3\right) + \frac{e}{2h}\left(yr + x^2\log(r+y)\right) + fy + \frac{g}{x}h^2\tan^{-1}(y/x) \end{aligned}$$

$$\boxed{\begin{array}{c} a = -\frac{1}{7}, \quad b = \frac{5}{3}, \quad c = -8, \quad d = 20, \quad e = -\frac{80}{3}, \quad f = 16, \\ g = -\frac{64}{21}, \quad r = \sqrt{x^2+y^2} \end{array}}$$

Primitive equation for the gradient of gamma in the vicinity of a single boundary

• For $0 < r \leqslant h$:

$$\begin{aligned} G(x,y) &= \frac{a}{48h^5}\left(33x^4yr + 26y^3rx^2 + 8y^5r + 15x^6\log(y+r)\right) \\ &+\frac{b}{h^4}\left(1/5\ y^5 + 2/3\ x^2y^3 + x^4y\right) + \frac{c}{8h^3}\left(5x^2yr + 2y^3r + 3x^4\log(y+r)\right) \\ &+\frac{d}{h^2}\left(x^2y + 1/3\ y^3\right) + ey \end{aligned}$$

$$\boxed{\begin{array}{c} a = 15, \quad b = -70, \quad c = 120, \quad d = -80, \quad e = 16 \\ r = \sqrt{x^2+y^2} \end{array}}$$

• For $h \leqslant r \leqslant 2h$:

$$\begin{aligned} G(x,y) &= \frac{a}{48h^5}\left(33x^4yr + 26y^3rx^2 + 8y^5r + 15x^6\log(y+r)\right) \\ &+\frac{b}{h^4}\left(1/5\ y^5 + 2/3\ x^2y^3 + x^4y\right) + \frac{c}{8h^3}\left(5x^2yr + 2y^3r + 3x^4\log(y+r)\right) \\ &+\frac{d}{h^2}\left(x^2y + 1/3\ y^3\right) + \frac{e}{2h}\left(yr + x^2\log(r+y)\right) + fy \end{aligned}$$

$$\boxed{\begin{array}{c} a = -1, \quad b = 10, \quad c = -40, \quad d = 80, \quad e = -80 \\ f = 32, \quad r = \sqrt{x^2+y^2} \end{array}}$$

Appendix C

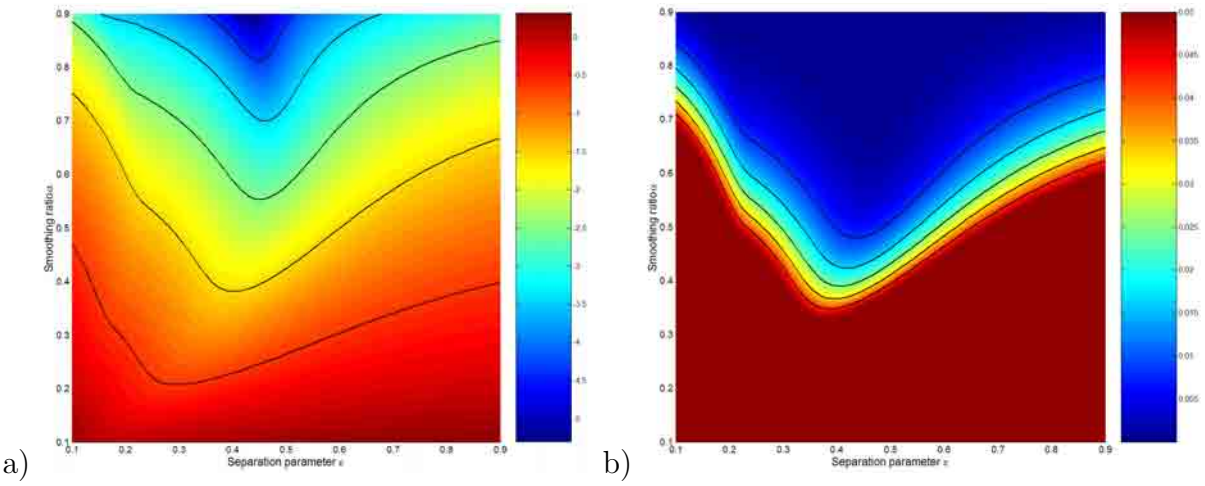
Density refinement results

Appendix C.1 contains the plots of the density refinement results for all the refinement patterns. The refinement errors and corresponding optimal mass distributions are given for $(\varepsilon, \alpha) = (0.4, 0.4)$ and $(0.6, 0.6)$.

In Appendix C.2 the resulting approximations to the original kernel by the refined particles are plotted for the 2D case.

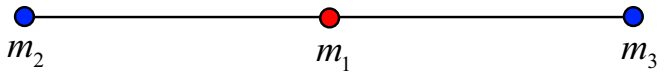
C.1 Density refinement error graphs

1D 3–particle refinement errors



a) b)

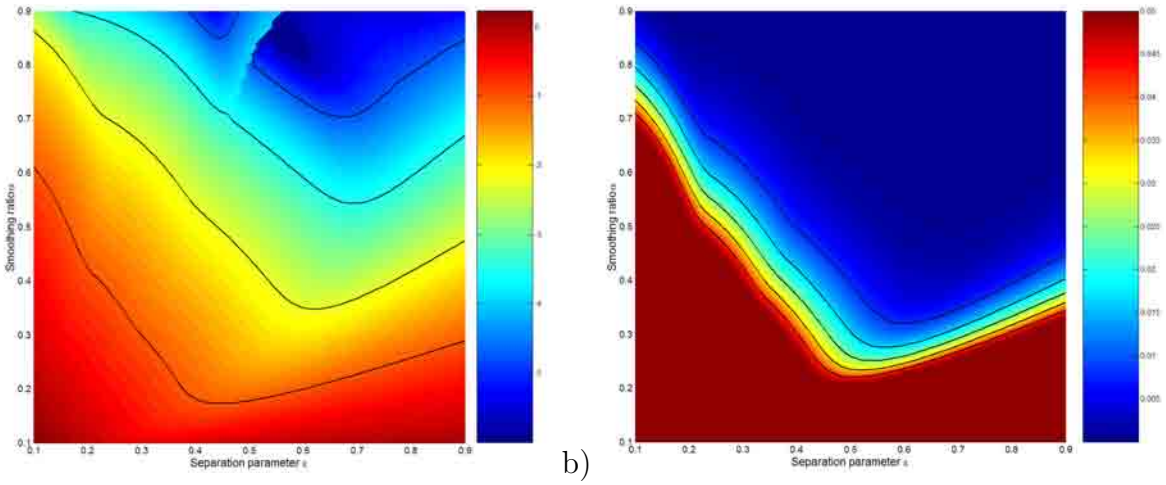
Figure C.1: Graph of density refinement error for 1D–3 particle refinement, a) $\log_{10}$ of error and b) error ≤ 0.05



ε	α	Refinement Error	m_1	m_2, m_3
0.4	0.4	0.0223	0.443	0.2785
0.6	0.6	0.0088	0.6477	0.17615

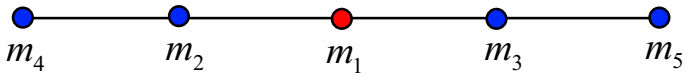
Table C.1: Refinement errors and optimal mass distributions for 1D 3–particle refinement

1D 5–particle refinement errors



a) b)

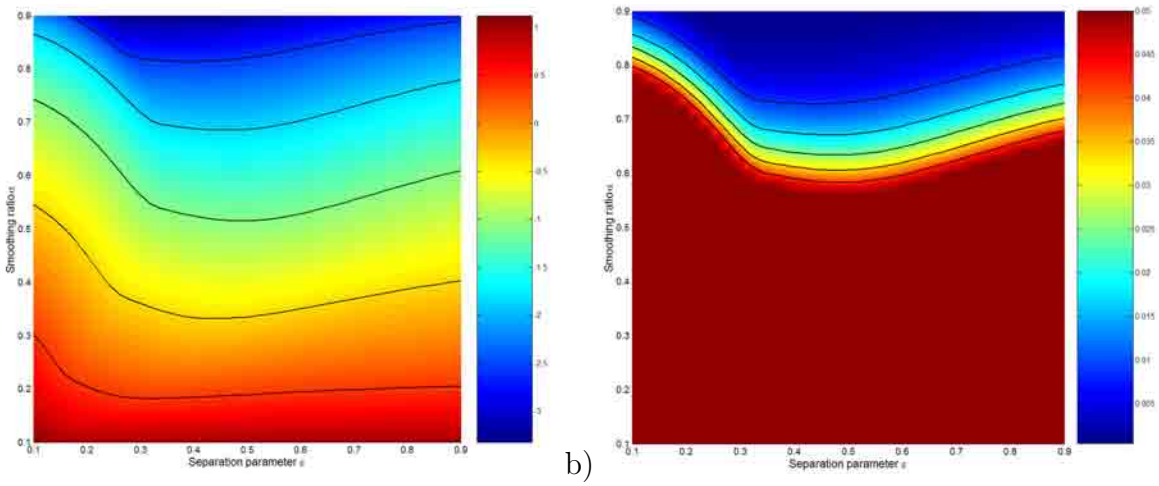
Figure C.2: Graph of density refinement error for 1D–3 particle refinement, a) $\log_{10}$ of error and b) error ≤ 0.05



ε	α	Refinement Error	m_1	m_2, m_3	m_4, m_5
0.4	0.4	0.02077	0.353	0.078	0.244
0.6	0.6	0.000273	0.433	0.187	0.095

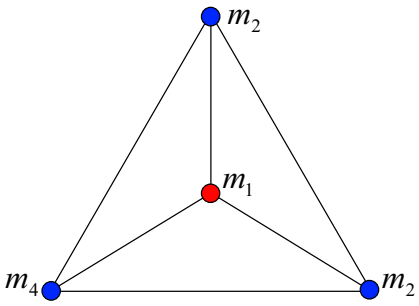
Table C.2: Refinement errors and optimal mass distributions for 1D 5–particle refinement

2D Triangular refinement errors



a) b)

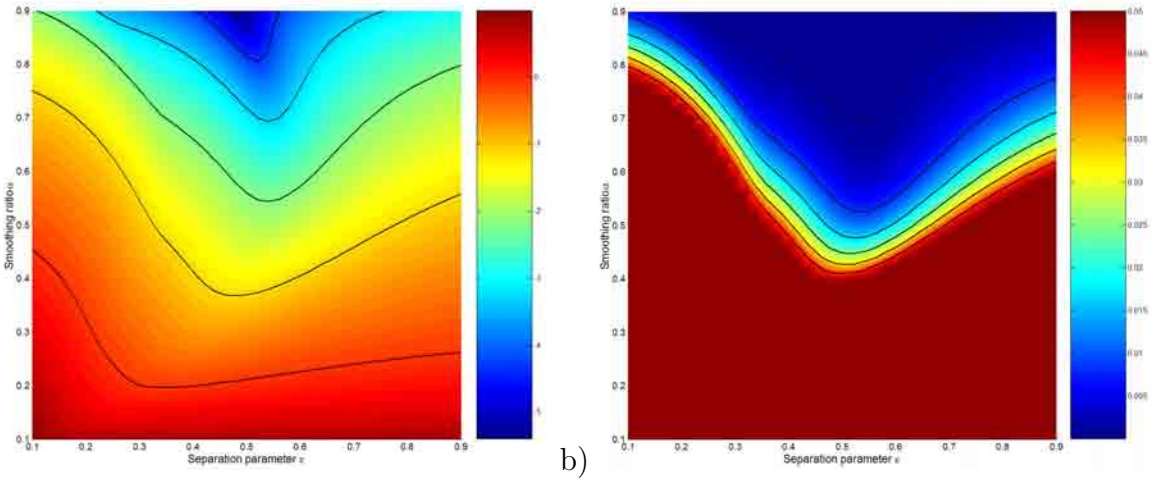
Figure C.3: Graph of density refinement error for 2D triangular refinement, a) $\log_{10}$ of error and b) error ≤ 0.05



ε	α	Refinement Error	m_1	$m_2 \rightarrow m_4$
0.4	0.4	0.242399	0.229399	0.256867
0.6	0.6	0.040509	0.451033	0.182989

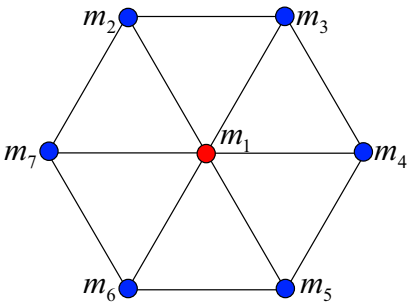
Table C.3: Refinement errors and optimal mass distributions for 2D triangular refinement

2D Hexagonal refinement errors



a) b)

Figure C.4: Graph of density refinement error for 2D hexagonal refinement, a) $\log_{10}$ of error and b) error ≤ 0.05



ε	α	Refinement Error	m_1	$m_2 \rightarrow m_7$
0.4	0.4	0.081334	0.139394	0.143434
0.6	0.6	0.004180	0.386914	0.102181

Table C.4: Refinement errors and optimal mass distributions for 2D hexagonal refinement

3D Hexagonal refinement errors

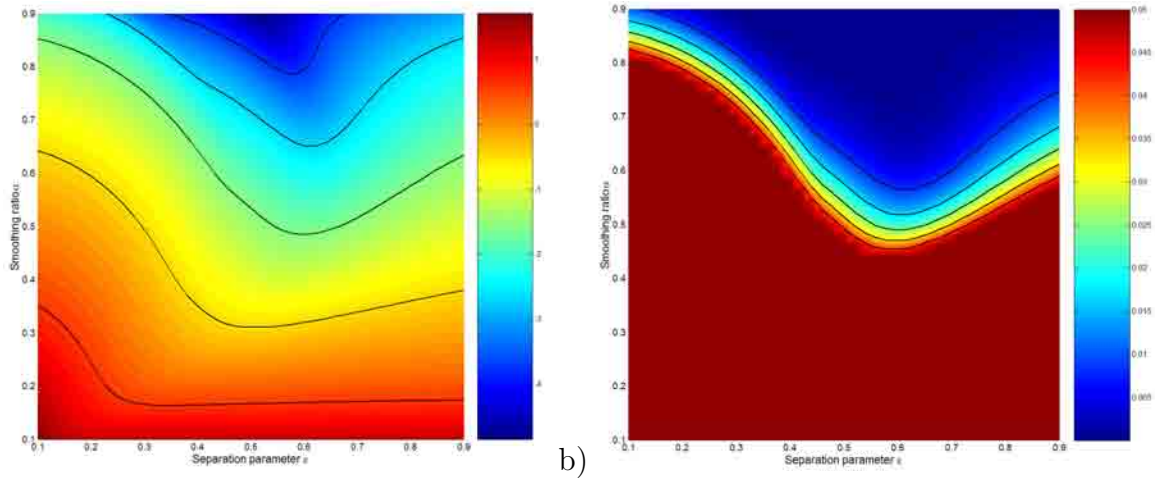


Figure C.5: Graph of density refinement error for 2D hexagonal refinement, a) $\log_{10}$ of error and b) error ≤ 0.05

ε	α	Refinement Error	m_1	$m_2 \rightarrow m_{13}$
0.4	0.4	0.223548	0.0	0.08333
0.6	0.6	0.004180	0.200349	0.066638

Table C.5: Refinement errors and optimal mass distributions for 3D hexagonal refinement (where m_1 is the mass of the particle in the initial unrefined position)

C.2 2D kernel approximations

Unrefined 2D kernel

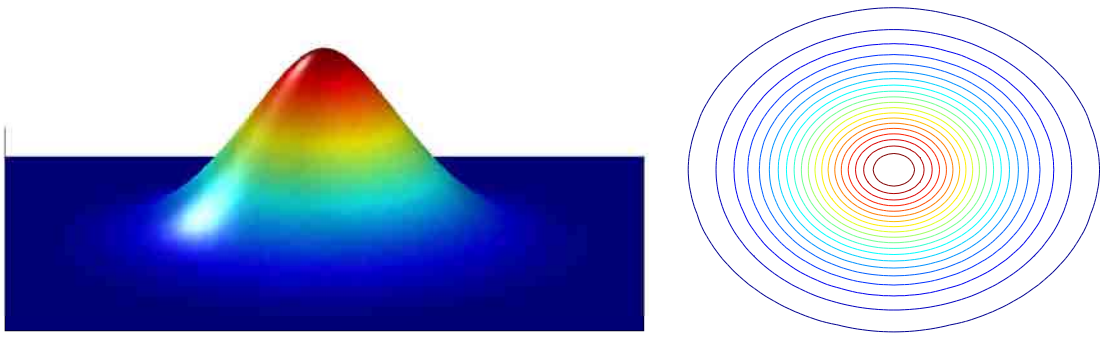


Figure C.6: Unrefined quinic kernel in 2D ($h = 1$)

Refined approximations to the 2D kernel

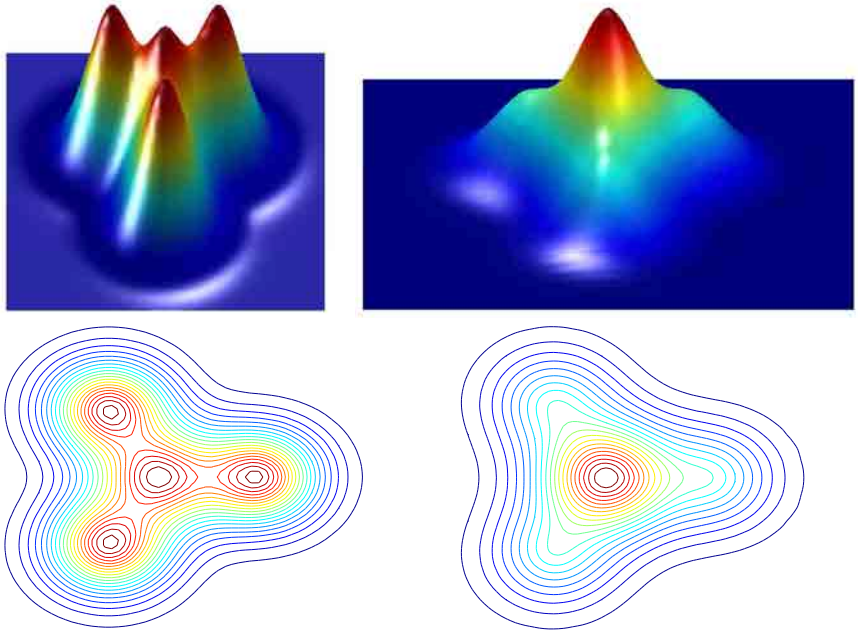


Figure C.7: Refined particles approximation to original kernel for 2D triangular refinement ($(\varepsilon, \alpha) = (0.4, 0.4)$ and $(0.6, 0.6)$ respectively)

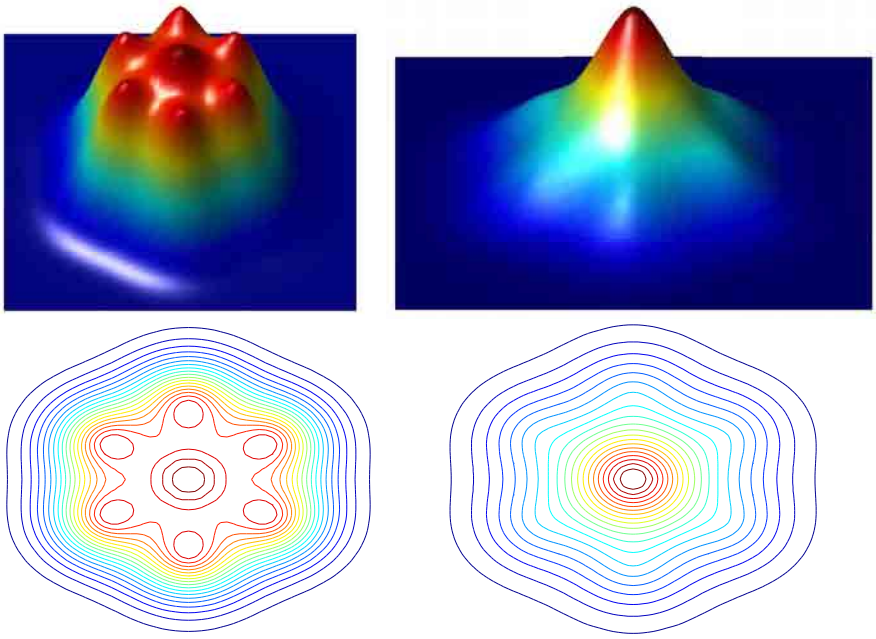


Figure C.8: Refined particles approximation to original kernel for 2D hexagonal refinement ($(\varepsilon, \alpha) = (0.4, 0.4)$ and $(0.6, 0.6)$ respectively)

Appendix D

SPHere2004 Code

D.1 The corrected SPH algorithm

For clarity the SPH algorithm with variable h and all the equations coming from the variational formulation are presented below.

It was shown by Lok [84] that it is essential to apply the variational SPH equations in a consistent manner. For example it should be noted that CSPH corrections cannot be used in updating particle densities when using the direct density method. Particle force equations and density evaluation formulae from different derivations should not be mixed.

• Initialise particle properties : $\mathbf{x}$, $\mathbf{v}$, V, ρ, P, $\cdots$

Begin Timestepping :

- Calculate timestep : Δt
- Search for particle neighbours.
- Calculate kernel correction terms for each particle : α_a, $\boldsymbol{\gamma}_a$, $\mathbf{L}_a$

$$\alpha_a = \sum_b V_b w_b\left(\mathbf{x}_a, h_b\right) \qquad \boldsymbol{\gamma}_a = \frac{1}{\alpha_a}\sum_b V_b \nabla w_b\left(\mathbf{x}_a, h_b\right) \qquad \mathbf{L}_a = \left(\sum_b V_b \nabla \hat{w}_b\left(\mathbf{x}_a, h_b\right) \otimes \mathbf{x}_b\right)^{-1}$$

$$\hat{w}_b\left(\mathbf{x}_a, h_b\right) = \frac{1}{\alpha_a} w_b\left(\mathbf{x}_a, h_b\right) \qquad \nabla \hat{w}_b\left(\mathbf{x}_a, h_b\right) = \frac{1}{\alpha_a}\left(\nabla w_b\left(\mathbf{x}_a, h_b\right) - w_b\left(\mathbf{x}_a, h_b\right)\right)\boldsymbol{\gamma}_a$$

$$\tilde{\nabla} \hat{w}_b\left(\mathbf{x}_a, h_b\right) = \mathbf{L}_a \nabla \hat{w}_b\left(\mathbf{x}_a, h_b\right)$$

- Calculate rate of deformation tensor for each particle : $\mathbf{d}_a = \frac{1}{2}\left(\nabla \mathbf{v}_a + \nabla \mathbf{v}_a^{\mathrm{T}}\right)$

$$2\mathbf{d}_a = \sum_b V_b \left(\mathbf{v}_b \otimes \tilde{\nabla} \hat{w}_b\left(\mathbf{x}_a, h_b\right) + \tilde{\nabla} \hat{w}_b\left(\mathbf{x}_a, h_b\right) \otimes \mathbf{v}_b\right)$$

• Construct deviatoric stress tensor for each particle : $\boldsymbol{\sigma}'_a = 2\mu\left(\mathbf{d}_a - \frac{1}{3}\mathrm{tr}\left(\mathbf{d}_a\right)\right)$

For each particle :

- Update density :

Continuity Method or Direct Density Method

$$\dot{\rho}_a = -\rho_a \sum_b V_b \mathbf{v}_b \cdot \tilde{\nabla}\hat{w}_b\left(\mathbf{x}_a, h_b\right) \qquad \dot{\rho}_a = \sum_b m_b\left(\mathbf{v}_a - \mathbf{v}_b\right)\cdot\nabla w_b\left(\mathbf{x}_a, h_b\right)$$

$$\rho_a^{n+1} = \rho_a^n e^{-\Delta t\sum_b V_b\mathbf{v}_b\cdot\tilde{\nabla}\hat{w}_b(\mathbf{x}_a,h_b)} \qquad \rho_a^{n+1} = \rho_a^n e^{\Delta t\left[\frac{1}{\rho_a}\sum_b m_b(\mathbf{v}_a-\mathbf{v}_b)\cdot\nabla w_b(\mathbf{x}_a,h_b)\right]}$$

- Update volume : $V_a = \frac{m_a}{\rho_a}$
- Update pressure : $P_a = P_0\left(\left(\frac{\rho_a}{\rho_0}\right)^\gamma - 1\right)$
- Calculate particle boundary contact forces : $\mathbf{T}_a^B$
- Calculate particle body forces : $\mathbf{F}_a$
- Calculate particle forces :

Continuity Method

$$\mathbf{T}_a = \sum_b V_a V_b \boldsymbol{\sigma}_b \tilde{\nabla}\hat{w}_a\left(\mathbf{x}_b, h_a\right) \qquad \boldsymbol{\sigma} = -P\mathbf{I} + \boldsymbol{\sigma}'$$

or Direct Density Method

$$\mathbf{T}_a = \sum_b m_a m_b\left(\frac{P_a}{\rho_a^2}\nabla w_b\left(\mathbf{x}_a, h_b\right) - \frac{P_b}{\rho_b^2}\nabla w_a\left(\mathbf{x}_b, h_a\right)\right) + \sum_b V_a V_b \boldsymbol{\sigma}'_b \tilde{\nabla}\hat{w}_a\left(\mathbf{x}_b, h_a\right)$$

- Calculate particle accelerations : $\mathbf{a}_a = \frac{1}{m_a}\left(\mathbf{F}_a - \mathbf{T}_a + \mathbf{T}_a^{\mathrm{B}}\right)$
- Update particle position and velocity :
  - If first timestep : $\mathbf{v}_a^{\frac{1}{2}} = \mathbf{v}_a^0 + \frac{1}{2}\Delta t^1 \mathbf{a}_a^0$
  - Otherwise update via Explicit Leap-Frog :

$$\mathbf{v}_a^{n+\frac{1}{2}} = \mathbf{v}_a^{n-\frac{1}{2}} + \overline{\Delta t}\mathbf{a}_a^n \quad \text{where} \quad \overline{\Delta t} = \frac{1}{2}\left(\Delta t^n + \Delta t^{n+1}\right)$$

$$\mathbf{x}_a^{n+1} = \mathbf{x}_a^n + \Delta t^{n+1}\mathbf{v}_a^{n+\frac{1}{2}}$$

  - Update problem time : $t_{\mathrm{new}} = t_{\mathrm{old}} + \Delta t$
- Output timestep data.
- If $t_{\mathrm{new}} < t_{\mathrm{STOP}}$ continue to the next timestep.

• $t_{\mathrm{new}} > t_{\mathrm{STOP}}$ then end simulation.

D.2 Input file formats

*.in file

File layout:

```
--------------------------------------|
 Title of corresponding problem       |
  &init_cond                          |
        endtime = *******             |
                |                     |
                |                     |
                |                     |
               \ /                    |
          ljmag = *******             |
  &end                                |
--------------------------------------|
```

```
------------|-------------------------------------------------------------
            | MAIN PROGRAM CONTROLS
------------|-------------------------------------------------------------
endtime     | end time of simulation
outputno    | output results after fixed no. of iterations (used if not=0)
outputtime  | time interval for result output (used if outputno=0)
tstepfac    | timestep scaling factor
------------|-------------------------------------------------------------
            | SPH CONTROLS
------------|-------------------------------------------------------------
alpha       | smoothing parameter
sphform     | specifies formulation of sph governing equations:
            |   sphform=1: Continuity Density Form
            |   sphform=2: Direct Density Form
bndtype     | specifies which boundary method is to be used
            |   bndtype=0: No Boundary Method
            |   bndtype=1: Image Particles
            |   bndtype=2: Lennard Jones Potential
            |   bndtype=3: Gamma Function in 2D
            |   bndtype=4: Bounce Back
varh        | variable smoothing length on/off
csph        | specifies sph corrections:
            |   csph=0: No CSPH
            |   csph=1: With CSPH
            |   csph=2: With CSPH and stabilization
eta         | hessian correction constant
xsph        | xsph on/off
engy        | solve energy equation on/off
integscheme | specifies which integration scheme is to be used
            |   integscheme=1: Euler Method
            |   integscheme=2: Euler-Cromer Method
```

```
            |   integscheme=3: Explicit Leap-Frog Scheme
period      | periodic b.cˆs on/off for couette flow
refine      | specifies refinement method
            |   refine=0: No refinement
            |   refine=1: Static refinement
            |   refine=2: Dynamic refinement
artvisc     | specifies artificial viscosity method
            |   artvisc=0: No Artificial Viscosity
            |   artvisc=1: Monaghan formulation
            |   artvisc=2: von Neumann-Richtmyer formulation
kerfunc     | specifies which kernel is to be used:
            |   kerfunc=1: Quintic Kernel(2h)
------------|-------------------------------------------------------
            | OUTPUT CONTROLS
------------|-------------------------------------------------------
log         | write .log file on/off
xydat       | xy.dat output on/off
povout      | povray output on/off
vout        | output to array-viewer on/off
vdata       | specifies variable for output:
            |   vdata=1: velocity
            |   vdata=2: pressure
            |   vdata=3: density
            |   vdata=4: iobj
            |   vdata=5: gmp
            |   vdata=6: icol
outputscreen| output info to screen after fixed no. of interations
------------|-------------------------------------------------------
            | OTHER CONTROLS
------------|-------------------------------------------------------
extcal      | calculates exterior particles on/off
grav        | gravity on/off
cfl         | cfl number
gamma       | exponent in stiff equation of state
elast       | coefficent of elasticity
nadt        | intermittent ADT searching
------------|-------------------------------------------------------
            | LENNARD-JONES CONTROLS
------------|-------------------------------------------------------
ljcutoff    | lj-force cutoff parameter
ljmag       | lj-force magnitude parameter
------------|-------------------------------------------------------
```

***.mat file**

File layout:

```
-------------------------------------------------------------------------------|
 Title of corresponding problem                                                |
        ******  nmat                                                           |
%------------------------------------------------------------------------------ |
%       material properties                                                    |
%                                                                              |
% material index, itype, density                                              |
% OPTIONS: itype =0 for fluid; =1 for elastic solid                            |
%                                                                              |
% ITYPE = 0 : material index, viscosity                                       |
% ITYPE = 1 : material index, iconstit, ieqstate, bulk modulus, shear modulus  |
%                                                                              |
% iconstit determines the constituitive eq; ieqstate determines the eq of state |
% if iconstit = 1 then read a third line containing plasticity data           |
%                 initial yield stress, hardening modulus                      |
%------------------------------------------------------------------------------ |
     1      0   ******                                                        |
     1   ******                                                              |
     2      1   ******                                                        |
     2   ****** ****** ****** ******                                         |
%------------------------------------------------------------------------------ |
-------------------------------------------------------------------------------|
```

- Every material is given a density.
- If the material is a FLUID(itype=0):
  - the material is given a viscosity.
- If the material is a SOLID(itype=1):
  - the material is given the corresponding constitutive eq. and e.o.s
  - the material is given a bulk modulus and a shear modulus

*.bnd file

File layout:

```
-----------------------------------------------------------|
% Title of corresponding problem                           |
 <number of boundary points>                               |
 <x-coordinate> <y-coordinate> <interior angle of point>   |
                      |                                    |
                      |                                    |
                      |                                    |
                     \ /                                   |
 <x-coordinate> <y-coordinate> <interior angle of point>   |
% Additional Lennard-Jones Boundary Particles              |
 <number of additional lj-boundary particles>              |
      1   <x-coordinate> <y-coordinate>                    |
                      |                                    |
                      |                                    |
                      |                                    |
                     \ /                                   |
      n   <x-coordinate> <y-coordinate>                    |
-----------------------------------------------------------|
```

Boundary line segments only implemented for 2D problems.

The boundary file consists of the number of boundary points, followed by a list of their coordinates and the interior angle at the point:

- End points have interior angle = 0.D00.
- File allows for disjoint boundary parts:
  - each part begins and ends with an end point with interior angle = 0.D00.
  - each part begins and ends at distinct points (ie. no closed boundaries.)
  - boundary line segments are described counter-clockwise

Lennard-Jones Boundary Particles

• 2D
Lennard-Jones particles are automatically generated along boundary segments in 2D. Additional Lennard-Jones particles can be included in the boundary file.

• 3D
Lennard-Jones particles can be used in 3D. All particles need to be listed in the boundary file with an additional <z-coordinate> (no particles are automatically generated in 3D).

***.mes file**

Header layout:

```
-------------------------------------------------------------------------|
 Title of corresponding problem                                          |
 <no. dims>                                                              |
 <no. particles >                                                        |
 <start time>                                                            |
 <start timestep>                                                        |
 <total mass>                                                            |
 <total volume>                                                          |
 <particle separation>                                                   |
 <bulk pressure>                                                         |
 <x lower limit> <x upper limit>                                         |
 <y lower limit> <y upper limit>                                         |
 <z lower limit> <z upper limit>                                         |
 <x boundary condition> <y boundary condition> <z boundary condition>  |
-------------------------------------------------------------------------|
```

- The z-coordinate limits and boundary conditions only appear in 3D simulations.
- The upper and lower limits specify the extent of the problem domain.
- The problem domain boundary conditions: 0– none, 1– delete

Main body layout:

If the problem is fully specified then the main body contains one row for each particle with 18 columns of data in 2D and 21 columns of data in 3D.

```
---------------------------------------|
<particle no.>                         |
<x position> <y position> <z position>  |
<x velocity> <y velocity> <z velocity>  |
<volume>                               |
<material id>                          |
<colour id>                            |
<object id>                            |
<movement id>                          |
<x normal> <y normal> <z normal>       |
<initial density> <density>            |
<initial pressure> <pressure>          |
<mass>                                 |
<smoothing length>                     |
---------------------------------------|
```

Note:
Output files are numbered *.dat files and are generated in the same format as the initial *.mes file. To restart simulations only the name of the file needs to be changed.

Bibliography

Bibliography

[1] J.-M. Alimi, A. Serna, C. Pastor, and G. Bernabeu. Smooth particle hydrodynamics: importance of correction terms in adaptive resolution algorithms. *Journal of Computational Physics*, 192:157–174, 2003.

[2] V. Armenio. An improved MAC method (SIMAC) for unsteady high-Reynolds free surface flows. *International Journal For Numerical Methods in Fluids*, 24:185–214, 1997.

[3] V. I. Arnold. *Mathematical Methods of Classical Mechanics.* Springer, 1967.

[4] S. N. Atluri and T. Zhu. New concepts in meshless methods. *International Journal For Numerical Methods in Engineering*, 47:537–556, 2000.

[5] D. S. Balsara. von Neumann stability analysis of smoothed particle hydrodynamics - suggestions for optimal algorithms. *Journal of Computational Physics*, 121:357–372, 1995.

[6] M. Basa, M. Lastiwka, and N. Quinlan. Poiseuille flow with SPH: effects of viscosity model and boundary implementation. ECCOMAS Thematic Conference on Meshless Methods, 2005.

[7] G. K. Batchelor. *An Introduction to Fluid Dynamics.* Cambridge University Press, 1967.

[8] T. Belytschko and Y. Krongauz. Enforcement of essential bounday conditions in meshless approximations using finite elements. *Computer Methods in Applied Mechanics and Engineering*, 131:133–145, 1996.

[9] T. Belytschko, Y. Krongauz, J. Dolbow, and C. Gerlach. On the completeness of meshfree particle methods. *International Journal For Numerical Methods in Engineering*, 43:785–819, 1998.

[10] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. Meshless methods: An overview and recent developments. *Computer Methods in Applied Mechanics and Engineering*, 139:3–47, 1996.

[11] T. Belytschko, Y. Y. Lu, and L. Gu. Element-free Galerkin methods. *International Journal For Numerical Methods in Engineering*, 37:229–256, 1994.

[12] T. Belytschko and S. Xiao. Stability of particle methods with corrected derivatives. *International Journal Computers and Mathematics with Applications*, 43:329–350, 2002.

[13] J. Bonet and S. Kalasegaram. Correction and stabilization of smooth particle hydrodynamics methods with applications in metal forming simulations. *International Journal For Numerical Methods in Engineering*, 47:1189–1214, 2000.

[14] J. Bonet and S. Kulasegaram. Finite increment gradient stabilization of point integrated meshless methods for elliptic equations. *Communications in Numerical Methods in Engineering*, 16:475–483, 2000.

[15] J. Bonet and S. Kulasegaram. Remarks on tension instability of Eulerian and Lagrangian corrected smooth particle hydrodynamics methods. *International Journal For Numerical Methods in Engineering*, 52:1203–1220, 2001.

[16] J. Bonet and S. Kulasegaram. A simplified approach to enhance the performance of smooth particle hydrodynamics methods. *Applied Mathematics and Computation*, 126:133–155, 2002.

[17] J. Bonet, S. Kulasegaram, and T.-S. L. Lok. Corrected smooth particle hydrodynamics methods for fluid and solid mechanics computations. ECCOMAS, 1999.

[18] J. Bonet, S. Kulasegaram, T.-S. L. Lok, and M. Rodreigues-Paz. Corrected smooth particle hydrodynamics - A meshless method for computational mechanics. ECCOMAS, 2000.

[19] J. Bonet and T.-S. L. Lok. Variational and momentum preservation aspects of smooth particle hydrodynamic formulations. *Computer Methods in Applied Mechanics and Engineering*, 180:97–115, 1998.

[20] J. Bonet and J. Peraire. An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems. *International Journal For Numerical Methods in Engineering*, 31:1–17, 1991.

[21] J. Bonet and M. X. Rodriguez-Paz. Hamiltonian formulation of the variable-h SPH equations. *Journal of Computational Physics*, 209:541–558, 2005.

[22] S. Børve, M. Omang, and J. Trulsen. Regularized smoothed particle hydrodynamics: A new approach to simulating magnetohydrodynamic shocks. *Astrophysics Journal*, 561:82–93, 2001.

[23] J. U. Brackmill and H. M. Ruppel. FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational Physics*, 65:314–343, 1986.

[24] J. Campbell, R. Vignjevic, and L. Libersky. A contact algorithm for smooth particle hydrodynamics. *Computer Methods in Applied Mechanics and Engineering*, 184:49–65, 2000.

[25] R. K.-C. Chan and R. L. Street. A computer study of finite-amplitude water waves. *Journal of Computational Physics*, 6:68–94, 1970.

[26] A. K. Chaniotis, D. Poulikakos, and P. Koumoutsakos. Remeshed smoothed particle hydrodynamics for the simulation of viscous and heat conducting flows. *Journal of Computational Physics*, 182:67–90, 2002.

[27] J. K. Chen and J. E. Beraun. A generalized smooth particle hydrodynamics method for nonlinear dynamic problems. *Computer Methods in Applied Mechanics and Engineering*, 190:225–239, 2000.

[28] P. W. Cleary. Modelling confined multi-material heat and mass flows using SPH. *Applied Mathematical Modelling*, 22:981–993, 1998.

[29] P. W. Cleary and J. Ha. Three-dimensional smoothed particle hydrodynamics simulation of high pressure die casting of light metal components. *Journal of Light Metals*, 2:169–183, 2002.

[30] P. W. Cleary, J. Ha, V. Alguine, and T. Nguyen. Flow modelling in casting processes. *Applied Mathematical Modelling*, 26:171–190, 2002.

[31] P. W. Cleary and J. J. Monaghan. Conduction modelling using smoothed particle hydrodynamics. *Journal of Computational Physics*, 148:227–264, 1999.

[32] P. W. Cleary and M. Prakash. Discrete-element modelling and smoothed particle hydrodynamics: potential in the environmental sciences. *Phil. Tran. R. Soc. London*, 362:2003–2030, 2004.

[33] P. W. Cleary, M. Prakash, J. Ha, N. Stokes, and C. Scott. Smooth particle hydrodynamics: Status and future potential. *4^{th} International Conference on CFD in the Oil and Gas, Metallurgical & Process Industries*, 2006.

[34] A. Colagrossi and M. Landrini. Numerical simulation of interfacial flows by smoothed particle hydrodynamics. *Journal of Computational Physics*, 191:448–475, 2003.

[35] R. Courant, K. Friedrichs, and H.Lewy. Über die partiellen differentzengleichungen der mathematischen physik. *Mathematische Annelen*, 100:32–74, 1928.

[36] L. Cueto-Felgueroso, I. Colominas, G. Mosqueira, F. Navarrina, and M. Casteleiro. On the Galerkin formulation of the smoothed particle hydrodynamics method. *International Journal For Numerical Methods in Engineering*, 60:1475–1512, 2004.

[37] S. J. Cummins and M. Rudman. The SPH Projection method. *Journal of Computational Physics*, 152:584–607, 1999.

[38] G. A. Dilts. Moving least-squares particle hydrodynamics - I. Consistency and stability. *International Journal For Numerical Methods in Engineering*, 44:1115–1155, 1999.

[39] G. A. Dilts. Moving least-squares particle hydrodynamics - II. Conservation and boundaries. *International Journal For Numerical Methods in Engineering*, 48:1503–1524, 2000.

[40] C. A. Duarte and J. T. Oden. h-p Clouds - An h-p meshless method. *Numerical Methods for Partial Differential Equations*, 12:673–705, 1996.

[41] C. T. Dyka, P. W. Randles, and R. P. Ingel. Stress points for tension instability in SPH. *International Journal For Numerical Methods in Engineering*, 40:2325–2341, 1997.

[42] M. Ellero, M. Kroger, and S. Hess. Viscoelastic flows studied by smooth particle dynamics. *Journal of Non-Newtonian Fluid Mechanics*, 105:35–51, 2002.

[43] S. Fernández-Méndez. *Mesh-Free Methods and Finite Elements: Friend or Foe?* PhD thesis, Universitat Politecnica de Catalunya, 2001.

[44] S. Fernández-Méndez, J. Bonet, and A. Huerta. Continuous blending of SPH with finite elements. *Computers and Structures*, 83:1448–1458, 2005.

[45] D. A. Fulk. *A Numerical Analysis of Smoothed Particle Hydrodynamics*. PhD thesis, School of Engineering Air University, 1994.

[46] D. A. Fulk and D. W. Quinn. An analysis of 1D smoothed particle hydrodynamics kernels. *Journal of Computational Physics*, 126:165–180, 1996.

[47] M. Griebel and M. A. Schweitzer. *Meshfree Methods for Partial Differential Equations.* Springer, 2002.

[48] R. Gutfraind and S. B. Savage. Smooth particle hydrodynamcs for the simulation of broken ice-flows: Mohr-Coulomb type rheology and frictional boundary conditions. *Journal of Computational Physics*, 134:203–215, 1997.

[49] J. M. Haile. *Molecular Dynamics Simulation: Elementaty Methods.* John Wiley & Sons, 1997.

[50] F. H. Harlow. The particle-in-cell computing method for fluid dynamics. *Methods in Computational Physics*, 3:319–343, 1964.

[51] F. H. Harlow and J. E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids*, 8:2182–2189, 1965.

[52] F. H. Harlow and J. E. Welch. Numerical study of large-amplitude free-surface motions. *Physics of Fluids*, 9:842–851, 1966.

[53] U. Häussler-Combe and C. Korn. An adaptive approach with the element free Galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 162:203–222, 1998.

[54] L. Hernquist and N. Katz. TreeSPH– A unification of SPH with the hierarchical tree method. *The Astrophysical Journal Supplement Series*, 70:419–446, 1989.

[55] D. L. Hicks and L. M. Liebrock. SPH hydrocodes can be stabilized with shape-shifting. *International Journal Computers and Mathematics with Applications*, 38:1–16, 1999.

[56] D. L. Hicks and L. M. Liebrock. Conservative smoothing with B-splines stabilizes SPH material dynamics in both tension and compression. *Applied Mathematics and Computation*, 150:213–234, 2004.

[57] D. L. Hicks, J. W. Swegle, and S. W. Attaway. Conservative smoothing stabilizes discrete-numerical instabilities in SPH material dynamics computations. *Applied Mathematics and Computation*, 85:209–226, 1997.

[58] S. R. Idelsohn, E. Oñate, N. Calvo, and F. D. Pin. The meshless finite element method. *International Journal For Numerical Methods in Engineering*, 58:893–912, 2003.

[59] G. R. Johnson and Stephen R. Beissel. Normalized smoothing functions for SPH impact computations. *International Journal For Numerical Methods in Engineering*, 39:2725–2741, 1996.

[60] I. Kaljević and S. Saigal. An improved element free Galerkin formulation. *International Journal For Numerical Methods in Engineering*, 40:2953–2974, 1997.

[61] S. Kitsionas. *Particle Splitting: A New Method for SPH Star Formation Simulations.* PhD thesis, University of Wales Cardiff, 2000.

[62] S. Kitsionas and A. P. Whitworth. Smoothed particle hydrodynamics with particle splitting, applied to self-gravitating collapse. *Monthly Notices of Royal Astronomical Society*, 330:129–136, 2002.

[63] S. Koshizuka, A. Nobe, and Y. Oka. Numerical analysis of breaking waves using the moving particle semi-implicit method. *International Journal For Numerical Methods in Fluids*, 26:751–769, 1998.

[64] Y. Krongauz and T. Belytschko. Enforcement of essential boundary conditions in meshless approximations using finite elements. *Computer Methods in Applied Mechanics and Engineering*, 131:133–145, 1996.

[65] Y. Krongauz and T. Belytschko. A Pentrov-Galerkin diffuse element method (PG DEM) and its comparison to EFG. *Journal of Computational Mechanics*, 19:327–333, 1997.

[66] S. Kulasegaram. *Particle based methods for metal forming problems.* Master's dissertation, University of Wales Swansea, 1996.

[67] S. Kulasegaram. *Development of Particle Based Meshless Meothods with Applications in Metal Forming Simulations.* PhD thesis, University of Wales Swansea, 1999.

[68] S. Kulasegaram, J. Bonet, R. W. Lewis, and M. Profit. High pressure die casting simulation using a Lagrangian particle method. *Communications in Numerical Methods in Engineering*, 19:679–687, 2003.

[69] S. Kulasegaram, J. Bonet, M. Profit, and W. Lewis. A variational formulation based contact algorithm for SPH applications. *Journal of Computational Mechanics*, 33:316–325, 2004.

[70] M. Lastiwka, N. Quinlan, and M. Basa. Adaptive particle distribution for smoothed particle hydrodynamics. *International Journal For Numerical Methods in Fluids*, 47:1403–1409, 2005.

[71] M. Lee and Y. H. Yoo. Analysis of ceramic/metal armour systems. *International Journal of Impact Engineering*, 25:819–829, 2001.

[72] L. D. Libersky and A. G. Petschek. Cylindrical smoothed particle hydrodynamics. *Journal of Computational Physics*, 109:76–83, 1993.

[73] L. D. Libersky, A. G. Petschek, T. C. Carney, J. R. Hipp, and F. A. Allahdadi. High strain lagrangian hydrodynamics. *Journal of Computational Physics*, 109:67–75, 1993.

[74] T. J. Liszka, C. A. Duarte, and W. W. Tworzydlo. hp-Meshless cloud method. *Computer Methods in Applied Mechanics and Engineering*, 139:263–288, 1996.

[75] G. R. Liu. *Meshfree Methods.* CRC Press, 2002.

[76] G. R. Liu and M. B. Liu. *Smoothed Particle Hydrodynamics: A meshfree particle method.* World Scientific, 2003.

[77] M. B. Liu, S. Jun, S. Li, J. Adee, and T. Belytschko. Reproducing kernel particle methods for structural dynamics. *International Journal For Numerical Methods in Engineering*, 38:1655–1679, 1995.

[78] M. B. Liu, G. R. Liu, K. Y. Lam, and Z. Zong. Smoothed particle hydrodynamics for numerical simulation of underwater explosion. *Journal of Computational Mechanics*, 30:106–118, 2003.

[79] M. B. Liu, G. R. Liu, Z. Zong, and K. Y. Lam. Computer simulation of high explosive explosion using smoothed particle hydrodynamics methodology. *Computers and Fluids*, 32:305–322, 2003.

[80] W. K. Liu, S. Jun, D. T. Sihling, Y. Chen, and W. Hao. Multiresolution reproducing kernel particle method for computational fluid dynamics. *International Journal For Numerical Methods in Fluids*, 24:1391–1415, 1997.

[81] W. K. Liu, S. Jun, and Y. F. Zhang. Reprodcing kernel particle methods. *International Journal For Numerical Methods in Fluids*, 20:1081–1106, 1995.

[82] W. K. Liu, S. Li, and T. Belytschko. Moving least square reproducing kernel methods (I) Methodology and convergence. *Computer Methods in Applied Mechanics and Engineering*, 143:113–154, 1997.

[83] E. Y. M. Lo and S. Shao. Simulation of near-shore solitary wave mechanics by an incompressible SPH method. *Applied Ocean Research*, 24:275–286, 2002.

[84] T.-S. L. Lok. *Analysis of Smoothed Particle Hydrodynamics Method for 2D Free-Surface Flow Applications.* PhD thesis, University of Wales Swansea, 2001.

[85] Y. Y. Lu, T. Belytschko, and M. Tabbara. Element-free Galerkin method for wave propagation and dynamic fracture. *Computer Methods in Applied Mechanics and Engineering*, 126:131–153, 1995.

[86] L. B. Lucy. A numerical approach to the testing of the fission hypothesis. *Astronomical Journal*, 82:1013–1024, 1977.

[87] J. C. Martin and W. J. Moyce. Part IV. An experimental study of the collapse of liquid columns on a rigid horizontal plane. *Phil. Tran. R. Soc. London*, 244:312–324, 1952.

[88] J. J. Monaghan. Extrapolating B-splines for interpolation. *Journal of Computational Physics*, 60:253–280, 1985.

[89] J. J. Monaghan. Introduction to SPH. *Computer Physics Communications*, 48:89–96, 1988.

[90] J. J. Monaghan. On the problem of penetration in particle methods. *Journal of Computational Physics*, 82, 1989.

[91] J. J. Monaghan. Simulating free surface flows with SPH. *Journal of Computational Physics*, 110:399–406, 1992.

[92] J. J. Monaghan. Smoothed particle hydrodynamics. *Annual Review Astronomy and Astrophysics*, 30:543–74, 1992.

[93] J. J. Monaghan. SPH without a tensile instability. *Journal of Computational Physics*, 159:290–311, 2000.

[94] J. J. Monaghan, R. A. Cas, A. M. Kos, and M. Hallworth. Gravity currents descending a ramp in a stratified tank. *Journal of Fluid Mechanics*, 379:39–69, 1999.

[95] J. J. Monaghan and R. A. Gingold. Smoothed particle hydrodynamics: Theory and application to non-spherical stars. *Monthly Notices of Royal Astronomical Society*, 181:375–389, 1977.

[96] J. J. Monaghan and R. A. Gingold. Kernel estimates as a basis for general particle methods in hydrodynamics. *Journal of Computational Physics*, 46:429–453, 1982.

[97] J. J. Monaghan and R. A. Gingold. Shock simulation by the particle method SPH. *Journal of Computational Physics*, 52:374–389, 1983.

[98] J. J. Monaghan, J. P. Gray, and R. P. Swift. SPH elastic dynamics. *Computer Methods in Applied Mechanics and Engineering*, 190:6641–6662, 2001.

[99] J. J. Monaghan and A. Kos. Scott Russell's wave generator. *Physics of Fluids*, 12:622–630, 2000.

[100] J. J. Monaghan and J. P. Morris. A switch to reduce SPH viscosity. *Journal of Computational Physics*, 136:41–50, 1997.

[101] J. P. Morris, P. J. Fox, and Y. Zhu. Modelling low Reynolds number incompressible flows using SPH. *Journal of Computational Physics*, 136:214–226, 1997.

[102] B. B. Moussa. On the convergence of SPH method for scalar conservation laws with boundary conditions. Preprint.

[103] B. B. Moussa and J. P. Vila. Convergence of SPH method for scalar nonlinear conservation laws. *SIAM Jounal of Numerical Analysis*, 37:863–887, 2000.

[104] B. Nayroles, G. Touzot, and P. Villon. Generalizing the finite element method: diffuse approximation and diffuse elements. *Journal of Computational Mechanics*, 10:307–318, 1992.

[105] R. P. Nelson and C. B. Papaloizou. Variable smoothing lengths and energy conservation in smoothed particle hydrodynamics. *Monthly Notices of Royal Astronomical Society*, 270:1–20, 1994.

[106] M. Owen, J. V. Villumsen, P. R. Shapiro, and H. Martel. Adaptive smoothed particle hydrodynamics with application to cosmology: methodology II. *Astrophysical Journal Supplement Series*, 116:155–209, 1998.

[107] A. V. Potapov, M. L. Hunt, and C. H. Campbell. Liquid-solid flows using smoothed particle hydrodynamics and the discrete element method. *Powder Technology*, 116:204–213, 2001.

[108] P. W. Randles and L. D. Libersky. Smoothed particle hydrodynamics: Some recent improvements and applications. *Computer Methods in Applied Mechanics and Engineering*, 139:375–408, 1996.

[109] P. W. Randles and L. D. Libersky. Normalized SPH with stress points. *International Journal For Numerical Methods in Engineering*, 48:1445–1462, 2000.

[110] B. W. Ritchie and P. A. Thomas. Multiphase smoothed particle hydrodynamics. *Monthly Notices of Royal Astronomical Society*, 323:743–756, 2001.

[111] M. X. Rodreiguez-Paz. *Corrected Smooth Particle Hydrodynamics-Techniques for Debris Flow Simulations*. PhD thesis, University of Wales Swansea, 2002.

[112] M. X. Rodriguez-Paz and J. Bonet. A corrected smooth particle hydrodynamics formulation of the shallow water equations. *Computers and Structures*, 83:1396–1410, 2005.

[113] C. Schick. *Adaptivity for particle methods in fluid dynamics*. Diploma thesis, University of Kaiserslautern, 2000.

[114] S. Shao and E. Y. M. Lo. Incompressible SPH method for simulating Newtonian and non-Newtonian flows with a free surface. *Advances in Water Resources*, 26:787–800, 2003.

[115] L. Shaofan and W. K. Liu. Moving least square reproducing kernel methods-Part(II): Fourier analysis. *Computer Methods in Applied Mechanics and Engineering*, 139:159–193, 1996.

[116] P. R. Shapiro, H. Martel, J. V. Villumsen, and M. Owen. Adaptive smoothed particle hydrodynamics with application to cosmology: methodology. *Astrophysical Journal Supplement Series*, 103:269–330, 1996.

[117] L. Sigalotti, J. Klapp, E. Sira, Y. Melean, and A. Hasmy. SPH simulations of time-dependent Poiseuille flow at low Reynolds numbers. *Journal of Computational Physics*, 191:622–638, 2003.

[118] N. Sukumar, B. Moran, and T. Belytschko. The natural element method in solid mechanics. *International Journal For Numerical Methods in Engineering*, 43:839–887, 1998.

[119] D. Sulsky, Z. Chen, and H. L. Schreyer. A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118:179–196, 1994.

[120] D. Sulsky and H. L. Schreyer. Axisymmetric form of the material point method with applications to upsetting and Taylor impact problems. *Computer Methods in Applied Mechanics and Engineering*, 139:409–429, 1996.

[121] D. Sulsky, S.-J. Zhou, and H. L. Schreyer. Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications*, 87:236–252, 1995.

[122] J. W. Swegle, D. L. Hicks, and S. W. Attaway. Smooth particle hydrodynamics stability analysis. *Journal of Computational Physics*, 116:123–134, 1995.

[123] H. Takeda, S. M. Miyama, and M. Sekiya. Numerical simulation of viscous flow by smoothed particle hydrodynamics. *Progress of Theoretical Physics*, 92:939–960, 1994.

[124] J. A. Viecelli. A computing method for incompressible flows bounded by moving walls. *Journal of Computational Physics*, 8:119–143, 1971.

[125] R. Vignjevic. Review of development of the smoothed particle hydrodynamics (SPH) method. Technical report, Cranfield University, 2004.

[126] R. Vignjevic, J. Campbell, and L. Libersky. A treatment of zero-energy modes in the smoothed particle hydrodynamics method. *Computer Methods in Applied Mechanics and Engineering*, 184:67–85, 2000.

[127] J. P. Vila. On particle weighted methods and smooth particle hydrodynamics. *Mathematical Models and Methods in Applied Science*, 9:161–209, 1999.

[128] Y. You, J.-S. Chen, and H. Lu. Filters, reproducing kernel and adaptive meshfree methods. *Journal of Computational Mechanics*, 31:316–326, 2003.