



Terms and Conditions of Use of Digitised Theses from Trinity College Library Dublin

Copyright statement

All material supplied by Trinity College Library is protected by copyright (under the Copyright and Related Rights Act, 2000 as amended) and other relevant Intellectual Property Rights. By accessing and using a Digitised Thesis from Trinity College Library you acknowledge that all Intellectual Property Rights in any Works supplied are the sole and exclusive property of the copyright and/or other IPR holder. Specific copyright holders may not be explicitly identified. Use of materials from other sources within a thesis should not be construed as a claim over them.

A non-exclusive, non-transferable licence is hereby granted to those using or reproducing, in whole or in part, the material for valid purposes, providing the copyright owners are acknowledged using the normal conventions. Where specific permission to use material is required, this is identified and such permission must be sought from the copyright holder or agency cited.

Liability statement

By using a Digitised Thesis, I accept that Trinity College Dublin bears no legal responsibility for the accuracy, legality or comprehensiveness of materials contained within the thesis, and that Trinity College Dublin accepts no liability for indirect, consequential, or incidental, damages or losses arising from use of the thesis for whatever reason. Information located in a thesis may be subject to specific use constraints, details of which may not be explicitly described. It is the responsibility of potential and actual users to be aware of such constraints and to abide by them. By making use of material from a digitised thesis, you accept these copyright and disclaimer provisions. Where it is brought to the attention of Trinity College Library that there may be a breach of copyright or other restraint, it is the policy to withdraw or take down access to a thesis while the issue is being resolved.

Access Agreement

By using a Digitised Thesis from Trinity College Library you are bound by the following Terms & Conditions. Please read them carefully.

I have read and I understand the following statement: All material supplied via a Digitised Thesis from Trinity College Library is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of a thesis is not permitted, except that material may be duplicated by you for your research use or for educational purposes in electronic or print form providing the copyright owners are acknowledged using the normal conventions. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone. This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

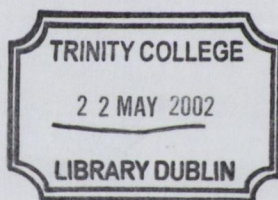
A new technique for vortex methods in computational fluid dynamics

Conor Sweeney

Department of Mechanical and Manufacturing Engineering,
Trinity College,
Dublin 2,
Ireland.

November, 2001

A thesis submitted to the University of Dublin
for the degree of Ph.D.

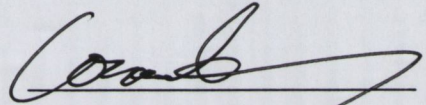


Thesis
6795

Declaration

I declare that I am the author of this thesis and that all the work described herein is my own, unless otherwise referenced. Furthermore this work has not been submitted in whole or part, to any other university or college for any degree or qualification.

I give permission for the Library to lend or copy this thesis upon request.

A handwritten signature in black ink, appearing to read 'Conor Sweeney', written over a horizontal line.

Conor Sweeney,

November 2001

Summary

The random vortex method is a Lagrangian technique which involves solving the stream function-vorticity formulation of the Navier-Stokes equations. Discrete vortex particles, which are created at the surface to satisfy zero slip, are convected in the inviscid flow, while diffusion is modelled as a random walk.

The interaction of the discrete vortices can be computed more efficiently by use of a cloud-in-cell technique. Here the vorticity of the particles is spread onto a fixed Eulerian mesh, providing a source term for the Poisson equation which can be easily solved to provide the stream function (and hence velocity) field on the mesh. The cloud-in-cell technique has traditionally been used with regular polar or Cartesian grids in two dimensions, thus requiring conformal transformation and/or multi-grid methods for application to complex geometries.

A new method, referred to herein as the cloud-in-element method, is presented which allows a single unstructured mesh to be used for multiple bodies and arbitrary geometries in two dimensions. The technique requires a computationally efficient location of point vortices in an unstructured mesh, which has been made possible by use of reference matrices. A finite element method is used to solve the velocity field on the unstructured mesh.

The method is shown to be computationally efficient, with a process time which varies linearly with the number of point vortices used in the simulation. A preliminary study of accuracy and mesh dependency has shown the method to be stable, with increasing accuracy for finer discretisations in space and time.

The algorithm is described, and results for the standard case of impulsively started flow over a circular cylinder are presented for a range of Reynolds numbers from 40 to 9500. Good quantitative agreement for Strouhal number and total drag has been achieved. Qualitative agreement of the instantaneous streamline pattern and flow structure was also observed. Vorticity plots are presented for flow over two cylinders, showing the application of the method to multiple bodies.

Drinking

The thirsty Earth soaks up the Rain,
And drinks, and gapes for drink again.
The Plants soak up the Earth, and are
With constant drinking fresh and fair.
The Sea itself, which one would think
Should have but little need of Drink,
Drinks ten thousand Rivers up,
So filled that they o'erflow the Cup.
The busy Sun (and one would guess
By his drunken fiery face no less)
Drinks up the Sea, and when he has done,
The Moon and Stars drink up the Sun.
They drink and dance by their own light,
They drink and revel all the night.
Nothing in Nature's sober found,
But an eternal health goes round.
Fill up the Bowl, then fill it high,
Fill all the Glasses there, for why
Should every creature drink but I,
Why, Man of Morals, tell me why?

Abraham Cowley, 1656

Acknowledgments

First and foremost, for all of his help, encouragement, guidance, and the odd drink (or two), I'd like to thank my supervisor, Craig Meskell. Thanks are also due to everyone in the Fluids group. The times were great, the atmosphere relaxed, and some lunches seemed to never end.

Some may say that I milked the closing stages of my research for all I could get, and so I would like to thank all of my friends and family for the support, understanding, and free dinners they offered (and I gratefully accepted) during the write-up.

You're all great!

Nomenclature

A	element area
A_i	area of triangular region i
L_i	area coordinate i
N	number of vortices
N_i	shape function i
Re	Reynolds number
St	Strouhal number
U_∞	free-stream flow velocity
a_{cyl}	cylinder radius
p	pressure
Δt	time step
u	local velocity in x -direction
v	local velocity in y -direction
$\vec{v}$	velocity vector
Γ	circulation
γ	circulation per unit length
ψ	stream function
ρ	fluid density
ν	kinematic viscosity
ω	vorticity
$[K]$	global stiffness matrix
$\{f\}$	circulation vector
$\{\psi\}$	stream function vector

Contents

Summary	ii
Nomenclature	v
1 Introduction	1
1.1 Objectives and overview	4
2 Background of the vortex method	6
2.1 Vortex sheet roll-up	7
2.2 Including viscosity	13
2.2.1 Vorticity creation	14
2.2.2 Viscous diffusion	15
2.3 Increasing efficiency	16
2.3.1 Velocity field calculation.	17
2.3.2 Problems near solid boundaries	19
2.4 Multiple bodies and arbitrary geometries	22
2.4.1 Overlapping meshes	22
2.4.2 Conformal transformation	24
2.5 The present work	25
3 The random vortex method	27
3.1 Vorticity	28
3.2 Circulation	29

3.3	The point vortex	31
3.4	Vorticity transport equation	32
3.5	Stream function-vorticity relation	33
3.6	The operator-splitting scheme	34
3.6.1	Vorticity convection	35
3.6.2	The random walk	36
3.7	Flow around a circular cylinder	37
3.7.1	The zero-permeability boundary condition	38
3.7.2	The no-slip boundary condition	38
3.7.3	Point vortices crossing the surface	42
3.8	Summary	45
4	Cloud-in-element	46
4.1	Singularities	46
4.2	The cloud-in-cell method	49
4.3	Computational efficiency	50
4.4	The cloud-in-element method	51
4.4.1	Locating points in the mesh	54
4.4.2	Confirming the home element	57
4.4.3	The neighbours matrix	60
4.4.4	Spreading the circulation	61
4.5	The finite element method	63
4.6	Calculating the velocity field	68
4.7	Summary	69
5	Preparing a simulation	71
5.1	Generating the mesh	71
5.1.1	Meshing near the surface	72
5.1.2	Meshing in the near wake	77
5.2	Defining zones in the mesh	79

5.3	Blockage effects	82
5.4	The outflow boundary	82
5.5	Processing the mesh	83
6	Stepping through the program	85
6.1	Solve for the stream function on the mesh	86
6.2	Calculate the velocity field	87
6.3	Convect the point vortices	87
6.4	Create new vortices at the surface	88
6.5	Diffuse all vortices	89
6.6	Locate the home elements	89
6.7	Spread circulation onto the mesh	90
6.8	Profiling the program	91
6.9	Summary	93
7	Results	94
7.1	Reynolds number = 40	95
7.2	Reynolds number = 550	100
7.3	Reynolds number = 1000	103
7.4	Reynolds number = 3000	105
7.5	Reynolds number = 9500	107
7.6	Strouhal number	113
7.7	Mean drag	113
7.8	Fluctuating velocity	114
7.9	Flow over two cylinders	116
8	Conclusions	118
8.1	Cloud-in-element	118
8.2	Results obtained	119
8.3	Future work	120

List of Figures

2.1	Modelling a vortex sheet.	7
2.2	Vortex sheet roll-up [18].	8
2.3	Random behaviour during vortex sheet roll up [3].	9
2.4	Moore's roll up by combining inner vortices [16].	10
2.5	Christiansen's roll up achieved by applying the cloud-in-cell method.	11
2.6	Chorin's vortex blob.	12
2.7	Chorin & Bernard's smooth roll up using vortex blobs.	13
2.8	Vortex sheet enforcing no-slip condition.	14
2.9	A hierarchy of clusters for a spiral distribution of vortex particles [9].	18
2.10	Point vortices crossing the solid boundary.	20
2.11	Overlapping meshes for vortex-in-cell computation.	24
2.12	Conformal transformations.	25
3.1	Vortex motion in a viscous fluid.	28
3.2	Closed loop C for determining circulation.	29
3.3	Circulation by cells.	30
3.4	Flow induced by a point vortex.	31
3.5	Euler's method.	35
3.6	Second-order Runge-Kutta convection scheme.	36
3.7	Streamlines around a cylinder for starting flow.	39
3.8	Uniform laminar flow past a plate.	39
3.9	Vortex sheet.	41

3.10	Vortex sheet enforcing no-slip condition at surface.	41
3.11	Velocity field around cylinder.	42
3.12	Point vortices crossing the surface.	43
3.13	Delete, Reflect, Absorb.	44
4.1	Velocity around a point vortex.	47
4.2	Core region of a point vortex.	48
4.3	Cutoff functions.	48
4.4	Cloud-in-cell method.	50
4.5	Different meshes.	53
4.6	Location in a regular grid.	54
4.7	A graphical look-up matrix.	55
4.8	Point location.	56
4.9	Incorrect home element.	57
4.10	Closest centroid.	57
4.11	Vector (cross) product.	58
4.12	Testing the home element.	59
4.13	Finding the correct home element.	60
4.14	Area-weighting in a triangle.	61
4.15	Triangle centroid at origin.	66
5.1	A simple complete mesh.	71
5.2	Spreading circulation onto boundary nodes.	73
5.3	Tangential velocity used when creating new point vortices.	74
5.4	Surface node dependency on surface mesh spacing.	75
5.5	Computed drag for $Re = 550$, as Δt is altered.	76
5.6	Computed drag for $Re = 550$, as MR is altered.	76
5.7	Computed drag for $Re = 550$, as Δt is altered.	78
5.8	CPU times for $Re = 550$, with different parameter settings.	78
5.9	The recirculation region.	79

5.10	Situations which (a) do, and (b) do not require the graphical look-up matrix.	81
5.11	Blockage effects.	83
6.1	The algorithm.	85
6.2	Streamlines for starting flow.	86
6.3	Calculated velocity field near the surface.	87
6.4	Convecting point vortices.	88
6.5	Enforcing the no-slip condition.	89
6.6	Diffuse all vortices.	90
6.7	Typical region covered by graphical look-up matrix.	90
6.8	Typical simulation time profile.	92
7.1	Streamlines for $Re = 40$	95
7.2	Drag coefficient, $Re = 40$	96
7.3	$Re = 40$, body circulation / body vorticity.	98
7.4	$Re = 40$: (a) Present work. (b) KL [13].	99
7.5	$Re = 550$, secondary eddy formation.	100
7.6	$Re = 550$, body circulation / body vorticity.	101
7.7	Drag coefficient, $Re = 550$	101
7.8	$Re = 550$, $t = 5.0$, computed streamlines and experimental streak-lines.	102
7.9	$Re = 1000$, body circulation / body vorticity.	103
7.10	Drag coefficient, $Re = 1000$	104
7.11	$Re = 3000$, body circulation / body vorticity.	106
7.12	Drag coefficient, $Re = 3000$	106
7.13	$Re = 9500$, body circulation / body vorticity.	108
7.14	Drag coefficient, $Re = 9500$	109
7.15	$Re = 9500$, $t = 2.0$, computed streamlines and experimental streak-lines.	110

7.16	$Re = 9500, t = 2.0 - 3.5$: (a) Present work. (b) KL [13].	111
7.17	$Re = 9500, t = 4.5 - 6.0$: (a) Present work. (b) KL [13].	112
7.18	Strouhal number: Red stars - present work. Other symbols - exper- imental [29].	114
7.19	Mean drag coefficient: Red stars - present work. Other symbols - experimental [29].	115
7.20	Velocity magnitude, $Re = 100$	115
7.21	Vorticity field for flow over two circular cylinders, $Re = 550$	117

Chapter 1

Introduction

Fluid dynamics, the study of fluids in motion, is important across a broad range of interests. The way weather systems form, how a plane flies through the air, the way our blood flows, and how the ocean forms currents and waves, all involve the motion of fluids and their interactions with their surroundings.

Fluid dynamics has traditionally been researched by empirical and analytical methods. The empirical approach involves running experiments to understand different types of flows. These experiments supply important data and flow visualisations, which allow a deeper understanding of the flow structure, and can yield empirical relations of great practical use. However, most empirical results can not be generalised to new types of flows or geometries. The analytical approach has resulted in mathematical descriptions of fluid characteristics, and rules governing their behaviour. Unfortunately, analytical approaches are often limited by too many simplifications and assumptions to be applicable to flows of practical engineering interest.

The emergence of large scale computing power has meant that physical systems which would have been impracticable to calculate by hand could now be solved. The area of computational fluid dynamics (CFD) has developed, which involves formulating the governing equations of fluid flow in ways which can be solved by computers. The role of computers in fluid dynamics is becoming more important

as advances are being made in simulation techniques, and computer hardware becomes more powerful and more widely available.

The governing equations require more than mere computational power to allow them to be solved for general applications. Direct numerical simulation (DNS) involves solving the governing equations on a fine mesh attempting to capture all of the small scale variations in space and time which may occur in the fluid. The computational effort involved in doing this is so great that this method can currently only be applied to simple cases and low Reynolds numbers. Some simplification of the governing equations is necessary if they are to be solved in a more efficient manner.

Large eddy simulation (LES) involves solving the governing equations on a mesh, as for DNS, except the resolution of the numerical model is not as fine. Some of the fluid behaviour will take place on scales smaller than those which can be captured by the solution on the grid. These sub-grid scales are removed from the main calculation by filtering, and accounted for with a calibrated model of their behaviour.

Most commercial CFD codes implement a further simplification whereby the governing equations are split into a time-averaged component and a random fluctuating component. The resulting equations of the mean behaviour of the flow are called the Reynolds averaged Navier-Stokes equations (RANS), and are solved on a grid. The fluctuating part gives rise to Reynolds stresses, the effects of which are simulated by some mathematical model.

All of these approaches to CFD follow an Eulerian method and discretise the flow *space* onto a grid before applying and solving the governing equations. This means that the flow is solved only for the nodes of the grid, and the solution is very dependent on the meshing techniques used. If the mesh is too coarse, then the small-scale motions of the flow may not be modelled accurately. Alternatively, if the mesh is too fine the computational efficiency decreases.

Vortex methods offer an alternative approach to the solution of the flow field

by using a Lagrangian method. This effectively discretises the *fluid* rather than the flow space. The vorticity field is discretised into a set of representative points and the progress of these points is followed as they interact with the flow and with each other. In flows where boundary-layer separation is an important physical phenomenon, such as flow past bluff-bodies, the vorticity creation process at solid wall boundaries and the subsequent transport of vorticity along the boundary is of primary importance. Modelling the flow by tracking particles of the discretised vorticity field is an efficient approach, because fluid vorticity often lies in concentrated regions of the flow, and thus a vorticity formulation reduces the mathematical description to its essential components.

The use of point vortices in a flow simulation requires that the velocity field due to the point vortices present be calculated at every time step, taking into account the presence of any solid boundaries in the flow. The velocity field calculation can be carried out efficiently by using an Eulerian (grid-based) solution for the velocity field of the point vortices. This mixed (Lagrangian - Eulerian) formulation, while efficient, is limited to use on regular grids which are superimposed on the flow. Using point vortices for simulations of flow over multiple bodies, and arbitrary geometries, becomes a difficult task.

The present work introduces a new technique whereby the velocity field due to all point vortices present may be calculated making use of an unstructured mesh. Location of point vortices in the unstructured mesh is made efficient by use of reference matrices, and the velocity field is solved on the mesh using a finite element method. The unstructured mesh may be used to cover flow fields which include multiple bodies and arbitrary geometries, allowing an efficient solution using a mixed Lagrangian-Eulerian method.

1.1 Objectives and overview

The primary objectives of the present work are:

- To allow the random vortex method to be easily applied to multiple bodies and arbitrary geometries in two dimensions by making use of a single unstructured mesh to calculate the velocity field
- To efficiently locate point vortices within the unstructured mesh.
- To investigate the numerical stability and accuracy of the method as simulation parameters are varied
- To validate the method by comparison of results with other computational and experimental results.

The documentation of how these objectives are achieved has been organised into seven sections.

The background of the vortex method will be reviewed in **Chapter 2**, from its introduction in the study of the roll-up of a vortex sheet, through the inclusion of viscous effects and the velocity field calculation, to the problems posed by multiple bodies and arbitrary geometries.

Chapter 3 will discuss the random vortex method. The governing equations used in this method will be described, and the way they are modelled by the creation, convection and diffusion of vortex particles will be discussed. An explanation of how boundary conditions which arise at a solid surface are enforced in the course of a simulation is given.

The discussion moves on to consider the calculation of the velocity field in **Chapter 4**. This not only concerns maintaining a bounded and physically realistic velocity calculation during the simulation, but also involves methods whereby these calculations may be carried out as efficiently as possible. The central issue of the

present work, which allows such efficient and bounded calculations to be applied to arbitrary geometries and multiple bodies is discussed here.

Chapter 5 discusses the more practical issues of the CFD program. Issues to be considered when generating the unstructured mesh are discussed, along with the effect of time step, mesh spacing, and other simulation parameters.

The computer algorithm is presented in **Chapter 6**, and the main processes at each stage of the algorithm are illustrated. A time profile of the simulation is also presented.

The simulation was run for the standard case of flow over a circular cylinder, and the results are presented in **Chapter 7**. Comparison is made with other computational and experimental results over a range of Reynolds numbers from 40 – 9500. The method allows application to arbitrary geometries and multiple bodies, and an illustrative simulation of the flow over two circular cylinders was executed.

In the concluding chapter, **Chapter 8**, the method introduced in this work, and the results of the present simulations are summarised. Some areas which could be of interest to future research are mentioned.

Chapter 2

Background of the vortex method

The first calculations which made use of tracking point vortices to investigate the behaviour of a vortex sheet were published in 1931 by Rosenhead [18]. The use of point vortices to predict the roll-up of a vortex sheet was the subject of subsequent publications by others, and the different problems encountered, and proposed solutions, are outlined in Section 2.1.

Initially, point vortices were used in calculations of inviscid flows. In 1973, Chorin [5] introduced the random-walk method to allow viscous effects to be included in the flow simulations. Another way of including viscous effects is the particle strength exchange method used in modern vortex blob methods [9]. These methods are discussed in Section 2.2.

When simulations involve a large number of point vortices, as may be required for high Reynolds numbers, high resolution, or long run times, the question of efficient computation becomes important. Both in the early years of research into vortex methods, when computing power was minimal in modern terms, and nowadays, on powerful supercomputers, the limiting factors for simulations are the available memory and processor speed. Methods to increase the efficiency of a simulation are discussed in Section 2.3.

Section 2.4 will discuss the methods of applying traditional vortex methods to flow past multiple bodies and arbitrary geometries, while Section 2.5 will introduce

the rationale for the method presented in this work.

2.1 Vortex sheet roll-up

The first attempt to numerically calculate a flow by solving an approximation to the vorticity equations was carried out without the aid of computer power. Rosenhead [18] considered the behaviour of a vortex sheet in 1931 tracking the position of point vortices with calculations performed by hand. A simple example of a vortex sheet is two inviscid fluids moving parallel to each other at different speeds (Figure 2.1(a)). The vorticity in this type of flow is zero everywhere except for along the thin sheet which separates the two fluids. This vortex sheet is unstable, and will roll up into a spiral structure if it is disturbed.

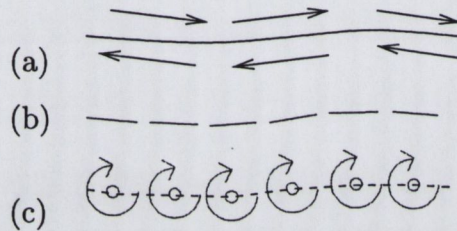


Figure 2.1: Modelling a vortex sheet.

Rosenhead proposed that instead of having a continuous distribution of vorticity over the surface of separation, the surface is broken up into many parts by lines at equal intervals (Figure 2.1(b)), and the vorticity in each of these sections is imagined concentrated into an elemental vortex at its centre (Figure 2.1(c)). In this way, Rosenhead allowed himself to consider a finite number of points instead of a continuous vortex sheet, with the strength of each point being the total circulation of the line segment it replaced. These points, used to replace sections of the vortex sheets, are called *point vortices*. The motion of these point vortices may now be considered, and at any time the line joining them gives an approximation to the vortex sheet.

In Rosenhead's calculations the sheet initially lay along the x axis, with a flow

velocity in the positive x direction above the sheet, and in the negative x direction below the sheet. A small initial sinusoidal disturbance was given to the vortex sheet, and the evolution of the sheet was tracked.

The instantaneous velocities of the vortices in their initial positions were calculated, and the vortices were then moved with this local velocity for a time step Δt to their new positions. The velocities of the vortices induced by all the vortices themselves were calculated again at their new positions, and the vortices convected. Using this first order time-stepping technique, Rosenhead was able to produce a qualitative picture of the process of periodic concentration of vorticity accompanied by a rolling up of the vortex sheet, shown in Figure 2.2, which had been noted experimentally.

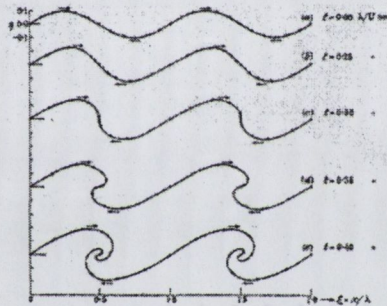


Figure 2.2: Vortex sheet roll-up [18].

Rosenhead's technique was a very successful approximation, however as all calculations were done by hand, he could only use 12 vortices per wavelength of the disturbance. The idea of tracking point vortices was investigated by others, and it was thought that with the advent of computers, more point vortices and smaller time steps could be used, and the accuracy of the results would be increased. However, this was not the case. In 1959 Birkhoff and Fisher [3] tried to reproduce the results Rosenhead obtained, but using 22 vortices per wavelength instead of twelve, and also taking smaller time steps. The results showed that although the vortices did gather in clusters, the paths they followed during the simulation were

irregular, and the point vortices crossed paths, which is physically impossible for a vortex sheet (Figure 2.3). They argued against the idea that vortex sheets roll up smoothly, and indicated that point vortex simulations may be expected to exhibit a tendency towards *randomisation* of position. Viscosity was said to be essential to stabilise the rolling up of real vortex sheets, which will only roll up smoothly without amalgamation by diffusion for one or two complete turns at most.

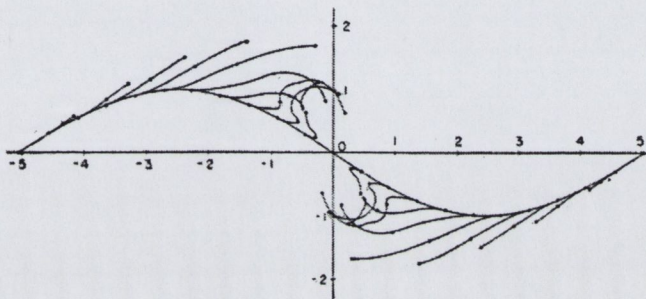


Figure 2.3: Random behaviour during vortex sheet roll up [3].

The challenge of obtaining smooth roll up of a vortex sheet approximated by an array of point vortices remained for a decade, until three different papers offered practical solutions. Moore [16] thought that it was the discretisation of the vortex sheet which was responsible for the chaotic motion. The area of most interest in vortex sheet roll up is the motion near the tips of the vortex sheet. As the vortex sheet rolls up, a spiral shape forms at the tip. If the distance between layers in the roll up, referred to as *turns*, is much less than the typical arc distance between the constituent vortices then there will be instants when vortices on neighbouring turns come very close together. This, Moore claimed, will lead to a spuriously large interaction between this pair which might disrupt the orderly evolution of the system. Thus any attempt to replace the sheet by a finite number of point vortices will cease to be sufficiently adequate near the centre of the spiral, where the number of turns is large. Moore suggested the following approach: “One can suppose that the velocity field of a turn of the spiral is adequately represented

by N or more point vortices, where the integer N has to be determined by trial and error. If a turn proves to contain less than N point vortices, one gives up the attempt to describe it individually, but represents instead the net effect of all such turns by a single vortex of appropriate strength at the centre of the spiral... In effect, vortices which try to get too far into the inner spiral are absorbed into the tip vortex, whereas the outer turns of the spiral have at least N vortices on each." Using this method, Moore obtained a smooth roll up of the vortex sheet (Figure 2.4).

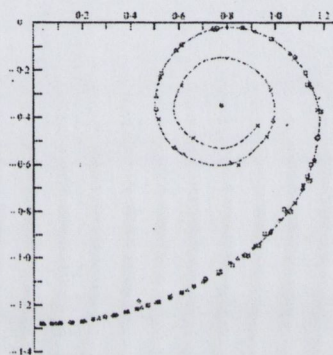


Figure 2.4: Moore's roll up by combining inner vortices [16].

Another successful simulation of vortex sheet roll up was obtained by Christiansen [8] when applying the *cloud-in-cell* (CIC) method to some hydrodynamic problems. The main intended benefit of using CIC with point vortices was increased computational efficiency, and CIC will be discussed further in Section 2.3. However, it also had the effect of avoiding the chaotic motion during the roll up of point vortices. Christiansen replaced a vortex sheet by a finite set of point vortices, and allowed these point vortices to move in a velocity field given on a Cartesian mesh by the CIC method. In the CIC method, a Cartesian mesh is superimposed on the flow field, and the vorticity of the point vortices is credited to surrounding mesh points through a bilinear interpolation. When the vorticity from all of the point vortices has been distributed to the mesh points, the stream function-vorticity equation can be solved on the mesh, and reverse interpolation

used to find the velocity at the locations of the point vortices. The relevant part of this method in avoiding chaotic motion is that the velocity field is obtained from a Cartesian mesh. If two point vortices lie very close to each other, direct vortex-to-vortex calculations will result in spuriously large interactions, as Moore had reported. With the CIC method, however, the vorticity is assigned to the nodes of the mesh before the stream function (and hence velocity) is calculated. Therefore, so long as the nodes of the mesh do not lie very close to each other, the calculated velocities will remain physically realistic, and the simulation may progress without randomisation of the point vortex positions. This method allowed Christiansen to obtain the picture of the roll up of a vortex sheet shown in Figure 2.5.

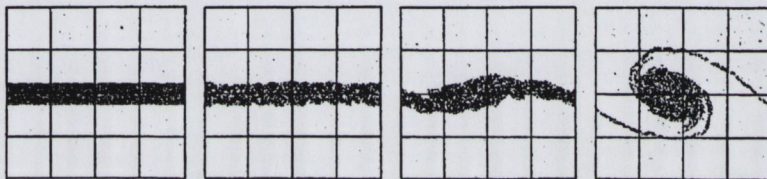


Figure 2.5: Christiansen’s roll up achieved by applying the cloud-in-cell method.

Chorin & Bernard [7] took a different approach to the problem. They recognised that the point vortex approximation to a vortex sheet (or to a continuous vorticity distribution) could not be taken too literally, since a point vortex induces a velocity field which becomes unbounded (very large velocities are predicted near the centre of the point vortex). However, they argued that as soon as the velocity field of the point vortices is smoothed out and made bounded, the approximation becomes reasonable. In fact, they suggested that the hand calculations of people such as Rosenhead offered apparently better results than more modern calculations performed on computers “because the limited accuracy of precomputer calculation had a smoothing effect”.

The problem they attacked was the “unduly large flow” produced by a point vortex when the typical distance between vortices became comparable to the intrinsic characteristic lengths of the problem, and the solution they offered was to

make use of *vortex blobs* (introduced by Chorin earlier that year [5]). Whereas the velocity induced by a point vortex approaches infinity as the distance from its centre approaches zero (Figure 2.6(a)), with vortex blobs a small *cut-off* length is used. The velocity outside this cut-off length is calculated the same way as for a point vortex, with the stream function varying by the log of the distance. Inside the cut-off length, Chorin set the stream function to reduce linearly towards zero, giving a constant induced velocity inside the core (Figure 2.6(b)). This results in a bounded velocity field and removes the problem of singularities due to vortices in close proximity. Later models involving vortex blobs would use Gaussian based velocity distributions in the core, with zero velocity at the centre.

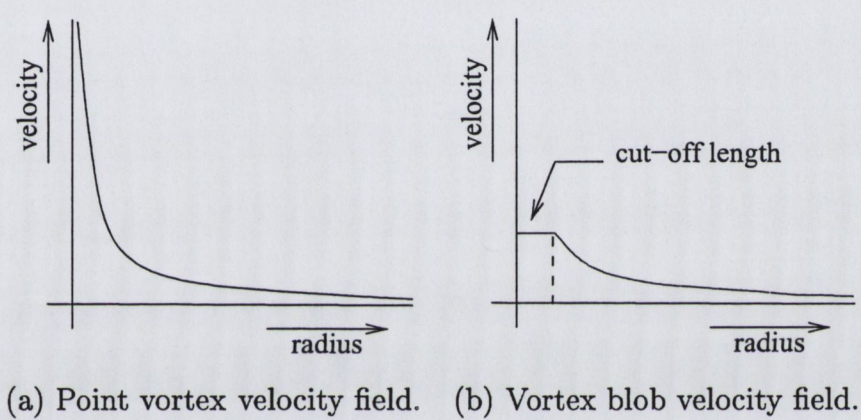


Figure 2.6: Chorin's vortex blob.

Applying vortex blobs to the case of vortex sheet roll up resulted in the chaotic motion near the tip being damped, and a smooth roll up was obtained, as shown in Figure 2.7. The paper concluded that Rosenhead's point vortex approximation is valid and useful, providing the singular character of all the vortices is obviated by some smoothing.

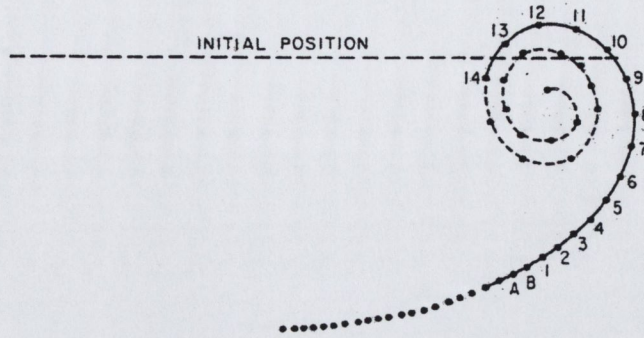


Figure 2.7: Chorin & Bernard's smooth roll up using vortex blobs.

2.2 Including viscosity

Research using point vortices to study the roll up of a vortex sheet had resulted in the development of new tools for vortex methods. The next step was to see if these methods could be extended into the region of viscous flow. The interest in vortex methods lay in their efficient representation of the flow. If the governing equations were to be solved in an Eulerian manner, using a finite-difference method to solve for the flow on a grid, then the mesh spacing would need to be fine enough to capture the behaviour in the boundary layer, and at Reynolds numbers of practical significance the number of mesh points as well as the amount of computational labour required to obtain a solution would be prohibitive [5]. Vortex methods offered a more efficient approach, where interest was focused on the vorticity in the flow. The flow information can be extracted from the vorticity field, and the vorticity field is compact compared to a mesh covering the entire flow domain. By discretising the vorticity field (into either point vortices or vortex blobs) a Lagrangian approach can be used, and the flow development calculated by considering the behaviour of the discretised vorticity field. The efficiencies offered by this vorticity-based Lagrangian method prompted research into extending vortex methods into the region of viscous flow. To do this, the effect of viscosity would have to be included into the calculations in a manner which suited vortex

methods. The effects of including viscosity in the flow are:

- the creation of vorticity at the surface to satisfy the no-slip condition,
- viscous diffusion.

The way in which these effects are included in vortex methods are discussed in the following sections.

2.2.1 Vorticity creation

Chorin investigated slightly viscous two-dimensional flow over a circular cylinder using vortex methods [5]. He enforced the no-slip condition, whereby the fluid velocity at the surface is zero due to the effects of viscosity, by creating a vortex sheet at the surface of the cylinder at each time step. If the strength of this vortex sheet was set so that it induced an equal but opposite velocity to that due to the flow present, then the total tangential velocity at the surface would be zero, and the no-slip condition enforced (Figure 2.8). Chorin suggested using a cut-off value for the circulation assigned to any point vortex. Thus, if the circulation required at a boundary point to satisfy zero tangential velocity was above this cut-off value, a number of point vortices would be created, each of strength less than the cut-off value, and totalling to the required circulation.

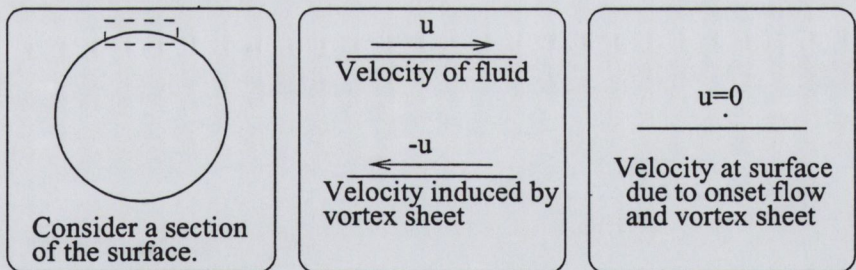


Figure 2.8: Vortex sheet enforcing no-slip condition.

The boundary condition of zero-permeability must also be enforced at the surface of the cylinder. Zero-permeability ensures that the fluid flows around,

and not through, the cylinder. Chorin enforced this condition by applying the source panel method [12]. The source panel method replaces the solid surface with source panels, the strengths of which have been calculated to reduce the normal velocity at the surface to zero, hence there is no flow through the surface and zero-permeability is enforced.

2.2.2 Viscous diffusion

The presence of viscosity in a fluid requires that the viscous diffusion term of the governing equations be solved. The two ways which are most commonly used by vortex methods to include the effects of viscous diffusion are the random walk method, and particle strength exchange.

The random walk method

The random walk method was introduced by Chorin [5]. A more detailed description of this method will follow in Chapter 3, but in short, the method involves modelling the viscous diffusion part of the governing equations by moving the point vortices at each time step by a random distance taken from a Gaussian distribution with a mean of zero, and a variance which is related to the length of the time step and the Reynolds number, hence statistically satisfying the effects of viscous diffusion [28].

Chorin used vortex blobs to overcome singularities, source panels for zero-permeability, and created a vortex sheet at the surface to enforce the no-slip condition. The simulation was advanced through time by discretising this vortex sheet into a number of vortex blobs, and using a fractional step algorithm, whereby the vortex blobs were moved in their local velocity field to solve the convective part of the governing equations, and then a random walk was added to their positions to solve the diffusive part of the governing equations. The new velocity field was calculated, the boundary conditions applied, and the process repeated. Using this

method, Chorin was able to obtain results for the drag coefficient which were in good agreement with experimental results, and also achieved a natural development of boundary layer separation at the surface.

Particle strength exchange

Vortex methods which use vortex blobs and the particle strength exchange scheme to solve for viscous flow are discussed in detail by Cottet & Koumoutsakos [9]. An outline of the method is offered here.

Viscous diffusion causes a change in the distribution of vorticity in a flow, while maintaining constant total circulation. The random walk method accounts for this change by moving the point vortices in a manner which statistically reproduces the effects of diffusion. An alternative to this would be to allow the point vortices to remain stationary while accounting for diffusion.

In the particle strength exchange scheme, the effects of viscous diffusion are modelled by changing the *strengths*, and not the *locations* of the point vortices. The viscous diffusion term is solved to find the change in the strengths of the point vortices required to give the new vorticity distribution after diffusion has taken place, while maintaining a constant total circulation in the flow. Convergence of this method requires that particles overlap enough to give a correct sampling of the (Gaussian) diffusion kernel. This calculation is included in the solution of the convective equation so that there is no longer any operator-splitting of the governing equations into a convective and diffusive term to be solved separately for each time step. Time discretisation can be done through classical time-stepping schemes, with no need for a fractional time step.

2.3 Increasing efficiency

Vortex methods could now be applied to viscous flows, and researchers started to consider the problem of obtaining flow simulations with as efficient a usage of

processor power and memory as possible. The two main areas of interest discussed here involve the calculation of the velocity field, and the treatment of vortices crossing a solid surface.

2.3.1 Velocity field calculation.

Christiansen [8] considered a collection of point vortices lying inside a rectangular region covered by a Cartesian mesh. He assumed that each point vortex possessed uniform vorticity within a unit square such that the corresponding unit amount of vorticity can be distributed to surrounding mesh points through a bilinear interpolation. At each time step, the vorticity of the point vortices is attributed to the nodes of the cell in which it is contained. He proposed that the vorticity be *spread* from the point vortices onto the nodes of the mesh by using the cloud-in-cell (CIC) method, sometimes referred to as the area-weighting technique. Using this technique, the vorticity is spread onto the nodes of its cell by means of a bilinear interpolation based on the sizes of the areas the point vortex forms within the cell.

Once all of the point vortices have been spread in this way onto the nodes of the mesh, the governing stream function - vorticity equation can be solved on the mesh using standard Eulerian mesh techniques, such as finite-difference. The velocity of each point vortex in this flow field can now be obtained from the stream functions at the mesh points by interpolation, and the simulation advanced.

This reduces the cost of the flow field calculation for N point vortices from $O(N^2)$ (calculating the effect of every point vortex on every other point vortex) to a number of computations that is roughly linear in the number of vortices and mesh points, $O(N + M)$ [15]. This forms the basis for the present work, and so a more detailed discussion of this method is given in Section 4.2.

An alternative method for efficient calculation of the velocity field due to N vortices is often used in modern vortex blob simulations [9]. The method developed by Reenact & Rohklin [10] offers $O(N)$ operation counts. The basic idea of the

method is to decompose the vortex blobs spatially into clusters of particles and build a hierarchy of clusters. This hierarchy is created by dividing the flow field into a coarse regular mesh. Any cell of the mesh which contains more than a certain number of vortices is subdivided into smaller cells, and the number of vortices inside the new cells is again counted. This process is repeated, going from the coarse cell size to finer and finer cells, until no cells contain more than the cut-off value of vortices (Figure 2.9).

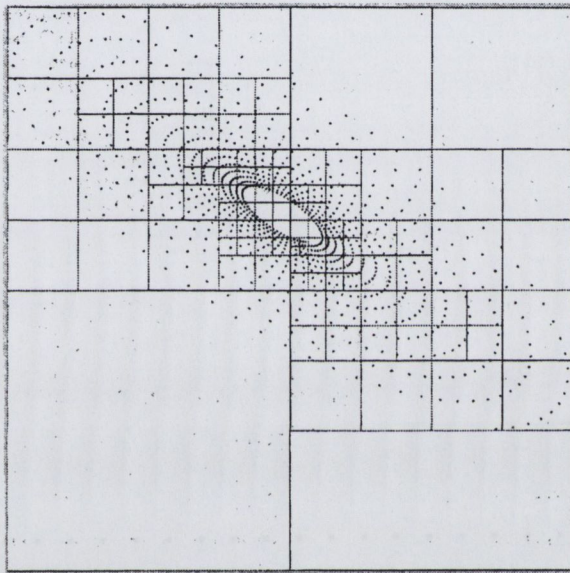


Figure 2.9: A hierarchy of clusters for a spiral distribution of vortex particles [9].

The velocity field is now calculated making use of this hierarchy. Cell to cell velocity calculations are carried out by treating distant cells as single vortices with strengths equal to the combined strengths of all vortices in the cell. Velocity calculations inside a cell are calculated for the direct interaction between all the vortices, hence a core model is essential, as vortices in the same cell may be close together. Although this method may offer up to $O(N)$ operation counts, it does have the fundamental weakness of requiring a re-mesh of the hierarchical cell structure at each time step. The application of this method, along with vortex blobs and the particle strength exchange method, was used by Koumoutsakos & Leonard [13]

to achieve high-resolution simulations of the flow around an impulsively started cylinder. The results they obtained are considered to be a benchmark for vortex methods, and will be compared to the results obtained in this work in Chapter 7.

2.3.2 Problems near solid boundaries

Chorin [6] presented a vorticity generation method to overcome the problem of a slow rate of convergence near boundaries. In his earlier paper, the tangential boundary condition was satisfied on the surface by creating a vortex sheet of strength u per unit length of the wall, where u is the tangential velocity at the wall. This vortex sheet was then broken up into elements and allowed to participate in the subsequent motion of the fluid. The vorticity elements which crossed the wall were deleted, and the effect of their vorticity recreated by the tangential surface velocity calculated at the next step. This formulation converged as the time step approached zero, but convergence was slow.

The method which Chorin proposed extended the velocity field across the solid surface. The tangential velocity inside the surface would be equal and opposite to the velocity outside the surface, giving a jump of velocity equal to $2u_0$ across the surface. Therefore, in order to satisfy the no-slip condition and reduce the tangential velocity on the surface to zero, a vortex sheet with strength $2u_0$ per unit length of the wall would need to be created (Figure 2.10(a)).

Vortices were convected by their local velocity field for the time step to solve the convective term of the governing equations, and a random walk added to their positions to solve the diffusive term. After these operations, some vortex particles may have moved to the inside of the solid surface, (Figure 2.10(b)). Any of the newly created vortices which cross the boundary are deleted, (Figure 2.10(c)), thus on average the amount of newly created point vortices which remain in the simulation will be half of those created, or a vortex sheet of strength u_0 , which is required to maintain the boundary condition from the external field to the surface.

Any old point vortices which cross the boundary are reflected back to their mirror-image location in the flow field (Figure 2.10(d)). Hence, all vorticity which enters the simulation remains in the simulation, and at each time step the correct amount of new vorticity is added to the simulation.

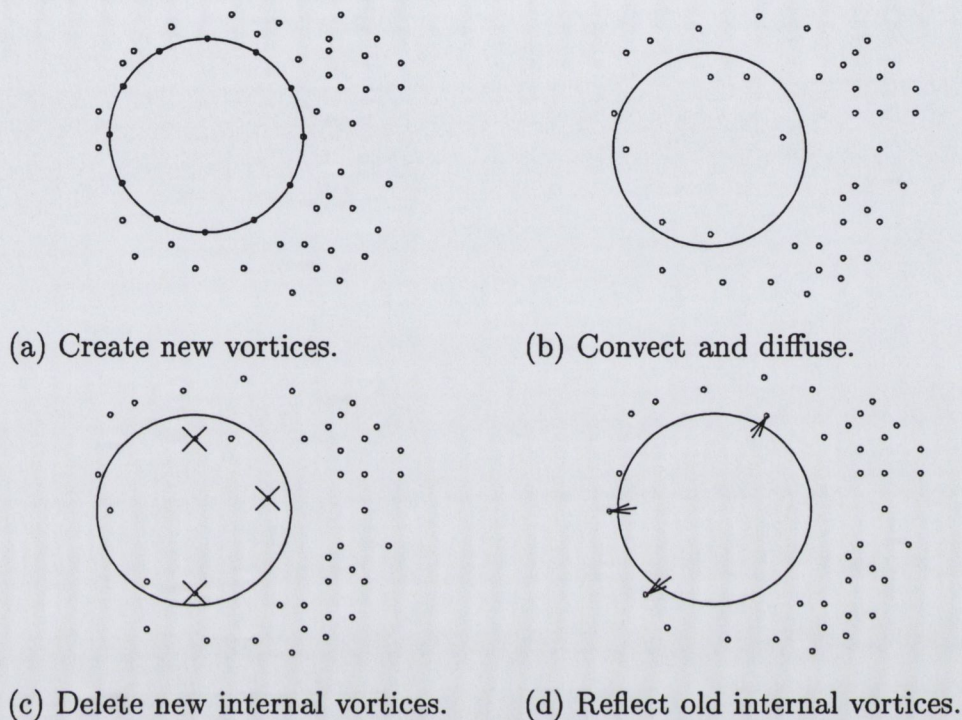


Figure 2.10: Point vortices crossing the solid boundary.

Stansby & Dixon [26] used Chorin's improved surface treatment of vortices which cross the boundary, except they perform the rather logical reflection of a single strength vortex sheet (u) rather than Chorin's double strength sheet idea ($2u$): "This simulation of diffusion may be conveniently effected by diffusing a sheet of the original strength and reflecting any sheets which move inside the surface on to their mirror-image positions on the outside. Sheets which were not initially on the surface but which cross due to their random walk are also reflected back. On average, this will balance those which would have crossed the surface from the inside in the course of their random walk. As well as being computationally desirable, these reflections ensure that the exact amount of vorticity created by

the zero-slip condition is maintained in the flow.” This simply treats all vortices in the same way after diffusion.

As computers became more powerful, research into the vortex method continued. Smith and Stansby [23] published a comprehensive paper on their results for impulsively started flow around a circular cylinder. They used Chorin’s random vortex method and Christiansen’s cloud-in-cell method on a single, radially expanding polar mesh, but also implemented a different scheme to calculate the surface conditions, whereby the value of the stream function on and inside the surface was set to a fixed value. Vortices which crossed the boundary were reflected to their mirror-image positions. Simulations were run in a range of Reynolds numbers from 2.5×10^2 to 10^5 . The number of vortices introduced per time step needed to be increased as the Re increases (and for longer computations) in order to obtain accurate velocity predictions. They compared their calculations with analytical solutions, Eulerian finite-difference solutions and experimental flow visualisations, and there was good agreement between the results.

Smith & Stansby later went on to introduce an efficient method of dealing with vortices which crossed the surface [24]. Vortex particles which crossed the solid boundary were removed from the calculation, and their circulation was included in the circulation of particles created at the surface at the next time step, with an adjustment made to their random walk. This method of absorbing particles rather than reflecting them reduced the overall number of particles remaining in the simulation with no loss of accuracy. Hence more particles could be introduced at the surface at each time step than could be used with the reflect method, giving better accuracy and efficiency. Absorbing internal vortices is used in the present calculations, and the method is explained further in Section 3.7.3.

2.4 Multiple bodies and arbitrary geometries

All simulations which use the CIC method solve the Poisson equation on a regular grid, using a finite-difference approach. This introduces problems when there are certain areas in the flow where a fine grid resolution is required, or if the flow over multiple bodies is to be computed. The two methods which have been used in the past to allow the regular grids required by CIC to be adapted for use in these conditions involve using overlapping meshes, and using conformal transformations.

2.4.1 Overlapping meshes

The paper by Stansby & Dixon [26] (which reflected all point vortices crossing the boundary, as discussed above) used more vortices at the surface of the cylinder than Chorin had (120 instead of 20), applied the cloud-in-cell method to handle the larger number of vortices, and maintained the zero-velocity condition on the surface by using only a vortex sheet (Chorin used a source sheet as well as creating vorticity).

Stansby & Dixon also used overlapping meshes in their simulation. The cell size of the cloud-in-cell mesh determines the lower limit of the resolution of the velocity field calculation. A very small dimension is required on the cylinder boundary to model the behaviour in the boundary layer, while a large dimension will be adequate to transport large vortex structures which have convected large distances from the body. To achieve this overlapping meshes were used and solved together. An inner mesh, just large enough to cover the body, was used to provide sufficient definition for the boundary. The middle mesh covered the *formation region*, where vorticity shed from the surface forms large-scale structures, which are eventually convected downstream on the outer mesh. The velocity calculation was performed from the outside mesh towards the inner mesh. Boundary conditions for an internal mesh were obtained from its overlapping mesh. For convenience, mesh points on larger meshes coincided with the smaller mesh points where they

overlapped. Poisson's equation was thus solved three times and the velocity of a vortex was calculated from the stream function distribution of the mesh with the smallest possible cell size. As the surface boundary condition was satisfied by a vortex sheet, and multiple meshes were used to calculate the flow in different areas of the flow field, this method could be applied to multiple bodies and arbitrary geometries which need not follow mesh contours.

Stansby & Dixon also mention that the CIC method predicts velocities due to a vortex sheet accurately on the sheet and at distances from the sheet greater than about one cell size. The influence of vortex sheets within one cell size of the surface, therefore, may not be represented accurately at the surface. Tangential velocities for the boundary integral (vortex sheet) calculation were calculated a small distance inside the surface, and this idea is followed in the present calculations, where the circulation strengths of newly created vortices are defined from the tangential velocity calculated inside the surface.

Slaouti & Stansby [22] investigated the flow around two circular cylinders in side-by-side and tandem arrangements, using multiple meshes. This required some thought to be put into the solution of the governing equations on overlapping meshes with the flow field boundary conditions. Figure 2.11 shows a typical arrangement of overlapping meshes used.

The coarse outer mesh extends to a large distance away from the cylinders, where uniform flow may be assumed. The stream function on the outer boundary is thus simply given by $\psi = U_0 y$. The outer mesh is solved first, giving the stream function boundary values for the finer intermediate mesh by linear interpolation. This mesh is then solved, giving the stream function values on the boundaries of the polar meshes again by interpolation. A velocity is obtained from the stream function values on the mesh with the smallest size available. The boundary condition for the cylinder surfaces is enforced by using a coarse vortex-sheet boundary-integral method to enforce zero normal velocity and zero slip inside the cylinders. The value of the stream function is constant within each cylinder, and can now

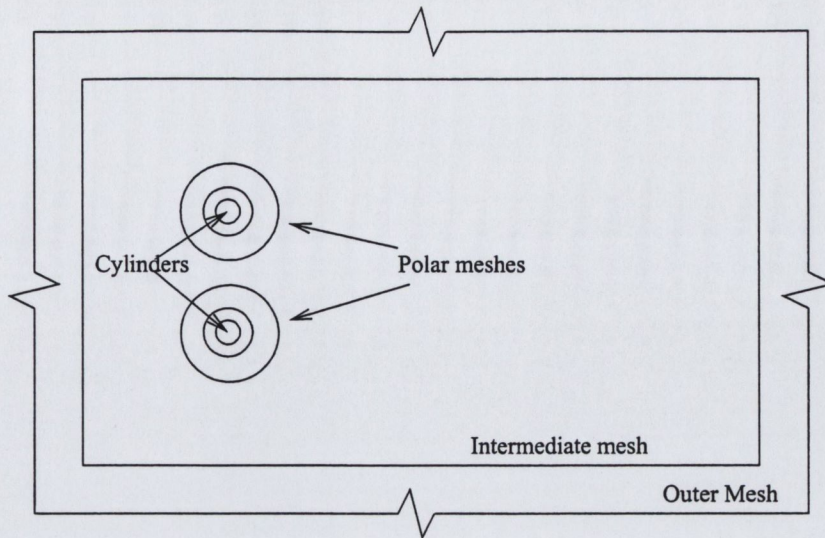


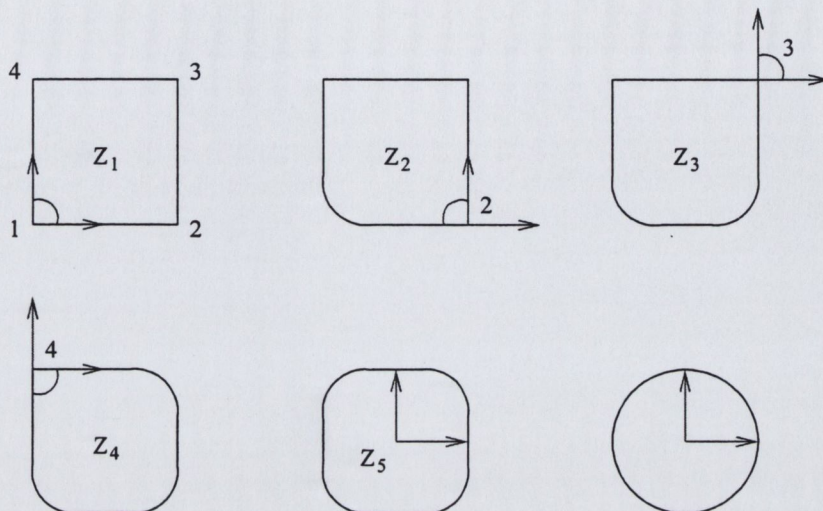
Figure 2.11: Overlapping meshes for vortex-in-cell computation.

be calculated at the cylinder centre and applied to the surface control nodes. The coarse vortices are then removed, and a more accurate tangential velocity calculated to determine the circulation of the vorticity to be created at the surface. Their simulations of flow over two cylinders in both side-by-side and tandem configurations agreed with experimentally observed features and allowed some further understanding of the interactions in the flow. Multiple meshes were also used in this way to study blockage effects for flow over a circular cylinder [27].

2.4.2 Conformal transformation

Smith & Stansby [25] extended the random-vortex method to prismatic bodies of arbitrary cross-section. Using a conformal transformation which was divided into a number of separate transformations (Figure 2.12) they were able to calculate the oscillatory flow past cylindrical bodies with a number of sharp edges, such as the flow past a sharp-edged square. The conformal transformation maps the region outside the cylinder in the complex physical plane onto the region outside the unit circle in the transform plane, and the vortex calculations were performed in the

transform plane, as if for a circular cylinder. Reasonable agreement with experiment was attained, although it was noted that the real flows have three-dimensional components which were not modelled in the two-dimensional simulations.



The sharp edges are removed by successive applications of the Karman-Trefftz transformation. The resulting near-circle is mapped onto a unit circle using the Theodorsen-Garrick transformation [25].

Figure 2.12: Conformal transformations.

2.5 The present work

The development of the CIC method resulted in a combined Eulerian-Lagrangian technique, whereby the governing equations are solved by convection in the local flow field, diffusion by a Gaussian random walk, and the flow field itself is calculated by spreading the vorticity onto a regular grid. The advantages of solving the flow by tracking the vorticity field is offered, along with the efficiency of using a fixed grid to solve the Poisson equation, and hence calculate the interaction of a large number of vortices. The problem which faced this method, however, was that the CIC method was only applied for regular Cartesian or polar geometries, where point

location in the mesh is trivial. For simulations of the flow over multiple bodies or arbitrary geometries techniques such as conformal transformations (requiring the problem to be formulated in the transformed plane), or multiple meshes (requiring mesh interaction schemes) needed to be used.

In the work presented here, a new technique is proposed which allows the method of point vortices to be used efficiently and easily on multiple bodies or arbitrary geometries by extending the idea of the CIC method so that it can be applied to a single unstructured mesh.

Chapter 3

The random vortex method

The flow of an incompressible Newtonian fluid is governed by the continuity and the Navier-Stokes equations. The continuity equation is an expression of the conservation of mass, and in two dimensions becomes

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \quad (3.1)$$

The Navier-Stokes equations concern the momentum equation and the viscous stresses acting in the fluid. They were derived independently by Claude Navier in 1827 and George Stokes in 1845 and are expressed in two dimensions as

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \quad (3.2)$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = -\frac{1}{\rho} \frac{\partial p}{\partial y} + \nu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) \quad (3.3)$$

There are a number of books which discuss the derivation of these governing equations (e.g. [2],[21],[14]).

In the random vortex method, the governing equations of the flow are solved by considering the creation, interaction and motion of point vortices. Before this method is described, a short review of point vortices is presented.

3.1 Vorticity

Consider the flow with spatial variations in the velocity as shown in Figure 3.1. If a small cogwheel is placed in the fluid, it will begin to rotate. At point A in the figure the cogwheel rotates counterclockwise, while at B, clockwise. This rotational motion is characterised by the *vorticity*, defined as the curl of the velocity field at that point. In two-dimensional flow, this can be considered as a scalar, as the direction of the vorticity vector will always point normal to the 2D plane.

$$\begin{aligned}
 \omega &= \text{curl}(\vec{v}) \\
 &= \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right) \times (u, v) \\
 &= \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}
 \end{aligned} \tag{3.4}$$

The vorticity, ω (units s^{-1}), is equal to twice the angular velocity of the cogwheel. Vorticity is defined in the present work as positive for counterclockwise rotation, and negative for clockwise rotation.

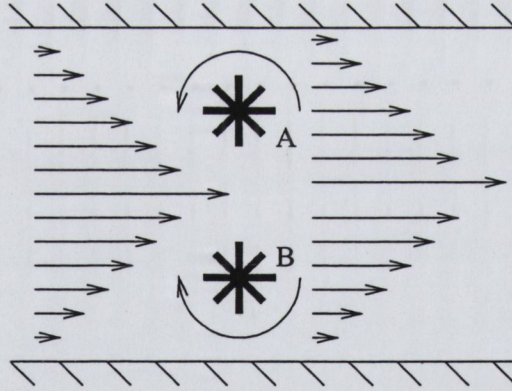


Figure 3.1: Vortex motion in a viscous fluid.

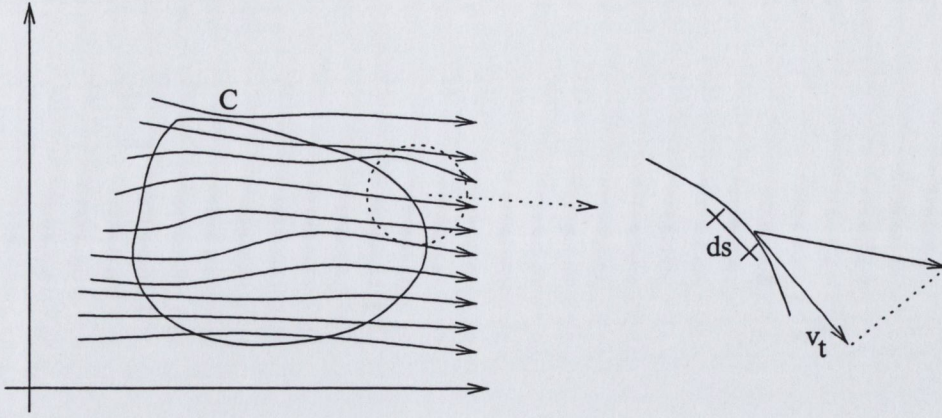


Figure 3.2: Closed loop C for determining circulation.

3.2 Circulation

Figure 3.2 shows a closed loop C in a flow. The *circulation*, Γ , is defined as the *counterclockwise* line integral, around a closed curve C , of arc length ds times the velocity component tangent to the curve.

$$\Gamma = \oint_C \vec{v} \cdot d\vec{s} = \oint_C (u dx + v dy) \quad (3.5)$$

where $d\vec{s} = (dx, dy)$ is the incremental distance along the loop. Non-zero circulation implies a component of *rotation* of flow in the system. An alternative definition of circulation, chosen in aerodynamics for convenience in the analyses of airfoil and wing theory, carries out the integral in a clockwise direction, which would be indicated by a minus sign in Eq. (3.5). In this work, for mathematical consistency, positive circulation is defined in a counterclockwise direction.

Another way of perceiving circulation is by considering the circuit of integration to consist of a large number of rectangular cells of area $\delta x \delta y$, as shown in Figure 3.3. Applying the integral $\Gamma = \oint_C (u dx + v dy)$ for the element which contains point $P(x, y)$, where the velocity is (u, v) , gives

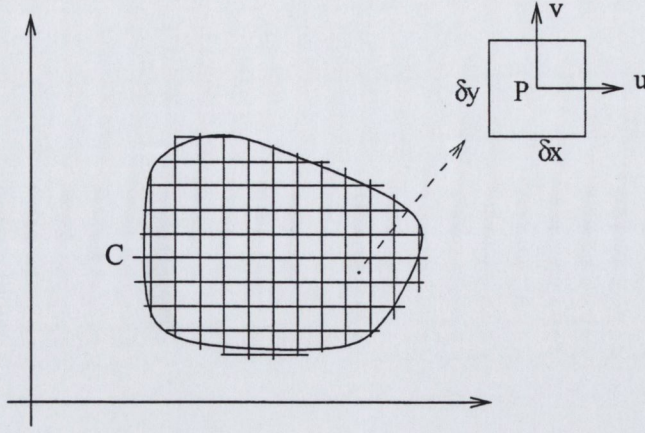


Figure 3.3: Circulation by cells.

$$\begin{aligned}\Delta\Gamma &= \left(v + \frac{\partial v}{\partial x} \frac{\delta x}{2}\right) \delta y - \left(u + \frac{\partial u}{\partial y} \frac{\delta y}{2}\right) \delta x - \left(v - \frac{\partial v}{\partial x} \frac{\delta x}{2}\right) \delta y + \left(u - \frac{\partial u}{\partial y} \frac{\delta y}{2}\right) \delta x \\ &= \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right) \delta x \delta y\end{aligned}$$

The sum of the circulations of all the areas is clearly the circulation of the circuit as a whole because, as the circulation of each cell is added to that of the neighbouring cell, the contributions of the common sides disappear.

Applying this argument from cell to neighbouring cell throughout the area, the only sides contributing to the circulation when the circulation of all areas are summed together are those sides which actually form the circuit itself. This means that for the circuit as a whole we are left with an expression of Stokes' theorem:

$$\Gamma = \underbrace{\iint \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right) dx dy}_{\text{over the area}} = \underbrace{\oint (u dx + v dy)}_{\text{around the circuit}} \quad (3.6)$$

A relationship between circulation and vorticity can be obtained by considering the circulation of a fluid element, and the definition of vorticity given by Eq. (3.4):

$$\Gamma = \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right) \delta x \delta y = (\text{vorticity})(\text{area of element}) \quad (3.7)$$

Thus vorticity can be considered to be the local circulation per unit area.

3.3 The point vortex

A point vortex can be thought of as a point with a defined circulation strength. A point vortex acting on its own in a two-dimensional plane will result in circular streamlines, as shown in Figure 3.4.

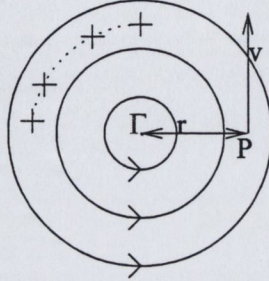


Figure 3.4: Flow induced by a point vortex.

A point vortex is, in fact, something of a misnomer, as it is really a point of defined *circulation*, and not vorticity. However, point vortex is the term commonly used in the literature and so it will be used here. A single point vortex will induce a circular flow around it, however this flow is not *rotational*. An infinitesimally small cogwheel placed in the flow would travel in a circular path around the point vortex, but would not rotate around its own axis. The only position in which the cogwheel would rotate on its own axis would be if it were placed with its centre on the point vortex. By this example, it can be seen that the vorticity of the flow is zero everywhere, except at the point vortex itself, where (due to the infinitely small area of a point vortex) the vorticity is infinite. The induced velocity at a point P will depend on the circulation of the vortex, Γ , and the distance from the point vortex to P , r

$$\vec{v}_\theta = \frac{\Gamma}{2\pi r} \quad (3.8)$$

This is often referred to as the Biot-Savart law.

3.4 Vorticity transport equation

Vortex methods are used to solve the governing equations of fluid flow by concentrating on the behaviour of the vorticity in the flow. To do this, the governing equations need to be recast to describe the behaviour of the vorticity. The vorticity is the curl of the velocity, Eq. (3.4), and so to state the Navier-Stokes equations in terms of vorticity, differentiation of Eq. (3.3) with respect to x and Eq. (3.2) with respect to y must be carried out

$$\begin{aligned} \frac{\partial}{\partial t} \left(\frac{\partial v}{\partial x} \right) + u \frac{\partial}{\partial x} \left(\frac{\partial v}{\partial x} \right) + \frac{\partial u}{\partial x} \frac{\partial v}{\partial x} + v \frac{\partial}{\partial x} \left(\frac{\partial v}{\partial y} \right) + \frac{\partial v}{\partial x} \frac{\partial v}{\partial y} \\ = -\frac{1}{\rho} \frac{\partial^2 p}{\partial x \partial y} + \nu \left(\frac{\partial^2}{\partial x^2} \left(\frac{\partial v}{\partial x} \right) + \frac{\partial^2}{\partial y^2} \left(\frac{\partial v}{\partial x} \right) \right) \\ \frac{\partial}{\partial t} \left(\frac{\partial u}{\partial y} \right) + u \frac{\partial}{\partial y} \left(\frac{\partial u}{\partial x} \right) + \frac{\partial u}{\partial x} \frac{\partial u}{\partial y} + v \frac{\partial}{\partial y} \left(\frac{\partial u}{\partial y} \right) + \frac{\partial v}{\partial y} \frac{\partial u}{\partial y} \\ = -\frac{1}{\rho} \frac{\partial^2 p}{\partial x \partial y} + \nu \left(\frac{\partial^2}{\partial x^2} \left(\frac{\partial u}{\partial y} \right) + \frac{\partial^2}{\partial y^2} \left(\frac{\partial u}{\partial y} \right) \right) \end{aligned}$$

exchange the order of differentiation and subtract to get

$$\begin{aligned} \frac{\partial}{\partial t} \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) + u \frac{\partial}{\partial x} \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) + v \frac{\partial}{\partial y} \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) + \\ \frac{\partial v}{\partial x} \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) - \frac{\partial u}{\partial y} \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) \\ = -\frac{1}{\rho} \left(\frac{\partial^2 p}{\partial x \partial y} - \frac{\partial^2 p}{\partial x \partial y} \right) + \nu \left(\frac{\partial^2}{\partial x^2} \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) + \frac{\partial^2}{\partial y^2} \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) \right) \end{aligned}$$

Using the expressions for vorticity in Eq. (3.4) and continuity in Eq. (3.1) this equation reduces to the vorticity transport equation

$$\frac{\partial \omega}{\partial t} + u \frac{\partial \omega}{\partial x} + v \frac{\partial \omega}{\partial y} = \nu \left(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2} \right) \quad (3.9)$$

The vorticity transport equation is an expression of the Navier-Stokes equations which intrinsically includes the continuity equation. This is the governing equation for the fluid flow. In order to solve this equation, however, the velocity field (u, v)

must be calculated. The velocity field can be obtained from the vorticity field by use of the stream function, ψ .

3.5 Stream function-vorticity relation

A *streamline* is a curve along which the tangent at every point is in the same direction as the flow velocity vector. The *stream function* is defined as being constant along a streamline, and is given by

$$u = \frac{\partial \psi}{\partial y}, \quad v = -\frac{\partial \psi}{\partial x} \quad (3.10)$$

If the stream function was related to the vorticity then the velocity field could be calculated from the vorticity field. The relationship between the two is straightforward. The expression for vorticity given by Eq. (3.4) relates vorticity to velocity.

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}$$

By substituting stream functions for velocities,

$$\frac{\partial}{\partial x} \left(-\frac{\partial \psi}{\partial x} \right) - \frac{\partial}{\partial y} \left(\frac{\partial \psi}{\partial y} \right) = \omega$$

the stream function-vorticity relation can be obtained:

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega$$

This is often seen written in the form

$$\nabla^2 \psi = -\omega \quad (3.11)$$

which is the Poisson equation.

3.6 The operator-splitting scheme

The governing equations which are needed to describe fluid flow are given by the vorticity transport equation, Eq. (3.9), and the stream function-vorticity equation, Eq. (3.11). The next challenge is to find some technique that can be used to efficiently solve these equations. Such a technique, the random vortex method, was proposed by Alexandre Chorin in 1973. Chorin [5] has shown that an operator-splitting scheme can be used to solve the vorticity equation. To do this, the equation is separated into a convection term and a diffusion term:

$$\frac{\partial \omega}{\partial t} = \left[-\frac{\partial \psi}{\partial y} \frac{\partial \omega}{\partial x} + \frac{\partial \psi}{\partial x} \frac{\partial \omega}{\partial y} \right]_C + \left[\nu \left(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2} \right) \right]_D \quad (3.12)$$

The processes of convection and viscous diffusion are denoted by the suffices C and D , respectively. Convection is solved by moving the point vortices in the velocity field, while diffusion is solved by adding a random walk to their positions. The final position of each point vortex at the end of a time step is given by the combination of the diffusion and convection processes. The simulation then advances onto the next time step, and the new velocity field is calculated and the boundary conditions applied. This time-stepping procedure by which the governing equations are solved is called the random vortex method. If the random vortex method were applied to a *continuum* of vorticity, with an infinitesimal time step, it would offer an *exact* solution to the viscous equations of motion [20]. No approximation is involved until the infinite system is reduced to a finite number of point vortices, with a discrete time step. When this is done, the solution accuracy depends on the size of the time step, and the number of point vortices in the simulation. However, as point vortices are being used as a discretised approximation of the spatially compact vorticity field, and the evolution of this vorticity field is being calculated in a Lagrangian manner by following the point vortices, coarser discretisation is required than would be with a primitive variable Eulerian scheme, which solves velocity formulations on the nodes of a meshed flow domain.

3.6.1 Vorticity convection

The convection part of the vorticity transport equation is solved by an inviscid calculation, in which vortex particles are convected in their local velocity field, which can be obtained by solving the Poisson equation in stream function and vorticity, $\nabla^2\psi = -\omega$. The most straightforward way of doing this is to use Euler's method of discrete integration, which is a first-order approximation. Starting with vortex position (x, y) , the point vortex is moved according to the local velocity field, as in Eq. (3.13) and Figure 3.5.

$$x(t + \Delta t) = x(t) + u\Delta t \quad (3.13)$$

$$y(t + \Delta t) = y(t) + v\Delta t$$

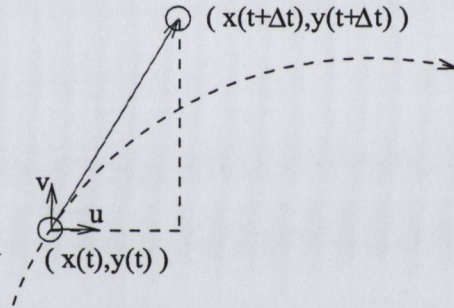


Figure 3.5: Euler's method.

It has been suggested (Smith & Stansby [23]) that as the Reynolds number of the flow increases, the accuracy of the random vortex method depends on the accuracy of the solution of vorticity convection. To obtain higher accuracy, therefore, a second-order Runge-Kutta convection scheme could be used. The point vortices are convected in their local velocity field for the time step Δt , as for the first-order case (Figure 3.6(a)). The Poisson equation is solved to obtain the new velocity field, and convective distances calculated for Δt once more. The final vortex position is then set as the average of the two convective distances, as in Figure 3.6(b) and Eq. (3.14).

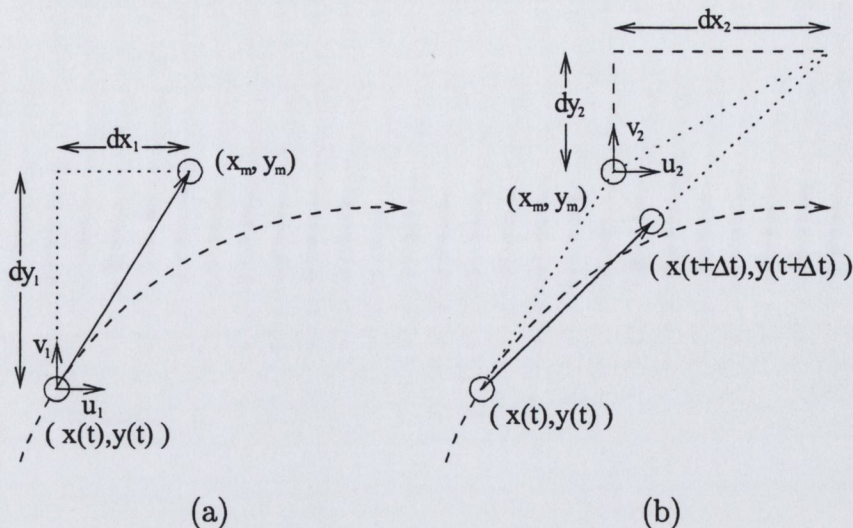


Figure 3.6: Second-order Runge-Kutta convection scheme.

$$dx_1 = u_1 \Delta t \quad , \quad dy_1 = v_1 \Delta t \quad (3.14)$$

$$dx_2 = u_2 \Delta t \quad , \quad dy_2 = v_2 \Delta t$$

$$x(t + \Delta t) = x(t) + \frac{1}{2}(dx_1 + dx_2)$$

$$y(t + \Delta t) = y(t) + \frac{1}{2}(dy_1 + dy_2)$$

This second-order accuracy, however, requires a second solution of the stream function-vorticity equation to obtain the new velocity field at positions (x_m, y_m) . In the present work, the simpler first-order scheme is used when convecting the point vortices. Although it may not offer the same resolution of the velocity field, it is more computationally efficient, and allows simulations to be calculated in less time than would be required by the second-order scheme, while still capturing the main features of the flow, as will be shown by the results in Chapter 7.

3.6.2 The random walk

The viscous diffusion term of the vorticity transport equation is solved following the method introduced by Chorin [5]. The effect of viscosity is simulated by

walking the point vortices through a distance taken at random from a Gaussian distribution. This is achieved by changing the position of each point vortex in the direction of the two axes by an amount, η , chosen at random from a Gaussian distribution with standard deviation $\sqrt{2\nu\Delta t}$ so that:

$$\begin{aligned}x(t + \Delta t) &= x(t) + \eta_1 \\y(t + \Delta t) &= y(t) + \eta_2\end{aligned}\tag{3.15}$$

where η_1 and η_2 are the two random walks. This method can be shown to be a statistically accurate solution to diffusion term of the governing equation [9], while offering the advantages of being simple to apply, and maintaining the Lagrangian character of vortex methods.

3.7 Flow around a circular cylinder

The random vortex method can be used to calculate how a flow develops by tracking points of prescribed circulation. Although in principle this method may be applied to any arbitrary shape, and indeed to more than one body, the discussion here will use the case of flow around a circular cylinder. This allows an easier explanation of the methods involved. Furthermore, as impulsively started flow over a circular cylinder is the most commonly used benchmark case in the literature, it also allows results to be compared to other published computational and experimental results.

If the random vortex method is to be applied to the case of flow over a circular cylinder, or indeed any solid boundary, there are certain boundary conditions which must be applied. These are:

- the zero-permeability condition, which ensures that the fluid does not flow through the solid,

- the no-slip condition, which enforces zero tangential fluid velocity at the surface.

The case of point vortices crossing the solid boundary during diffusion or convection must also be taken into account.

3.7.1 The zero-permeability boundary condition

The first boundary condition which needs to be enforced ensures that the fluid travels *around* and not *through* the cylinder, as in Figure 3.7. The condition that the fluid must not flow through the surface is known as the *zero-permeability* condition, and results in the normal velocity of the fluid being zero at the surface. This condition can be enforced by setting the value of the stream function at the surface to be a constant, hence making the surface a streamline, and ensuring no flow will cross it. Setting a variable to equal a constant value along a boundary is known as a *Dirichlet* boundary condition. Alternatively, a panel method can be used, whereby the strength of a collection of source, or vortex, panels is calculated from the onset velocity and the geometry of the solid, so as to ensure that the flow does not cross the boundary [12]. This is effectively a Neumann condition as the surface normal flux is set to zero.

3.7.2 The no-slip boundary condition

In order to understand the effect of viscosity for flow past a solid boundary, and how it can introduce vorticity into the flow, consider uniform laminar flow past a plate. For an inviscid fluid, shown in Figure 3.8(a), the velocity far from the plate will be the same as the velocity near the plate surface. As the viscosity is zero, there is no shear stress between the plate and the fluid. All of the layers of fluid move with the same velocity, and so there is no velocity gradient, and hence no rotational effect between the layers. No vorticity is created at the surface.

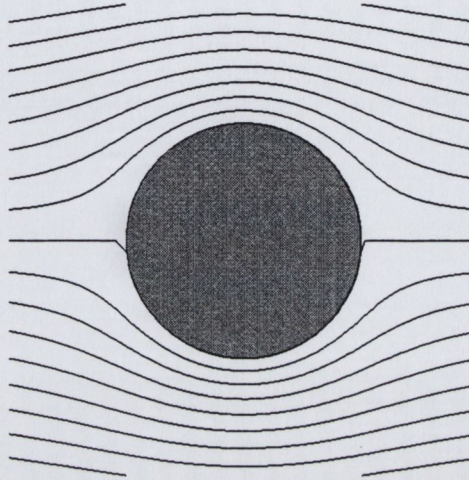


Figure 3.7: Streamlines around a cylinder for starting flow.

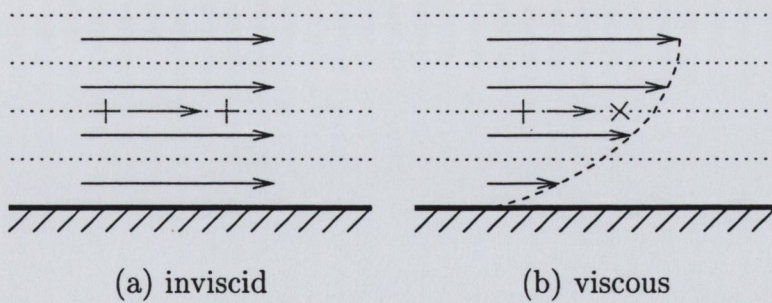


Figure 3.8: Uniform laminar flow past a plate.

If the fluid is viscous, as shown in Figure 3.8(b), the situation is different. The velocity of the fluid at the surface is zero, as viscosity causes the fluid to stick to the solid surface. This is called the *no-slip* condition. In terms of the stream function, zero tangential velocity at a boundary will occur when the normal gradient of the stream function equals zero at the boundary. Defining the normal gradient of a flow variable on a boundary is called a *Neumann* boundary condition. Viscosity also causes an interaction between adjacent layers of the fluid so that a velocity profile exists in a region increasing from zero at the surface to the free-stream velocity some distance from the surface. This region is the boundary layer. The velocity profile in the boundary layer introduces a rotational effect, hence vorticity is created. The vorticity transport equation states that the vorticity will be diffused and convected with time, hence vorticity moves away from the solid surface and downstream. New vorticity is constantly created in the boundary layer due to the no-slip condition.

In the simulation, the no-slip condition is enforced at the surface by using a vortex sheet of the appropriate strength to ensure zero tangential velocity, and this vortex sheet is then discretised into point vortices which diffuse into the flow. A vortex sheet is a sheet which has a defined circulation strength at every point of its surface. Figure 3.9 shows the flow for a perpendicular cross-section through a vortex sheet with a defined circulation per unit length, γ . The circulation for the dashed control volume of the vortex sheet can be calculated as follows:

$$\Gamma = \gamma ds = u_1 ds + v_1 dn - u_2 ds - v_2 dn$$

If the height of the box normal to the panel is allowed to approach zero, then the dashed box will approximate the length of vortex sheet ds , and the equation can be written as

$$\gamma = u_1 - u_2 \quad (3.16)$$

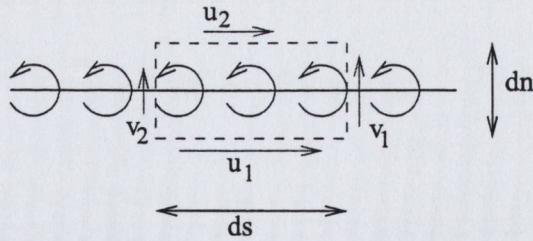


Figure 3.9: Vortex sheet.

The velocity jump across a vortex sheet, then, equals the strength (circulation per unit length) of the vortex sheet. In order to satisfy the no-slip condition, the tangential velocity at the surface must equal zero. This can be achieved by creating a vortex sheet at the surface which will reduce the overall tangential velocity to zero. Consider a fluid flowing past the surface with tangential velocity v_t , as shown in Figure 3.10(a). To reduce this velocity to zero, a vortex sheet must be created which will induce an equal and opposite velocity in the flow, Figure 3.10(b), and maintain a zero velocity at the surface. The overall velocity at the surface, due to the onset tangential velocity and the velocity induced by the vortex sheet, will equal zero and the no-slip condition will be satisfied, Figure 3.10(c). The strength of the created vortex sheet equals the velocity jump across it, which equals minus the onset tangential velocity.

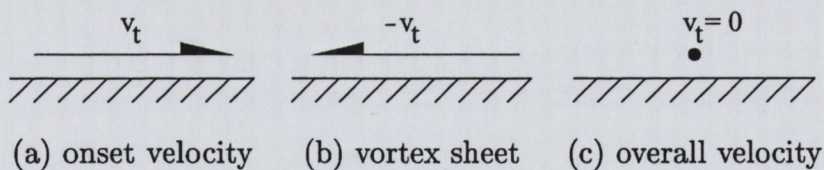


Figure 3.10: Vortex sheet enforcing no-slip condition at surface.

The velocity of the fluid along a cylinder surface changes at different positions on the cylinder surface, and so the strength of the vortex sheet required at each section will also change. These vortex sheets can then be discretised into the point vortices which this simulation of fluid behaviour is based upon.

3.7.3 Point vortices crossing the surface

In Section 3.7.1, zero-permeability was enforced as a boundary condition when calculating the instantaneous velocity field due to the point vortices present in the flow. When the point vortices are then convected and diffused through finite distances in this velocity field, it is possible that some point vortices may cross to the inside of the solid surface. This is not physically realistic, as a fluid cannot flow through a solid surface, but rather occurs due to the fact that we are moving particles by a distance which depends on the velocity at their initial position, and the discrete time step used in the simulation. In other words, it is an artifact of the numerical simulation. It is necessary, therefore, to decide what to do with point vortices which cross the surface in the course of a simulation. Consider, for example, the particles in the velocity field in Figure 3.11.

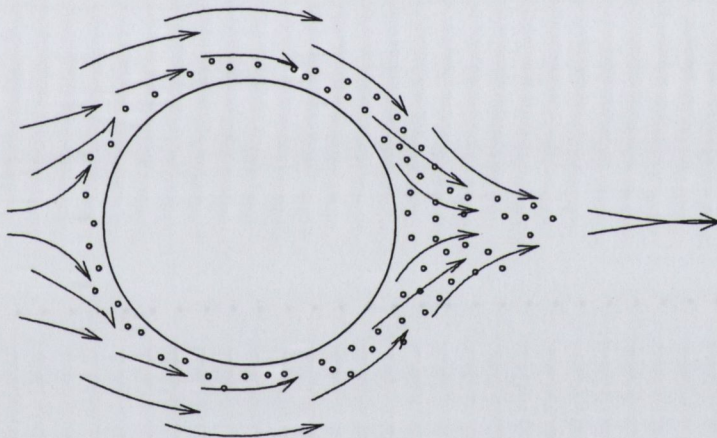


Figure 3.11: Velocity field around cylinder.

It is the process of convection and diffusion of the point vortices which can result in point vortices crossing the surface, as shown in Figure 3.12. The three main methods which have been used to deal with the case of point vortices crossing the surface during a simulation are illustrated in Figure 3.13. The simplest method, used in the original random vortex method paper by Chorin [5] is to delete any point vortices which are found to be inside the boundary. This will remove those

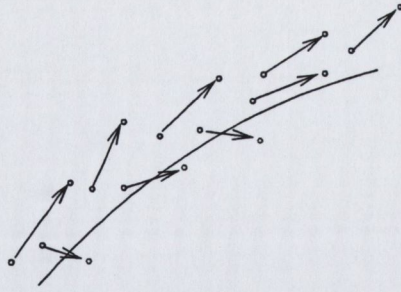


Figure 3.12: Point vortices crossing the surface.

point vortices from subsequent velocity field calculations. When the velocity field is next calculated, the velocity at the surface will have changed not only due to the convection and diffusion of all the point vortices in the flow, but also due to the fact that the circulation of the deleted point vortices is no longer included in the calculation. The calculated tangential velocity at the boundary will result in the effect of the circulation of any deleted internal point vortices being included again in the simulation by the creation of the new point vortices at the surface as required to satisfy the no-slip condition. This method is easy to implement, however a greater number of point vortices need to be created at the surface per time step to obtain the same resolution with this method than would be needed with methods introduced later. The removal of internal point vortices results in less point vortices being available when calculating the velocity field. To counteract this, more point vortices need to be introduced at the surface at each time step than would be necessary if no vortices were deleted.

In a later paper [6], Chorin used a different approach. Instead of deleting any point vortices found inside the boundary, he reflected the point vortices to their mirror-image positions outside the boundary. This ensures that any circulation added to the flow remains in the flow at all times, and the calculation of the velocity field is not threatened by deleted vortices.

A third approach involving absorbing any internal point vortices was proposed by Smith & Stansby in 1989 [24]. In this approach, the internal point vortices

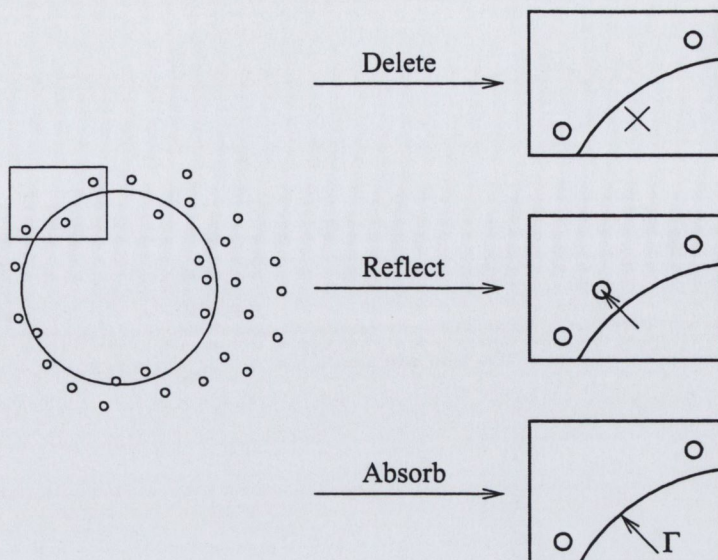


Figure 3.13: Delete, Reflect, Absorb.

were deleted, but their circulation was noted. The circulation of all internal point vortices was then assigned to the surface, and thus included in the next velocity field calculation. The no-slip condition was enforced by the creation of a vortex sheet, and new point vortices were created with circulation strengths defined by the total of the circulation assigned to the surface due to the deleted internal point vortices, and the circulation due to the vortex sheet enforcing no-slip. The new point vortices were diffused with a random walk which was altered slightly from the Gaussian distribution used for all other point vortices. The rest of the method used was identical to the standard random vortex method. In this way, although internal point vortices were deleted, the circulation strengths of the deleted point vortices was recorded and included in the calculation of the velocity field, and simulations could be obtained with less point vortices than would be required with previous methods.

3.8 Summary

In this chapter, the random vortex method has been described. With the random vortex method, it is possible to perform simulations of viscous flow efficiently by tracking particles of defined circulation, which are created at the solid boundary to satisfy the no-slip condition, and which then diffuse and convect with the flow field to satisfy the vorticity transport equation. Zero-permeability is enforced at the solid surface during the velocity field calculation by setting a constant value of stream function at the surface, or by using a panel method. Methods of dealing with point vortices which may cross the solid surface due to convection or diffusion have been discussed. In order for the random vortex method to advance in time, the velocity field must be calculated at each step. The issues which arise when calculating this velocity field are discussed in the next chapter.

Chapter 4

Cloud-in-element

The random vortex method satisfies the vorticity transport equation by convecting point vortices in their local velocity field and adding a random walk to their positions. The solution of the velocity field due to the onset flow, the solid boundary, and the point vortices present is thus required at each time step in the simulation. The velocity field calculation gives rise to two important issues:

- singularities which may cause physically unrealistic results due to the proximity of point vortices in the velocity field calculation must be avoided,
- the method of calculation should allow the simulation to be performed with the least possible amount of operations (CPU efficiency) and with a practical requirement of storage (memory efficiency).

The velocity field calculation and the resolution of these issues are described in this chapter.

4.1 Singularities

The velocity induced by a point vortex is tangential in direction, and increases as the centre is approached, as shown in Figure 4.1. The rate at which the velocity increases with decreasing radius is governed by Eq. (3.8):

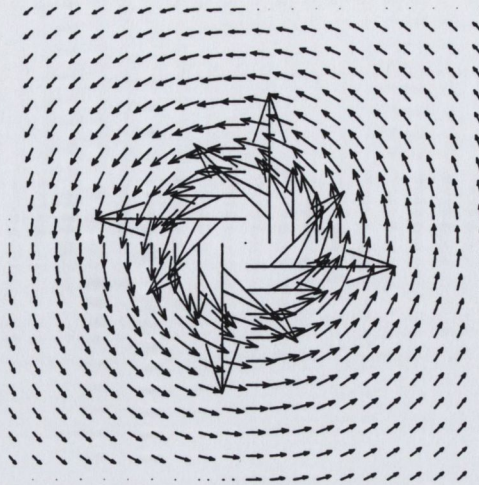


Figure 4.1: Velocity around a point vortex.

$$v_t = \frac{\Gamma}{2\pi r}$$

In a real vortex, the magnitude of the velocity is limited by viscosity. As the centre is approached this viscous effect becomes significant and reduces the velocity towards zero at the centre of the vortex. The area where this viscous effect is significant is called the *vortex core*. The point vortex is a mathematical model of the physical flow around a vortex, which works well for areas outside the core, (Figure 4.2). The model breaks down, however, as the centre is approached. As the distance from the centre approaches zero, the velocity induced by a point vortex approaches infinity, and a singularity exists at the centre of the point vortex. This velocity field is unbounded, and introduces problems when calculating the velocity field due to a number of point vortices. If one point vortex lies close to another point vortex, it will experience a very large induced velocity, and when convected for that time step it will move a large distance, which is not physically realistic.

If a velocity field is to be calculated for use in the simulation it must remain bounded, physically realistic, and so the effects of the singularity must be avoided. One way of doing this is by using *vortex blobs* as the computational elements in the simulation. Vortex blobs make use of a model for the core area of the

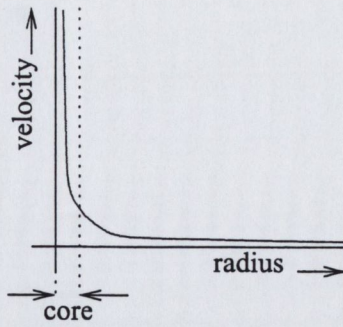


Figure 4.2: Core region of a point vortex.

vortex. A vortex blob uses a cutoff function to ensure that the induced velocity remains physically realistic inside the core area. The choice of a suitable cutoff function is important. The calculation of the velocity field will involve applying this cutoff function for the interaction of all the vortex blobs. Core models based on a Gaussian distribution of vorticity are often favored because of their smoothness and fast decay, but they are expensive to compute, and their values are often tabulated for actual computations. Two examples of Gaussian-based cutoff functions are shown in Figure 4.3 (see [9] for details).

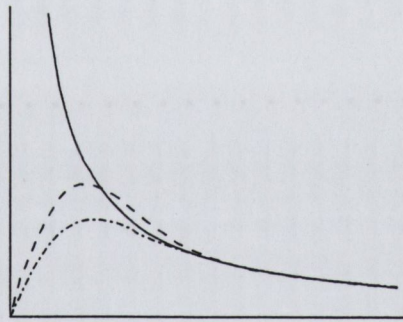


Figure 4.3: Cutoff functions.

Vortex blob techniques are an active area of research in modern vortex methods, and the paper by Koumoutsakos & Leonard [13], which applies the method of vortex blobs for the case of flow around an impulsively started cylinder, presents results of a better resolution than those previously obtained by vortex methods. A comprehensive description of the vortex blob method is given by Cottet &

Koumoutsakos [9]. Although the vortex blob method does deal with the problem of the vortex core, and offers high-resolution results, it introduces the issue of cutoff functions. A simpler way of avoiding nonphysical velocity fields is to use the cloud-in-cell method.

4.2 The cloud-in-cell method

In order to illustrate the cloud-in-cell method, consider the velocity field due to a number of point vortices, such as those shown in Figure 4.4(a). The cloud-in-cell method involves superimposing a mesh over the flow field, Figure 4.4(b). The aim is to assign all of the vorticity from the point vortices onto the nodes of the mesh. Consider the point vortex in Figure 4.4(c). This point vortex will cause a larger vorticity to be experienced at nodes closer to the point vortex than at nodes further away from the point vortex. An area-weighting technique may be used to assign the vorticity of the point vortex onto the nodes of the cell in which it is contained. The point vortex may be considered to split its cell into four areas. When spreading the vorticity onto the nodes of the cell, the area opposite the node is used to decide the fraction of the vorticity of the point vortex which gets assigned to that node. The vorticity experienced at node 1, being the furthest from the point vortex, will be the smallest. It is assigned a vorticity equal to $\omega(\frac{A_1}{A})$. Node 3, on the other hand, is closer to the point vortex, and is assigned a vorticity equal to $\omega(\frac{A_3}{A})$, and so on for the other nodes. This technique is repeated for all the point vortices, and results in the vorticity being *spread* onto the nodes of the mesh, Figure 4.4(d).

The stream function-vorticity (Poisson) equation may now be solved on this mesh, typically by employing a finite difference scheme, and the velocities can be obtained from the gradients of the stream function. The velocity of each point vortex may be found by interpolation. In practice, this is effectively applying the area-weighting technique in reverse, to obtain the velocity at each point vortex

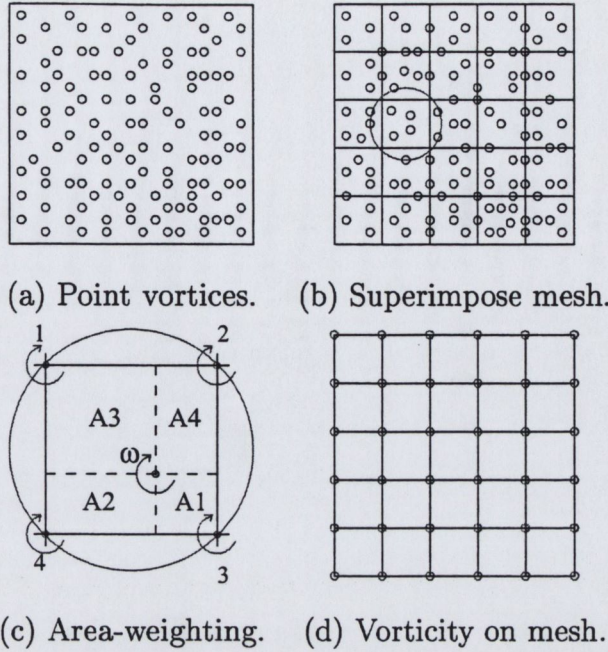


Figure 4.4: Cloud-in-cell method.

from the values of the nodes of its cell.

The cloud-in-cell method inherently overcomes the problem of an unbounded velocity field due to point vortices. As described above, in the cloud-in-cell method, the Poisson equation is solved at the nodes of a mesh which has had the vorticity of all of the point vortices assigned to it. If the nodes of the mesh are spaced sufficiently far apart from each other, then the calculated velocity field will remain bounded and the problem of singularities is automatically overcome.

4.3 Computational efficiency

One problem which quickly becomes apparent with the random vortex method is the amount of computational effort required to calculate the velocity field due to the vortices which exist in the simulation. At each time step, new vortices are created along the surface of the body to enforce the no-slip condition. Hence, as the simulation progresses through more time steps, there will be more vortices

present in the flow field. The effect of each point vortex on every other point vortex needs to be calculated in order to solve for the velocity field. This becomes more difficult as the simulation progresses and the total number of point vortices present increases, and the computer takes longer to calculate each new solution as there are more point vortices involved in the calculation. The cloud-in-cell method, however, offers greatly increased efficiency in this velocity field calculation.

The new velocity field is calculated by solving the stream function-vorticity Poisson equation. To solve this using a direct Biot-Savart summation for the interaction of N point vortices would take an operation count of $O(N^2)$. The cloud-in-cell method reduces the amount of calculations required to do this by solving the Poisson equation on a mesh, as opposed to calculating the interaction of the total number of point vortices in the simulation. As the geometry of the mesh used to calculate the velocity field need not change throughout the course of the simulation, the number of equations to be solved in the discretised velocity calculation remains the same for each time step, no matter how many vortices are in the flow. Once the stream function has been calculated at the nodes of the mesh, these values can be used to interpolate for the velocities at the positions of the point vortices which can then be convected, and so the flow simulation may retain the richness of modelling by moving points, but with better computational efficiency. The number of computations is roughly linear in the number of vortices and mesh points [15].

4.4 The cloud-in-element method

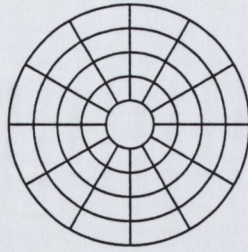
The usual way to approach the cloud-in-cell method is to use a regular mesh, such as a Cartesian or polar grid, and to *spread* the vorticity onto the closest mesh points at each time step. Hence, flow over a circular cylinder would be calculated on a single polar mesh, e.g. Figure 4.5(a). For more than one body, mesh interaction schemes would be required, with multiple meshes, e.g. Figure 4.5(b). The flow

would be calculated on each mesh, and the solution of one mesh must be related to neighbouring meshes. For complex shapes, either some form of conformal mapping is required, as illustrated in Figure 2.12, or else the entire boundary area would need to be covered with a fine mesh and the surface represented by a boundary-integral panel method, requiring a mesh such as that shown in Figure 4.5(c). For any shape more complex than a cylinder or square, the above methods are not easy to apply, and the efficiencies offered by the cloud-in-cell method are hindered by the complexities of the mesh handling.

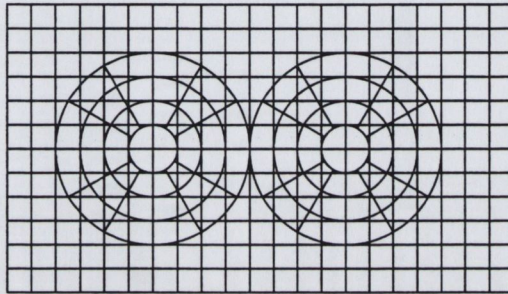
An alternative approach to the cloud-in-cell method is presented in this work, which makes use of an unstructured mesh, avoiding the difficulties of meshing together different Cartesian and polar grids, or conformal mappings. The unstructured mesh allows:

- complex geometries,
- a multiply connected flow domain (i.e. multiple bodies),
- a single mesh for the whole flow (Figure 4.5(d)).

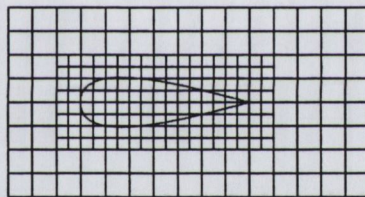
The cloud-in-element method superimposes an unstructured mesh over the flow field, and uses the finite element method to efficiently calculate the velocity field due to the point vortices in the flow. There are two stages involved in preparing the mesh to calculate the velocity field. First, each point vortex needs to be located in the mesh. This involves finding which triangular finite element of the mesh contains each point vortex. Once the point vortices have been located in their *home* elements, the strength of the point vortices must be assigned to the mesh nodes, so that the velocity field can be calculated.



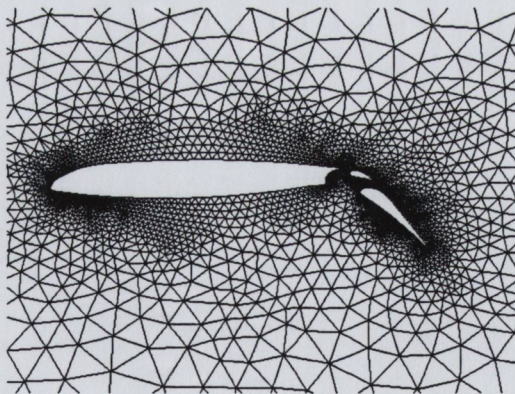
(a) Single radially expanding polar mesh for flow over a cylinder.



(b) Multiple meshes for flow over two cylinders.



(c) Fine regular mesh over non-regular body.



(d) Unstructured mesh for multiple non-regular bodies [1].

Figure 4.5: Different meshes.

4.4.1 Locating points in the mesh

Locating a point in a regular polar or Cartesian mesh is a simple procedure, as the coordinates of the point act as a pointer to the cell in which it belongs. Consider point location in a regular Cartesian grid, as shown in Figure 4.6.

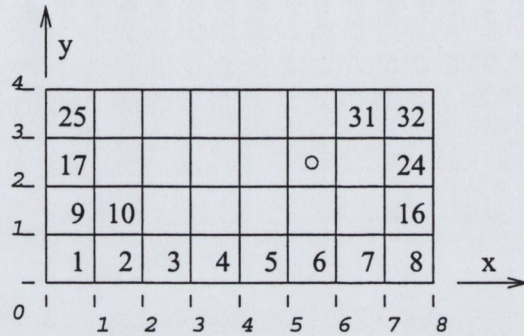


Figure 4.6: Location in a regular grid.

If *CEILING* is defined as a function which rounds a number up to the nearest integer, and *FLOOR* is defined as a function which rounds a number down to the nearest integer, then the home cell, *home*, of a vortex (x, y) could be given by ¹

$$home = CEILING(x) + FLOOR(y) \times 8$$

It is not so easy to locate points in an unstructured mesh, as the elements are not necessarily regular in size, or numbered sequentially. Although it is possible to determine if a point lies inside a triangle, given the coordinates of the point and the triangle vertices (as will be described in Section 4.4.2), to do this for N point vortices and an unstructured mesh with E elements would require a computational effort of $O(N \times E)$.

It is desirable to use the coordinates of the point vortex as a pointer to its home element. This can be achieved by building a matrix which reports the likely home element for the coordinates of a given point vortex. As an analogy, consider how

¹Even if the cells in a Cartesian grid are not the same size, the topology of the mesh is as in Figure 4.6.

an image is displayed on the screen by a computer. For each pixel, a number gives an index into a colormap which defines the colour for that pixel. The colormap information, then, can be considered to give a numerical representation of the image which is seen on the screen. The data for the image shown in Figure 4.7(a) is represented by the matrix in Figure 4.7(b). This matrix shall be referred to herein as the *graphical look-up matrix*.

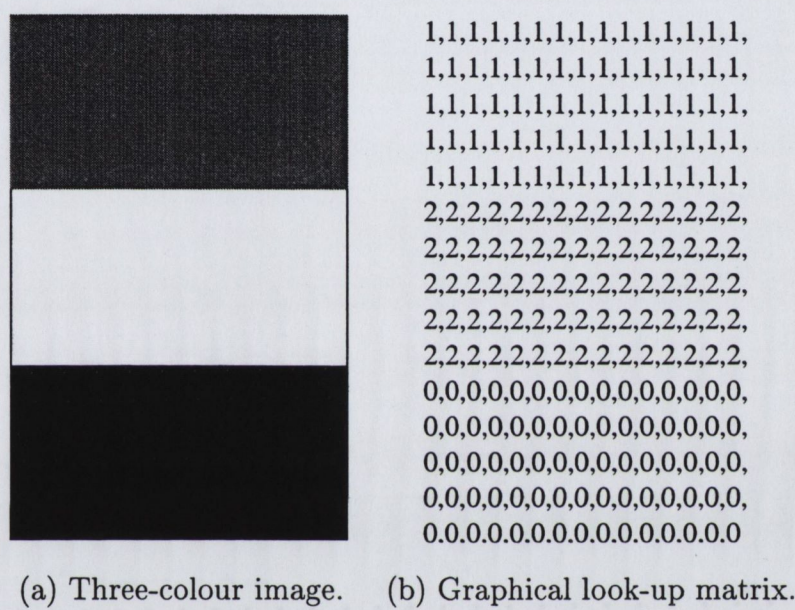
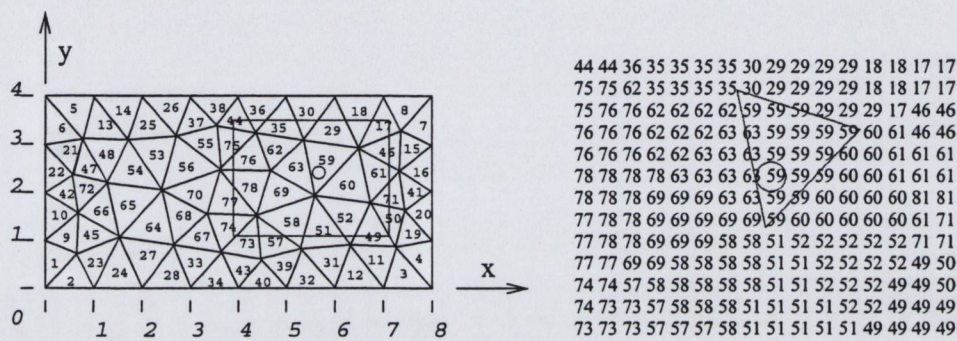


Figure 4.7: A graphical look-up matrix.

To see how this concept may be applied to locating a point in an unstructured mesh, consider Figure 4.8(a), which shows a point vortex located in element number 59 in an unstructured mesh. To enable the computer to efficiently locate this point in the mesh, a graphical look-up matrix for the unstructured mesh is required. This matrix is generated by stepping through the finite element mesh by a fixed amount, and at each step entering the number of that location’s home element as an index into a matrix. When a computer displays an image, the index points to a colormap which defines the colour to be displayed at that location. With the graphical look-up matrix, the index states the element number which contains the point at that location. This allows the coordinates of the point vor-

tex, once suitably truncated, to be used as pointers to the row and column indices of the graphical look-up matrix, hence returning the number of the most likely home element of the point vortex, Figure 4.8(b). As the mesh does not change during the course of the simulation, neither does the graphical look-up matrix, and it can be assembled at the pre-processing stage and stored for use during the simulation.



(a) Finite element mesh. (b) Graphical look-up matrix.

Figure 4.8: Point location.

The graphical look-up matrix offers an efficient way of getting the home element of a point in a finite element mesh, requiring only two FLOPs and a memory operation per vortex, and it can be coded as a vector operation. However, if the point lies near the boundary separating two or more elements, the home element returned from the look-up matrix may be incorrect. The accuracy of the look up matrix in finding the home element of a given point depends on the spatial step size chosen when assembling the matrix.

In Figure 4.9 a step size equal to 1 has been used to generate a graphical look-up matrix for the elements *A* and *B*. When this matrix is used to find the home element of point *P*, which lies near the boundary of *A* and *B*, the coordinates of *P* are rounded off to the closest step size entry, and the element of this entry is returned. The closest step size entry, however, lies in element *B*, and therefore the graphical look-up matrix will return *B* as the (incorrect) home element. A check

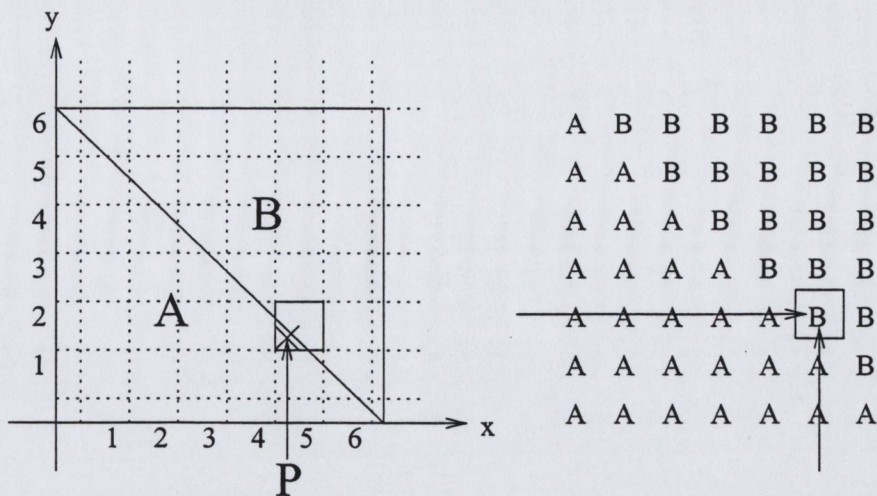


Figure 4.9: Incorrect home element.

needs to be made to ensure that the element number returned by the graphical look-up matrix is correct.

4.4.2 Confirming the home element

To find out if the element number given is indeed the home element of that point, a rigorous mathematical test for determining whether a point lies inside a given triangle is required. The simple case of finding the closest element centroid to the point may not work in cases where adjacent elements have different sizes and shapes, as illustrated in Figure 4.10. Although the point *P* lies inside element *B*, the closest centroid is that of element *A*.

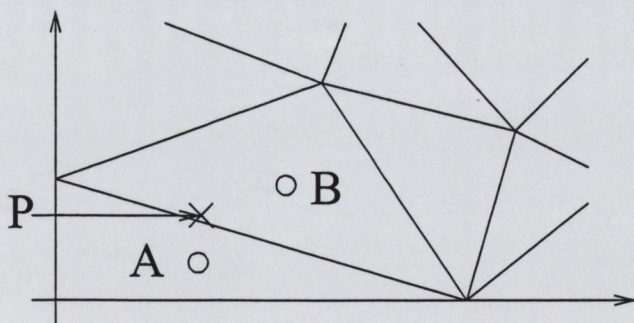


Figure 4.10: Closest centroid.

A more rigorous test is available by making use of the vector cross product. If $\vec{a}$ and $\vec{b}$ are collinear or one of these vectors is the zero vector, then $\vec{c} = \vec{a} \times \vec{b} = 0$. In any other case, $\vec{c} = \vec{a} \times \vec{b}$ is the vector whose length is equal to the area of the parallelogram with $\vec{a}$ and $\vec{b}$ as adjacent sides and whose direction is perpendicular to both $\vec{a}$ and $\vec{b}$ and is such that $\vec{a}$, $\vec{b}$, $\vec{c}$, in this order, form a right-handed triplet as shown in Figure 4.11.

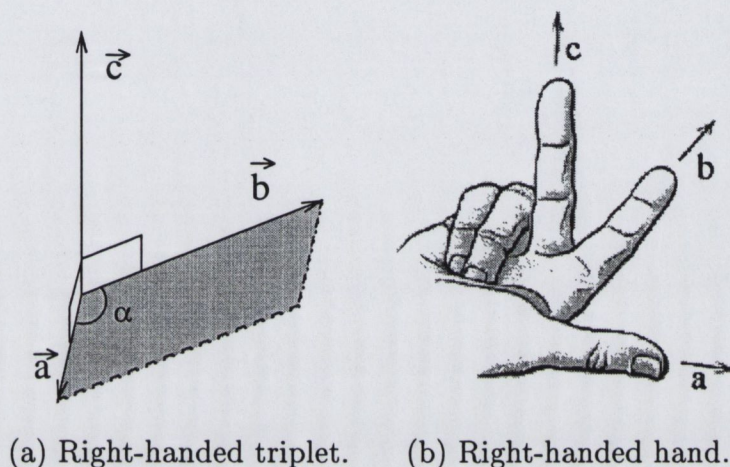


Figure 4.11: Vector (cross) product.

The parallelogram with $\vec{a}$ and $\vec{b}$ as adjacent sides has the area $|\vec{a}||\vec{b}|\sin\alpha$, where α is the angle between $\vec{a}$ and $\vec{b}$. We thus obtain

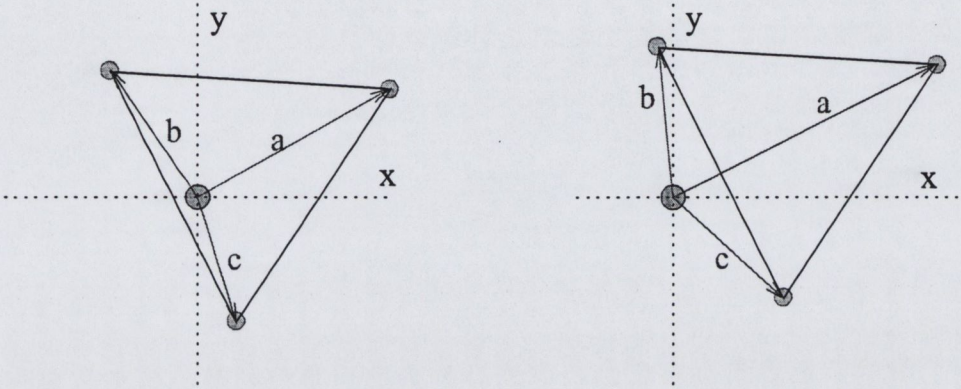
$$|\vec{c}| = |\vec{a}||\vec{b}|\sin\alpha. \quad (4.1)$$

Let $\vec{a} \times \vec{b} = \vec{c}$ and let $\vec{b} \times \vec{a} = \vec{d}$. Then, by definition, $|\vec{c}| = |\vec{d}|$, and in order that $\vec{b}$, $\vec{a}$, $\vec{d}$ form a right-handed triplet we must have $\vec{d} = -\vec{c}$. This implies that

$$\vec{b} \times \vec{a} = -\vec{a} \times \vec{b} \quad (4.2)$$

that is, cross multiplication of vectors is not commutative but anticommutative. Hence the order of the factors in a vector product is of great importance and must be carefully observed.

The vector product can be used to test whether a point lies inside a triangle by the following method. For each vortex and corresponding test element, the vortex coordinates are subtracted from the vertices of the element, which results in the element being positioned with the vortex at the origin. Figure 4.12 shows this operation for the cases when the vortex is inside and outside the element.



(a) Point inside the element. (b) Point outside the element.

Figure 4.12: Testing the home element.

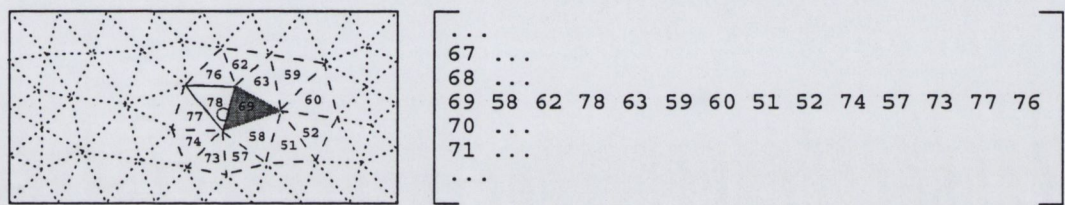
The vertices of the triangle can now be considered as vectors from the point vortex to the vertices. If the vortex is inside the triangle, then $\vec{a} \times \vec{b}$, $\vec{b} \times \vec{c}$ and $\vec{c} \times \vec{a}$ will all be positive, or alternatively $\vec{a} \times \vec{c}$, $\vec{c} \times \vec{b}$ and $\vec{b} \times \vec{a}$ will all be negative. They will all have the same sign, because the angles between them is less than π , as shown in Figure 4.12(a). If the vortex is outside the triangle, on the other hand, one of the cross products will have the opposite sign to the others, as one of the angles is greater than π (Figure 4.12(b)).

The order in which the cross products of the vectors are performed is important, as opposite directions produce opposite signs, and the sign of the cross product is the key to whether or not the point lies inside the triangle. However, if the connectivity matrix is written in the same order for all elements, say counterclockwise, then the cross products of the vectors can be taken in the order in which they are listed in the connectivity matrix, and the test will be valid for all elements in the mesh. This test is mathematically sufficient to verify that a point lies within

a triangular element. If it is found that the point vortex does *not* lie within the element given by the graphical look-up matrix, the *neighbours* matrix is used to find the correct home element.

4.4.3 The neighbours matrix

Figure 4.13 shows a point vortex near the edge of an element. The graphical look-up matrix returns the home element of this point as being the adjacent shaded element (#69), instead of the true home element (#78).



(a) Neighbouring elements.

(b) Neighbours matrix.

Figure 4.13: Finding the correct home element.

This is where the matrix of neighbouring elements can be used. Each row of the neighbours matrix lists all of the elements which share nodes with the element associated with that row. In Figure 4.13(a) the graphical look-up matrix has returned the incorrect element number 69 as the home element. However, by testing the elements listed in row 69 of the neighbours matrix, element number 78 will be found as the true home element.

The graphical look-up matrix offers an efficient way of locating a point in the mesh, and will usually return the correct element number. However, in cases when the point lies close to the boundary of its home element, the correct element can still be found within a few iterations with the help of the neighbours matrix. On average, the number of tests per vortex during a simulation is of $O(1.5)$.

4.4.4 Spreading the circulation

At this stage of the process every point vortex has had its home element verified. In order to solve for the velocity field caused by these point vortices, their circulation needs to be spread from the point vortices onto the nodes of their home elements. Then the Poisson equation can be solved on the unstructured mesh using the finite element method to obtain the stream function and hence the velocity field.

The circulation is spread from the point vortex P to the nodes of the triangle in which it is contained by using an area-weighting scheme similar to that used in the cloud-in-cell method described in Section 4.2. Hence, if the circulation of the vortex at P is Γ , then the circulation spread onto node 1 in Figure 4.14(a) will equal $\Gamma \frac{A_1}{A}$, that spread onto node 2 will equal $\Gamma \frac{A_2}{A}$, and for node 3 will equal $\Gamma \frac{A_3}{A}$, where A is the area of the triangle.

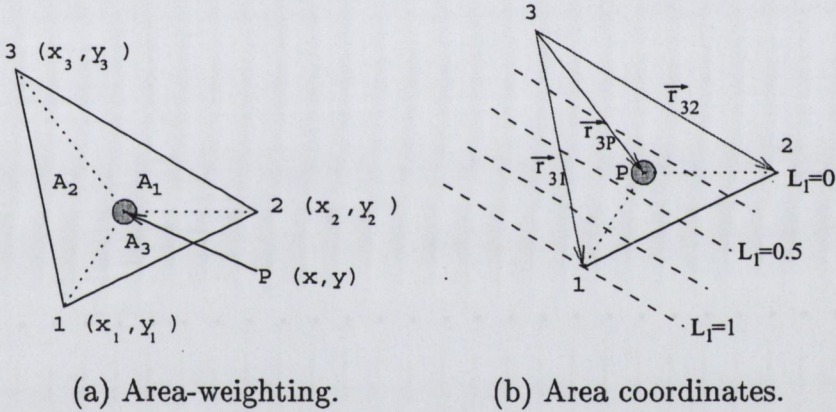


Figure 4.14: Area-weighting in a triangle.

Area-weighting is efficient to implement with a regular Cartesian or polar coordinate system as the ratio of the areas involved can be directly related to the coordinate system being used. When spreading onto the nodes of a triangular element in an unstructured mesh, these regular coordinate systems do not directly give information about the areas needed for area-weighting. A coordinate system based on the triangle itself is desirable. This can be achieved by using area coordinates (see, for example, Zienkiewicz & Taylor [30]). Area coordinates for a

triangular element are defined by the linear relation given in Eq. (4.3).

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} \quad (4.3)$$

This will produce the coordinate system (L_1, L_2, L_3) for the triangle in question. Each coordinate is equivalent to the fraction of the perpendicular distance from the opposite side to the node in question, as shown for L_1 in Figure 4.14(b). Gaussian elimination yields an expression for L_1 .

$$L_1 = \frac{(x - x_3)(y_2 - y_3) - (x_2 - x_3)(y - y_3)}{(x_1 - x_3)(y_2 - y_3) - (x_2 - x_3)(y_1 - y_3)} \quad (4.4)$$

To better understand how Eq. (4.4) can be used to efficiently apply the area-weighting technique, consider again Figure 4.14(b). The vectors from point 3 to points 1, 2, and P are given by

$$\begin{aligned} \vec{r}_{31} &= (x_1 - x_3)\vec{i} + (y_1 - y_3)\vec{j} \\ \vec{r}_{32} &= (x_2 - x_3)\vec{i} + (y_2 - y_3)\vec{j} \\ \vec{r}_{3P} &= (x - x_3)\vec{i} + (y - y_3)\vec{j} \end{aligned}$$

The magnitude of the cross product of $\vec{r}_{31}$ and $\vec{r}_{32}$ will give the area of the parallelogram whose two corner sides are described by those vectors, which will equal twice the area of the triangle 123, A_{123} .

$$|\vec{r}_{31} \times \vec{r}_{32}| = 2A_{123} = (x_1 - x_3)(y_2 - y_3) - (x_2 - x_3)(y_1 - y_3) \quad (4.5)$$

Similarly, the magnitude of the cross product of $\vec{r}_{3P}$ and $\vec{r}_{32}$ will equal twice the area of the triangle $P23$, A_{P23} .

$$|\vec{r}_{3P} \times \vec{r}_{32}| = 2A_{P23} = (x - x_3)(y_2 - y_3) - (x_2 - x_3)(y - y_3) \quad (4.6)$$

Comparing Eq. (4.5) and Eq. (4.6) to Eq. (4.4) the benefit of using area coordinates for area-weighting becomes apparent.

$$L_1 = \frac{(x - x_3)(y_2 - y_3) - (x_2 - x_3)(y - y_3)}{(x_1 - x_3)(y_2 - y_3) - (x_2 - x_3)(y_1 - y_3)} = \frac{A_{P23}}{A_{123}} \quad (4.7)$$

The circulation spread onto node 1 can now be expressed as $L_1\Gamma$, the circulation on node 2 is $L_2\Gamma$, and for node 3 is $L_3\Gamma$. The set of area coordinates may be rewritten as

$$\begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} = \frac{1}{2A_{123}} \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \begin{pmatrix} 1 \\ x \\ y \end{pmatrix}$$

where

$$\begin{aligned} a_1 &= x_2y_3 - x_3y_2 \\ b_1 &= y_2 - y_3 \\ c_1 &= x_3 - x_2 \end{aligned} \quad (4.8)$$

with cyclic rotation of indices 1, 2, 3. As the geometry of the unstructured mesh does not change during the course of the simulation, the values of a, b, c and A can be calculated for each element and stored before the simulation begins, and area-weighting can then be used to efficiently spread the circulation from a point vortex onto the nodes of the triangular element in which it is contained.

4.5 The finite element method

The finite element method is discussed in detail by, for example, Zienkiewicz and Taylor [30]. The aspects relevant to the problem of solving for the stream function-vorticity equation are discussed here. The governing differential equation to be solved on the finite element mesh is Eq. (3.11).

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} + \omega = 0$$

As the governing equation must equal zero at each point of the domain D , it follows that an integral form of this equation may be written as

$$\int_D h \left[\frac{\partial}{\partial x} \left(\frac{\partial \psi}{\partial x} \right) + \frac{\partial}{\partial y} \left(\frac{\partial \psi}{\partial y} \right) + \omega \right] dxdy = 0 \quad (4.9)$$

where h is some arbitrary function. If Eq. (4.9) is satisfied for all choices of h then the governing differential equation is satisfied at all points of the domain. The weak form of this equation is obtained by integrating by parts, using the general formulæ

$$\begin{aligned} \int_D h \frac{\partial}{\partial x} \left(\frac{\partial \psi}{\partial x} \right) dxdy &\equiv - \int_D \frac{\partial h}{\partial x} \left(\frac{\partial \psi}{\partial x} \right) dxdy + \oint_S h \left(\frac{\partial \psi}{\partial x} \right) n_x dS \\ \int_D h \frac{\partial}{\partial y} \left(\frac{\partial \psi}{\partial y} \right) dxdy &\equiv - \int_D \frac{\partial h}{\partial y} \left(\frac{\partial \psi}{\partial y} \right) dxdy + \oint_S h \left(\frac{\partial \psi}{\partial y} \right) n_y dS \end{aligned}$$

where S is the boundary, and n_x, n_y are the direction cosines between the outward normal and the x, y directions. This gives

$$\left[- \int_D \frac{\partial h}{\partial x} \frac{\partial \psi}{\partial x} dxdy - \int_D \frac{\partial h}{\partial y} \frac{\partial \psi}{\partial y} dxdy \right] + \left[\int_S h \left(\frac{\partial \psi}{\partial x} n_x + \frac{\partial \psi}{\partial y} n_y \right) dS \right] + \int_D h \omega dxdy = 0$$

Noting that the derivative along a normal is given by

$$\frac{\partial \psi}{\partial n} = \frac{\partial \psi}{\partial x} n_x + \frac{\partial \psi}{\partial y} n_y$$

and considering that the natural boundary condition will result in zero normal gradient at the boundary, the second term equals zero and the weak form is

$$\int_D \left[\frac{\partial h}{\partial x} \frac{\partial \psi}{\partial x} + \frac{\partial h}{\partial y} \frac{\partial \psi}{\partial y} \right] dxdy - \int_D h \omega dxdy = 0$$

The finite element method seeks a solution for ψ at the nodes of the mesh. The value of ψ at any location in the mesh can then be obtained by interpolating from

the nodal values of the elements. The functions used to interpolate a value at a location inside an element from the values at the nodes of that element are called the shape functions, N_i .

$$\psi = \sum N_i \psi_i$$

For the work presented here, first order triangular elements have been used, with nodes at the vertices of the elements. The shape functions give the nodal value when evaluated at a node, and a weighted ratio of the nodal values when evaluated at a location inside the element. These criteria are met by the area coordinates defined in Section 4.4.4, and the area coordinates and the shape functions are identical. As the Poisson equation is to be satisfied for each element of our mesh, we define the arbitrary function h for each element. The Galerkin method uses the shape functions N_i as the arbitrary function h in the weak form of the governing equation. The problem can now be expressed in the form

$$[K]\{\psi\} + \{f\} = 0$$

where $[K]$ is the global stiffness matrix, $\{\psi\}$ is the vector of nodal values of the stream function, and $\{f\}$ is the loading vector of nodal values of circulation, after the circulation has been spread from the point vortices onto the nodes of the mesh. For each element

$$N_i = \frac{a_i + b_i x + c_i y}{2A} \quad (4.10)$$

$$K_{ij}^e = \int_D \left[\frac{dN_i}{dx} \frac{dN_j}{dx} + \frac{dN_i}{dy} \frac{dN_j}{dy} \right] dx dy \quad (4.11)$$

$$= \frac{1}{4A} [b_i b_j + c_i c_j] \quad (4.12)$$

$$f_i^e = - \int_D N_i \omega_i dx dy \quad (4.13)$$

$$= - \frac{1}{2} a_i \omega_i \quad (4.14)$$

where the loading vector was calculated using the integration formulæ for a triangle with the origin at the centroid. In the cloud-in-cell method the *vorticity* is spread from the point vortex onto the nodes of the cell in which it is contained. In the cloud-in-element method the *circulation* is spread from the point vortex onto the nodes of the triangular element. To understand why this is so, consider the triangular element shown in Figure 4.15.

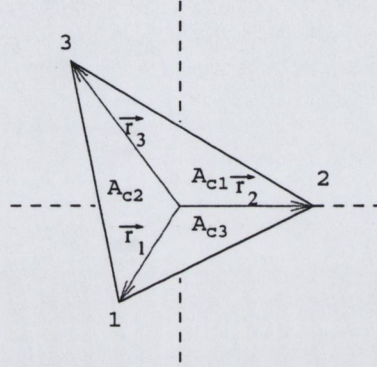


Figure 4.15: Triangle centroid at origin.

The loading vector for this element is given by

$$f_e = -\frac{1}{2} \begin{pmatrix} a_1 \omega_1 \\ a_2 \omega_2 \\ a_3 \omega_3 \end{pmatrix} \quad (4.15)$$

and the nodal vectors for this element are

$$\vec{r}_1 = x_1 \vec{i} + y_1 \vec{j}$$

$$\vec{r}_2 = x_2 \vec{i} + y_2 \vec{j}$$

$$\vec{r}_3 = x_3 \vec{i} + y_3 \vec{j}$$

It is known from Eq. (4.8) that

$$a_1 = x_2 y_3 - x_3 y_2$$

however,

$$|\vec{v}_2 \times \vec{v}_3| = x_2 y_3 - x_3 y_2$$

and so

$$a_1 = 2A_{c1}$$

and similarly for the other nodes. As the origin is at the centroid,

$$A_{c1} = A_{c2} = A_{c3} = \frac{1}{3}A$$

and the loading vector can now be written as

$$f_e = -\frac{1}{2} \begin{pmatrix} \frac{2}{3}A\omega_1 \\ \frac{2}{3}A\omega_2 \\ \frac{2}{3}A\omega_3 \end{pmatrix} \quad (4.16)$$

Area-weighting of circulation and vorticity onto node1 gives

$$\begin{aligned} \omega_1 &= \frac{A_1}{A}\omega \\ \Gamma_1 &= \frac{A_1}{A}\Gamma \end{aligned}$$

using the relationship between circulation and vorticity given by Eq. (3.7)

$$\omega = \frac{\Gamma}{A}$$

allows the spread circulation and vorticity to be related by

$$\omega_1 = \frac{\Gamma_1}{A}$$

resulting in the loading vector being expressed as

$$f_e = -\frac{1}{3} \begin{pmatrix} \Gamma_1 \\ \Gamma_2 \\ \Gamma_3 \end{pmatrix} \quad (4.17)$$

and so the loading vector consists of the circulation which has been spread onto the nodes as opposed to the vorticity.

4.6 Calculating the velocity field

Once the circulation has been spread from the distribution of point vortices present onto the nodes of the finite element mesh, the velocity field may be calculated. The system of linear equations at this stage may be considered to be

$$[K]\{\psi\} + \{f\} = 0$$

The loading vector $\{f\}$ contains the spread circulation, and the matrix $[K]$ can be assembled and stored before the simulation begins, as the mesh geometry does not change during the simulation. Solving this system yields the vector of the nodal values of stream function, $\{\psi\}$. The velocity field may be obtained from the values of stream function by using the definition of stream function, Eq. (3.10).

$$\begin{aligned} u &= \frac{\partial \psi}{\partial y} \quad , \quad v = -\frac{\partial \psi}{\partial x} \\ \psi &= \sum N_i \psi_i \\ N_i &= \frac{a_i + b_i x + c_i y}{2A} \end{aligned}$$

hence,

$$u = \frac{1}{2A} \sum c_i \psi_i \quad , \quad v = -\frac{1}{2A} \sum b_i \psi_i \quad (4.18)$$

The expressions for the velocities do not depend on the position (x, y) inside the element, therefore it can be seen that for triangular elements with three nodes, the velocity is constant within an element. All point vortices which lie in the same element will be convected by the same velocity. This means that the resolution of the calculated velocity field in a certain area will depend on the size of the elements used to mesh those areas. Areas which are not of interest, such as the flow upstream of the cylinder, or distant from the point vortices, where the velocity field is not expected to fluctuate much, may be meshed with larger elements. Areas where higher resolution is required, such as near the solid boundary and in the near wake, may be meshed with smaller elements to better capture the velocity field. In this way, computational effort may be concentrated in the areas of interest. A better resolution of the velocity field may also be obtained by using higher order finite elements, such as triangular elements with six nodes. This would allow a velocity field to be calculated which varies linearly across the face of each element. However, the global stiffness matrix $[K]$ increases in size as the number of nodes in the mesh increase, increasing the computational effort required to solve the Poisson equation. Although preliminary studies were carried out of simulations using triangular elements with six nodes, the pros and cons of higher order elements have not been investigated in the present work, which simply uses triangular elements with three nodes for all simulations, increasing mesh density where higher resolution is required.

4.7 Summary

The cloud-in-element method allows an efficient calculation of the velocity field due to a distribution of point vortices in an unstructured mesh. The home element of each point vortex can be estimated by using the graphical look-up matrix. The point is confirmed as lying inside this triangular element by using vector cross products, and if the point does not lie inside the triangle, then the neighbours

matrix can be used to find the true home element. Once all point vortices have had their home elements verified, the circulation is spread from the point vortices onto the nodes of the finite element mesh using an area-weighting scheme. This is efficiently achieved by making use of area coordinates, which are identical to the shapefunctions for triangular elements with three nodes. The mesh is then solved using the global stiffness matrix to find the stream function, and from these values of stream function the velocity is obtained for each element of the mesh.

Chapter 5

Preparing a simulation

5.1 Generating the mesh

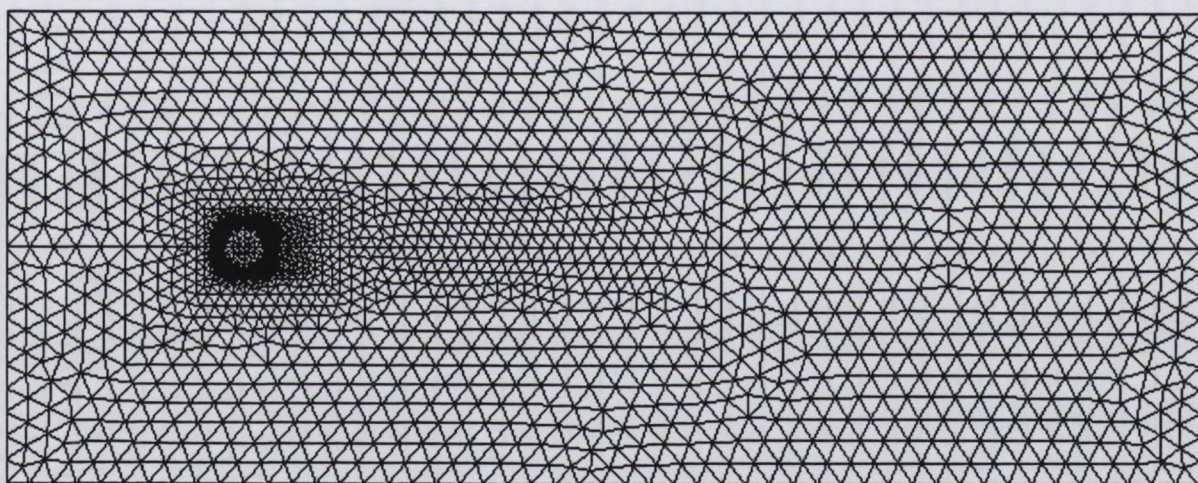


Figure 5.1: A simple complete mesh.

The first step in preparing a simulation is to generate the mesh. GAMBIT, the meshing program which is part of the FLUENT package, was used but any meshing program can be used which will output the nodal coordinates and the connectivities of the mesh generated. The areas of most interest will be the flow near the solid boundary where zero-permeability is enforced and the vorticity created to enforce the no-slip condition, and the flow in the region near the body after separation has

occurred, where recirculation and large scale shedding of vortical structures will be initiated. These areas are more densely meshed to provide more detail of the local velocity field. The area upstream of the body should experience no major flow fluctuations, and the area in the far wake should settle into a regular flow pattern.

5.1.1 Meshing near the surface

The surface of the circular cylinder is represented by a circle centred at the origin. In this study, a circular cylinder of radius $a_{cyl} = 1.0$ is studied in an impulsively started uniform velocity of $U_\infty = 1.0$. This allows distance and time to be regarded as non-dimensional. Dimensional time t^* , and velocity $\vec{u}^*$ can be obtained by

$$\begin{aligned} t^* &= \frac{a_{cyl} t}{U_\infty} \\ \vec{u}^* &= U_\infty \vec{u} \end{aligned}$$

The zero-permeability condition is enforced either by setting the value of the stream function to equal a constant, or by applying a boundary integral method, such as source panels. In both cases the required result is a velocity field which flows around (and not through) the body, and so exhibits a purely tangential velocity at the surface. The cloud-in-element method calculates the velocity field after the circulation has been spread from the point vortices onto the nodes of the home elements. This circulation spreading means that the resolution of the calculated velocity field is of the same scale as the element size. Some effects of spreading the circulation onto the mesh nodes must be taken into account. If the zero-permeability boundary condition is to be enforced by defining the value of the stream function on the boundary, then care must be taken to ensure circulation is not spread onto a boundary node. Figure 5.2(a) shows a point vortex near the boundary of a (shaded) solid in the flow. If the circulation of this point vortex is spread onto the nodes of the element in which it is contained, some

circulation will be spread onto the nodes where the zero-permeability condition is being enforced (called the *control* nodes, circled). The effect of this will be that the boundary condition will no longer be enforced as required, as shown in Figure 5.2(b). Therefore, when using the stream function boundary condition, the boundary condition is enforced at nodes slightly inside the surface.

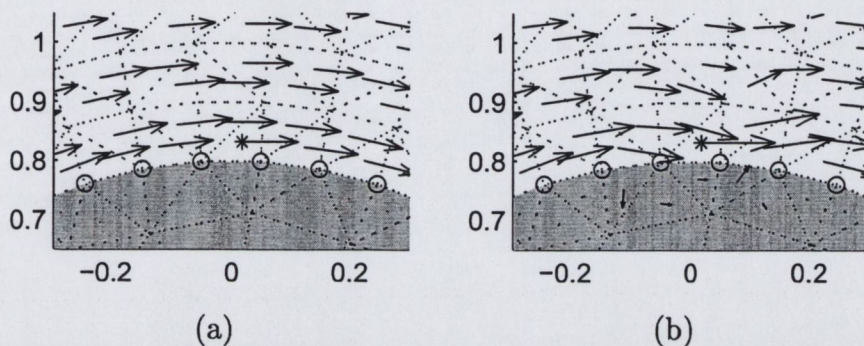


Figure 5.2: Spreading circulation onto boundary nodes.

The strengths of the point vortices which are created at each time step of the simulation to enforce the no-slip boundary condition depends on the tangential velocity calculated at the surface. Spreading circulation onto the mesh nodes means that the tangential velocity used to define the strengths of the newly created point vortices is not actually calculated at the surface, but slightly inside the surface. Figure 5.3(a) shows a point vortex located near the surface. The effect of this point vortex would be to increase the tangential velocity at the surface. Figure 5.3(b) shows the new velocity calculated once the circulation of this point vortex has been spread onto the mesh.

If the tangential velocity at the surface was taken from the velocity of the element which meshes from the surface into the fluid, the velocity would not have changed much due to the presence of the point vortex. If the tangential velocity is taken from an element just inside the surface, the velocity will have increased, as expected. Therefore, in order to ensure that the surface velocity calculated due to the point vortices present is accurate, it must be taken from elements slightly inside the surface.

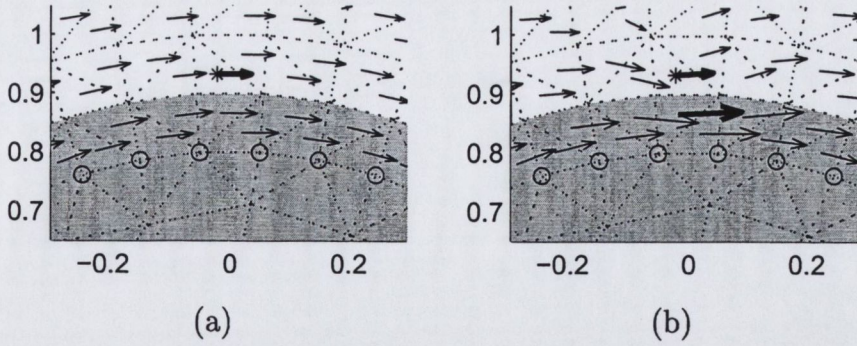


Figure 5.3: Tangential velocity used when creating new point vortices.

At each time step in the simulation, point vortices will be created at the surface with circulation strengths defined by the tangential velocities at the control nodes. These point vortices will then be allowed to diffuse away from the surface by applying a random walk with a standard deviation of $\sqrt{2\nu\Delta t}$ to their positions. In order to calculate the velocity field with a good resolution, the element size must be of the same order as the motion of the point vortices. Hence, when meshing near the surface, an element spacing of the order of the standard deviation of the random walk is used. This gives a higher resolution of the velocity field near the surface, where the creation and diffusion of point vortices is of primary importance.

It is easy to decide if a point vortex lies inside a simple geometry. In the case of a circle, for example, the distance of the point vortex from the centre of the circle could be calculated and compared to the radius of the circle. For more complex geometries, or for multiple bodies, such simple tests cannot be applied. For this reason, when the mesh is being generated the entire flow field is meshed, including the interior of any solid boundaries. A reference matrix is then constructed which designates all elements as being either inside or outside the solid body. Once the home element of a point vortex has been found, this reference matrix can be used to decide if the vortex lies in the flow, or inside the solid body, and the simulation proceeds accordingly.

The number of nodes used to represent the surface of the cylinder is related to the mesh spacing normal to the surface (hereafter referred to as the surface mesh

spacing). If the distance between surface nodes is large relative to the surface mesh spacing, the elements will have a bad aspect ratio, as in Figure 5.4(a). The same is true if the distance between surface nodes is small relative to the surface mesh spacing, Figure 5.4(c). Good aspect ratios are obtained when the space between surface nodes is approximately equal to the surface mesh spacing, Figure 5.4(b).

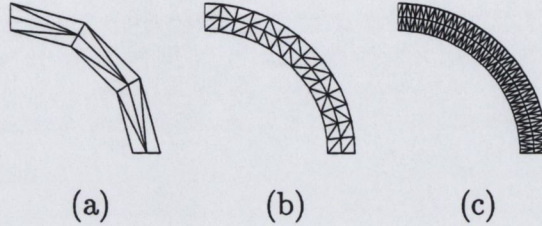


Figure 5.4: Surface node dependency on surface mesh spacing.

For a given Reynolds number, the standard deviation of the random walk depends on the size of the time step, Δt . As the standard deviation changes with Reynolds number and time step, so too does the surface mesh spacing, and the number of surface nodes.

$$\sqrt{2\nu\Delta t} = \sqrt{2 \left(\frac{2a_{cyl}U_\infty}{Re} \right) \Delta t} = \sqrt{\frac{4\Delta t}{Re}}$$

The effect of changing the time step and the surface mesh spacing was investigated by running simulations at $Re = 550$ and comparing the computed coefficient of drag, C_D , with that obtained by Koumoutsakos & Leonard [13]. Figure 5.5 shows how C_D varies with time step, when the ratio of surface mesh spacing to the standard deviation of the random walk equals 1. The ratio of surface mesh spacing to the standard deviation of the random walk will hereafter be represented by MR , hence $MR = 1$ for Figure 5.5.

As the time step increases, the magnitude of the computed drag is also seen to increase. The trend remains the same for the different time steps, except the initial drop off in drag is not as sharp with larger time steps as it is with the smaller values. As the time step increases for the different simulations, so too does

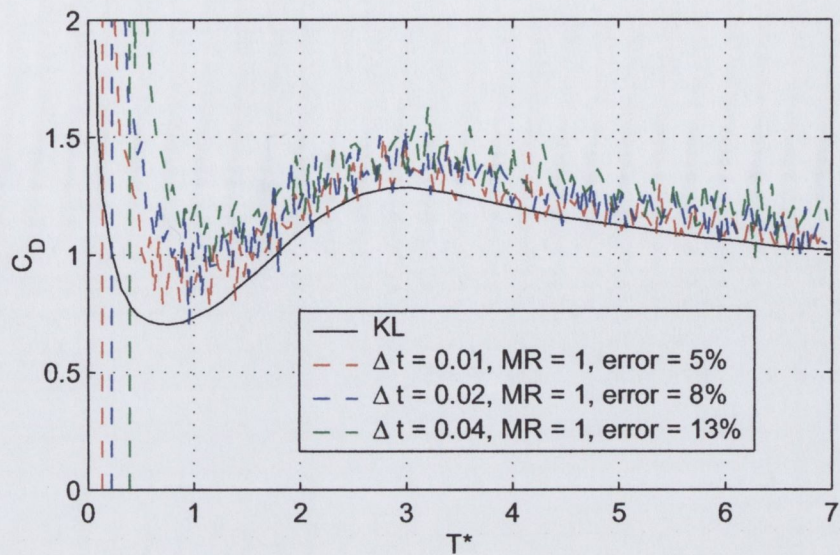


Figure 5.5: Computed drag for $Re = 550$, as Δt is altered.

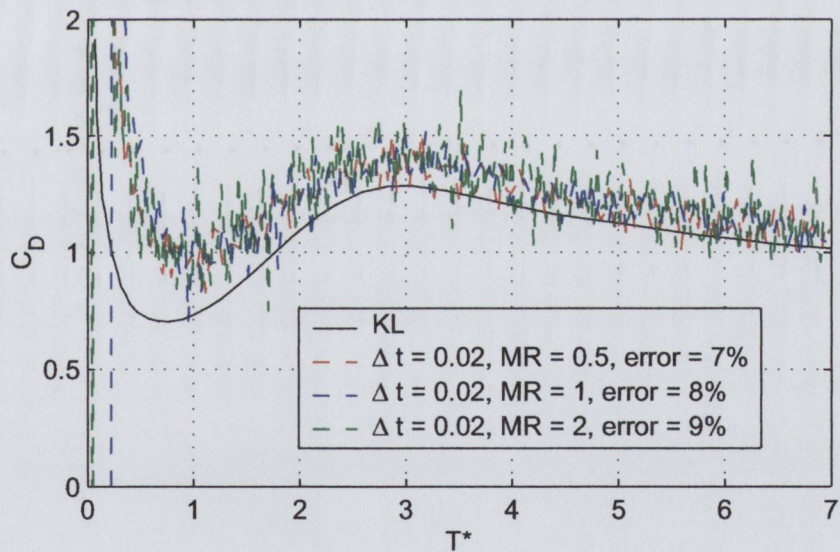


Figure 5.6: Computed drag for $Re = 550$, as MR is altered.

the standard deviation, and hence the surface mesh spacing also increases.

The effect of the time step and the mesh spacing size on the simulation results are investigated independently. Figure 5.6 shows the results obtained by varying the surface mesh spacing ratio while holding the time step constant. The effect of varying the time step while maintaining the same mesh spacing is shown in Figure 5.7

Figure 5.8 shows the CPU time required by the simulations in Figure 5.5, Figure 5.6 and Figure 5.7. The number of computations required for a simulation increases as the time step and mesh size decrease, and the overall number of point vortices in the simulation increase. Therefore, parameters have been chosen for the simulations presented in Chapter 7 which compromise between acceptable accuracy and fast simulation time.

Figure 5.8 also shows an important result about the efficiency of the cloud-in-element method. The slope of the line formed by plotting iteration time against the number of vortices present remains constant for the different simulations. Changes in mesh density cause a change in the initial offset of the plotted line. It can thus be seen that the iteration time varies linearly with the number of vortices present in the simulation, and this linear relationship is independent of the mesh density.

5.1.2 Meshing in the near wake

In areas further away from the surface, the point vortices will be convected as well as diffused, and the scale of motion becomes larger. Also, the effect which each point vortex in the flow field has at the surface of the body decreases with distance from the body. Hence the element size may be allowed to increase when meshing areas further away from the surface.

The region close to and behind the body is still of interest, however, and a good resolution is required. It is in this region that separation occurs, and recirculating regions will grow, interact, and move downstream. A good mesh definition in this

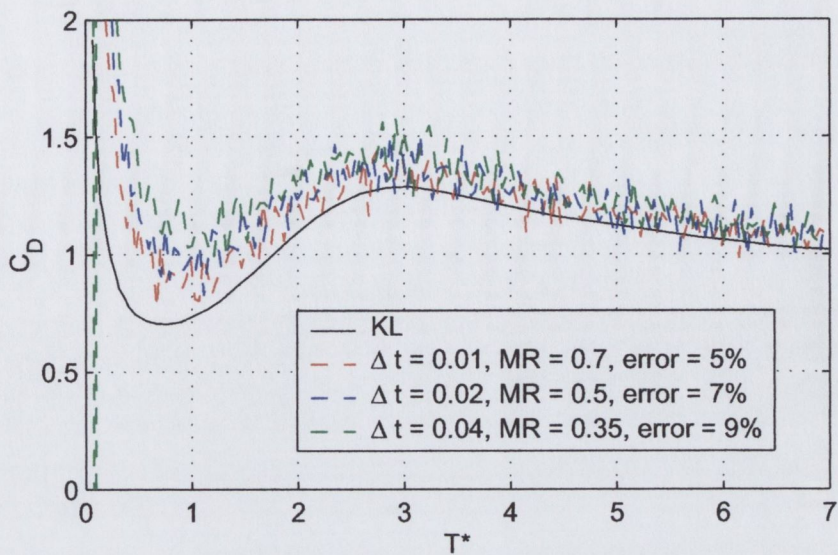


Figure 5.7: Computed drag for $Re = 550$, as Δt is altered.

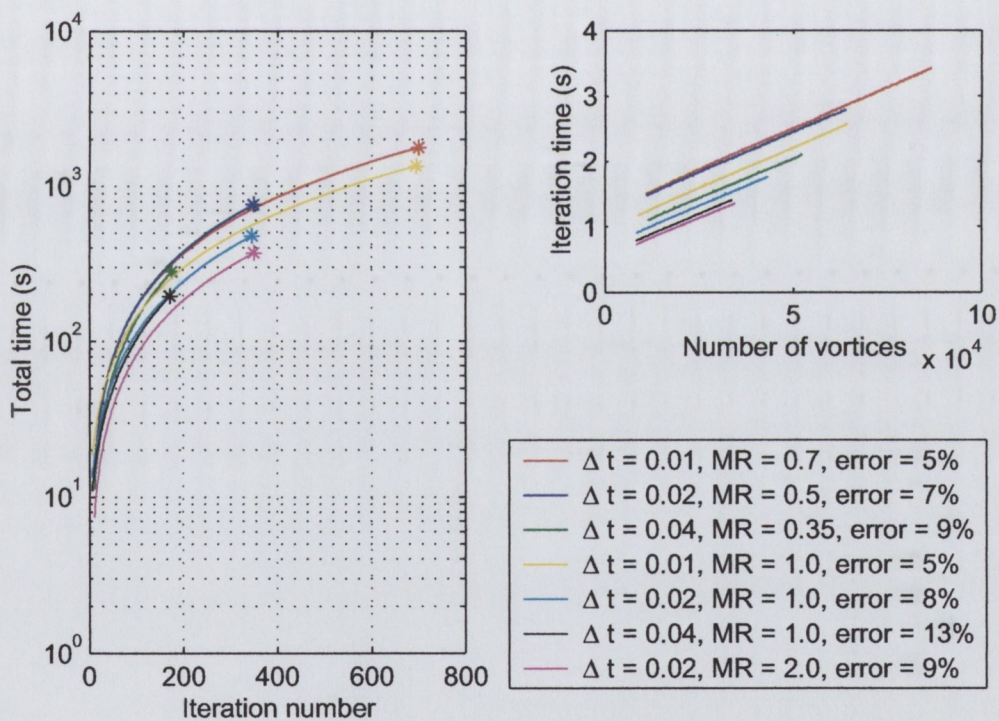


Figure 5.8: CPU times for $Re = 550$, with different parameter settings.

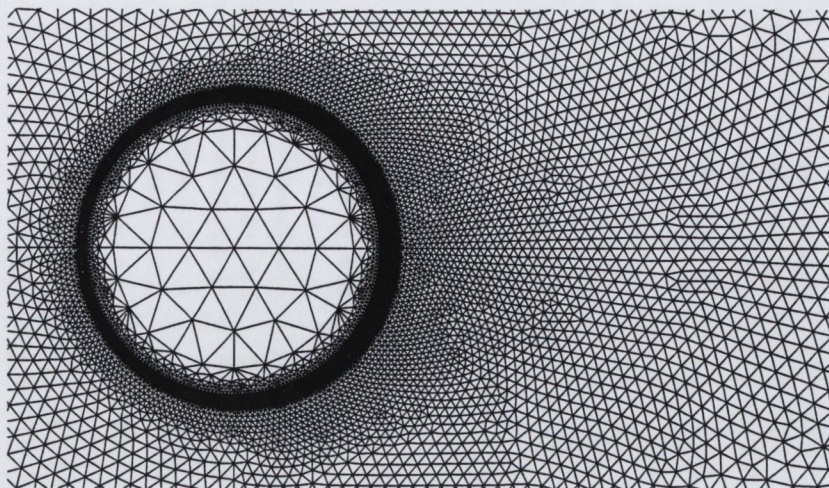


Figure 5.9: The recirculation region.

area will allow the behaviour of the recirculating regions to be resolved accurately. The recirculating region of a mesh is shown in Figure 5.9.

5.2 Defining zones in the mesh

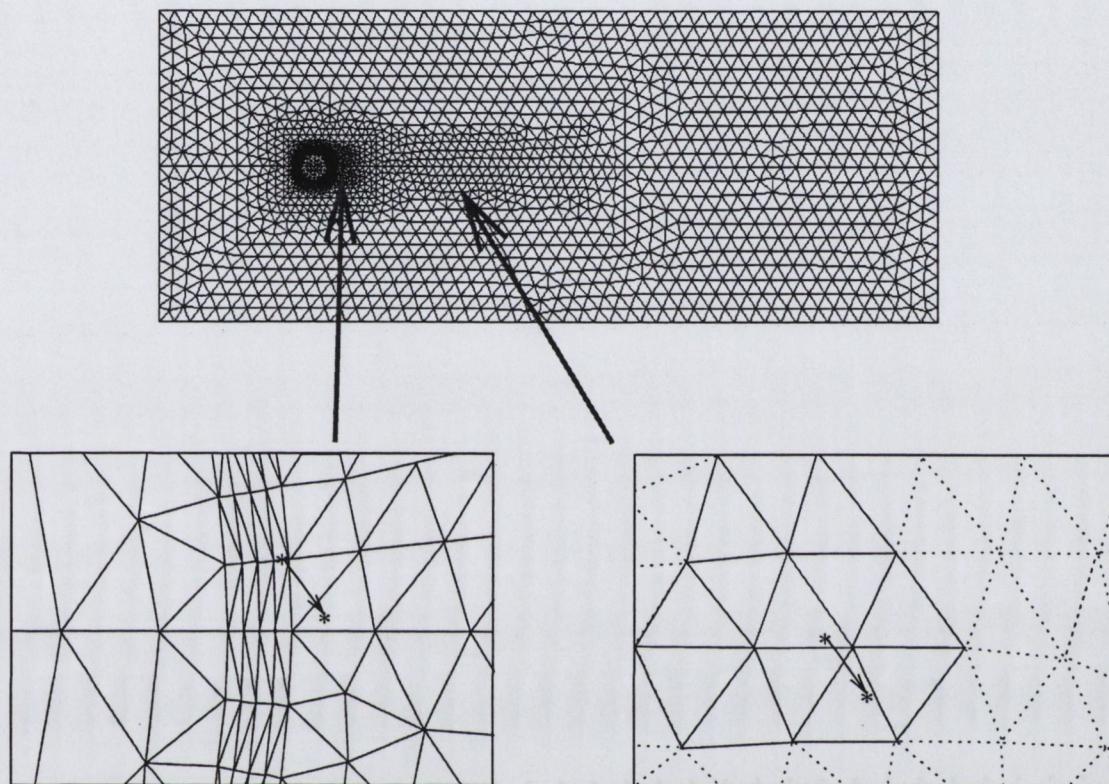
A useful function of GAMBIT, apart from simply generating a mesh, is the ability to define zones. This allows not only the nodal coordinates and element connectivities to be exported, but also allows boundaries and regions to be grouped together and named. These zones can then be exported along with the nodes and connectivities. The zones which it is useful to designate are:

- the top and bottom nodes - the list of nodes along the top and bottom of the mesh at which the stream function will be defined, to enforce the required onset velocity condition. In the present work, the inflow and outflow nodes had no boundary condition explicitly enforced, as the natural boundary condition ensures zero normal gradient of stream function, resulting in flow parallel to the x axis at the inflow and outflow boundaries.

- the control nodes - the list of nodes inside the surface which will be used to find the tangential velocities which define the circulation required to enforce no-slip. If zero-permeability is being enforced by setting the value of the stream function to a constant, then the stream function values will be set at these control nodes.
- the surface nodes - the list of nodes which define the surface of the body.
- internal elements - the list of elements which lie inside the surface of the body. Once the home element of a point vortex has been found, this list can be used to decide if it lies inside the body or not. If inside the body, it must be reflected or absorbed.
- graphical look-up elements - the list of elements contained in the region for which the graphical look-up matrix will be generated.

The last of these zones is required because the graphical look-up matrix is not generated for the entire mesh. The smallest element size is of the order of the standard deviation of the random walk. Therefore the step size required when building the graphical look-up matrix should be a fraction of this length. To cover the entire mesh by moving in steps of such a small length would require a very large graphical look-up matrix. Instead of doing this, the graphical look-up matrix is only generated in the region where the element sizes are small. Inside this region, a point vortex may easily move from its home element to a position a number of elements away in a single time step. If the element size is relatively large compared to the distance that a point vortex is expected to travel during one time step, then the point vortex will either be found in the same element at the end of the time step, or else in one of the neighbouring elements. In this case, the graphical look-up matrix is no longer needed. The first guess for the home element after convection and diffusion can be the previous home element of that point vortex. If this guess is incorrect, the neighbours matrix can be used to test

which one of the bordering elements the point now lies in. Hence the new home can be found without need for the graphical look-up matrix. This is illustrated by Figure 5.10.



(a) Vortex motion near the surface. (b) Vortex motion when the element size is larger.

Figure 5.10: Situations which (a) do, and (b) do not require the graphical look-up matrix.

In Figure 5.10(a) the point vortex lies near the surface of the cylinder, and has moved to a non-neighbouring matrix during the time step. The previous home element offers no information as to the new home element, and the graphical look-up matrix is required. In Figure 5.10(b), which shows a region further away from the surface, the point vortex will not move outside the elements defined by the previous home element and its neighbours during one time step, and so the graphical look-up matrix is not needed.

5.3 Blockage effects

The value of the stream function is set to equal the y -coordinates of the nodes along the top and bottom boundaries of the flow field. This results in flow parallel to the x axis at these boundaries.

When meshing the flow field, it is important to make sure that the boundaries are not too close to the solid surface. Continuity will result in an increase in the velocity of a constrained flow as the cross-sectional area decreases. Hence, if the boundaries are too close to the solid surface in the simulation, they may cause unwanted effects in the velocity field.

This blockage effect is typically characterised by the *blockage ratio*, the distance between the upper and lower boundaries divided by cylinder diameter. Stansby & Slaouti [27] showed that for the random-vortex method with blockage ratios of 16 or over, the effect of the boundaries on the calculated velocities and forces is small. The blockage ratio used in this work is always at least 20. As an illustration of this effect, Figure 5.11 shows the starting surface velocity for the cylinder calculated with a blockage ratio of 2 and 20, and compares these to the analytical solution of potential flow around a cylinder.

5.4 The outflow boundary

A problem with keeping the boundary distant from the point vortices occurs when a simulation is run over a large number of time steps. The point vortices will get closer to the outflow boundary condition as the simulation progresses. In the present work, this is dealt with by extending the flow mesh so that the outflow boundary remains distant from the point vortices. For simulations which will be run for long times, the outflow boundary was extended to 60 diameters downstream, allowing a simulation run to $t^* = 100$ without point vortices approaching the outflow boundary. The areas near the boundary and the near wake require a

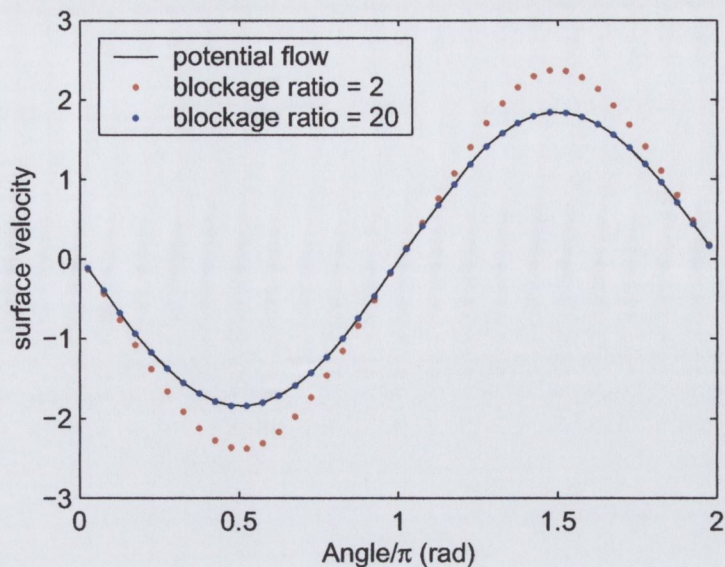


Figure 5.11: Blockage effects.

high density of elements to obtain a good resolution of the velocity field, but larger element sizes may be used in the wake where the flow pattern is expected to be more regular, and a high resolution of the velocity field is not required. Extending the outflow boundary downstream need not require a large number of extra elements, and hence need not impose a large penalty.

5.5 Processing the mesh

Once the mesh has been generated, it needs to be exported in a form which can be read by other programs. The mesh data can be exported from GAMBIT by using the neutral (*.neu*) export format. This results in an ASCII file which contains the nodes, connectivities and zones required to run the simulation. MATLAB is used to continue preparing the mesh for the simulation.

The first major operation is to assemble the global stiffness matrix for the finite element mesh. During this process the components of the shape function for each element are calculated. These values are saved to a file, as they will be needed for

area-weighting and calculating the velocity from the stream function during the simulation. Once the global stiffness matrix and the loading (circulation) vector have been assembled and the boundary conditions applied, they are saved. As they are both sparse, only the values and indices need to be saved. The MATLAB program then generates the graphical look-up matrix for the area of small mesh size where one is needed, defined by the relevant GAMBIT zone, and finally generates the neighbours matrix for the whole mesh.

All of the information required to run the simulation is now available, and the next chapter will describe the algorithm step-by-step.

Chapter 6

Stepping through the program

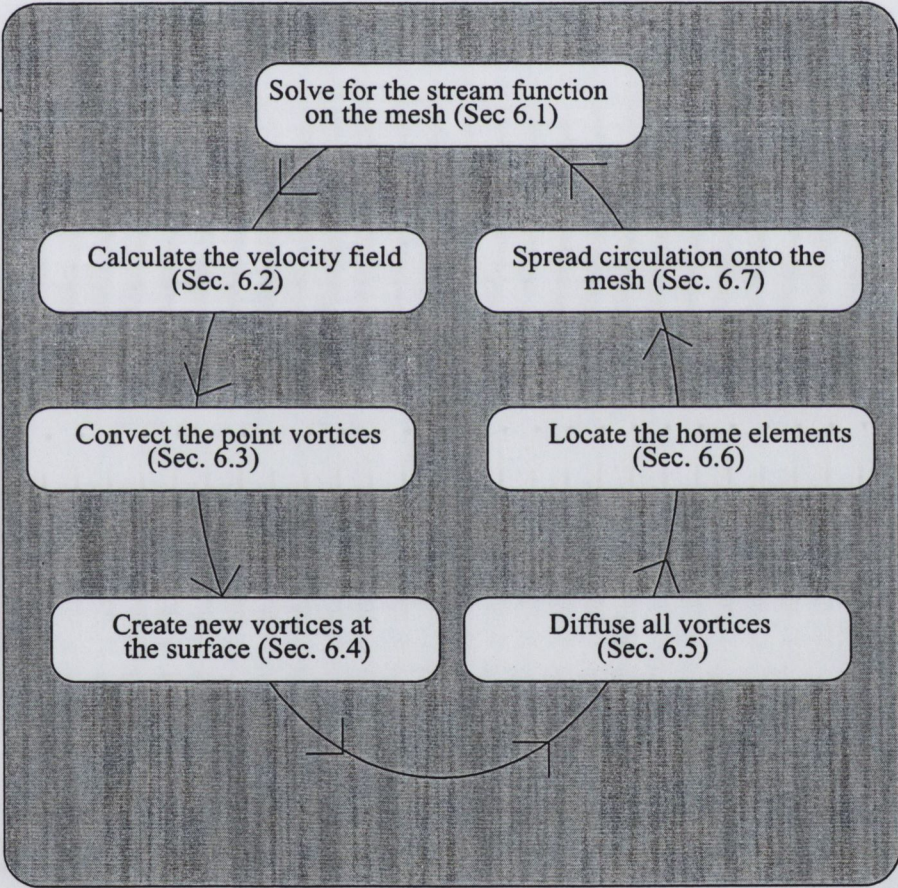


Figure 6.1: The algorithm.

In this chapter, the main operations of the simulation will be described sequentially, as they are encountered in the simulation program, showing how the theory of the previous chapters is put into practice. Figure 6.1 shows a flowchart of the algorithm. The simulation was written in FORTRAN90, and uses the PETSc suite of data structures and routines [19] to efficiently solve the system of linear equations defined by the large sparse global stiffness matrix and the loading vector of the values of circulation at the nodes of the mesh, hence obtaining the values of stream function at the nodes of the mesh.

6.1 Solve for the stream function on the mesh

Once the FORTRAN program has loaded all of the program parameters, and the data from the files prepared by MATLAB, the algorithm is ready to run. The first operation involves solving for the nodal values of stream function. The PETSc sparse linear equation solver returns the vector of nodal stream functions from the global stiffness matrix and the vector of nodal circulations. For the first step, there will be no circulation in the flow, and the solution will be equivalent to uniform potential flow about a circular cylinder. A streamline plot of the calculated starting flow is shown in Figure 6.2.

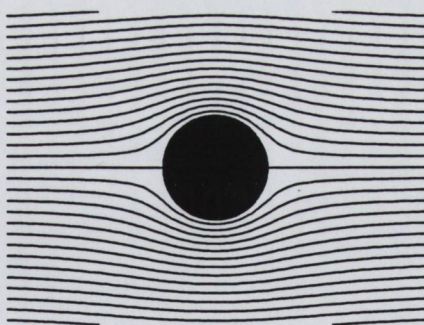


Figure 6.2: Streamlines for starting flow.

6.2 Calculate the velocity field

Once the stream function is known at the nodes of the mesh, the velocity field can be calculated by making use of Eq. (4.18).

$$u = \frac{1}{2A} \sum c_i \psi_i \quad , \quad v = -\frac{1}{2A} \sum b_i \psi_i$$

As noted before, the area of an element and the components of the shapefunction of that element are constant for that element. Hence the velocity calculated is uniform across each element. Figure 6.3 shows the velocity vectors plotted at the centroids of the elements for starting flow near the cylinder surface. As can be seen, the flow inside the cylinder is zero and zero-permeability has been enforced.

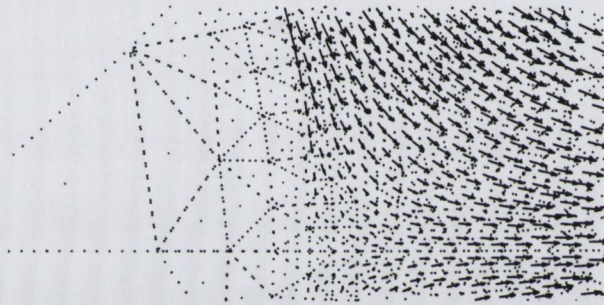


Figure 6.3: Calculated velocity field near the surface.

6.3 Convect the point vortices

For the first time step in the simulation, there will be no point vortices in the flow to convect. For later time steps, there will be point vortices present, and the home elements of these point vortices will be known from the previous time step, when they were located in their home elements and had their circulation spread onto the map. The convection process then simply involves moving the point vortices by the velocity of their home element for that time step. Figure 6.4 shows the convection of some point vortices near the surface.

$$x_{new} = x_{old} + u_{home} * \Delta t$$

$$y_{new} = y_{old} + v_{home} * \Delta t$$

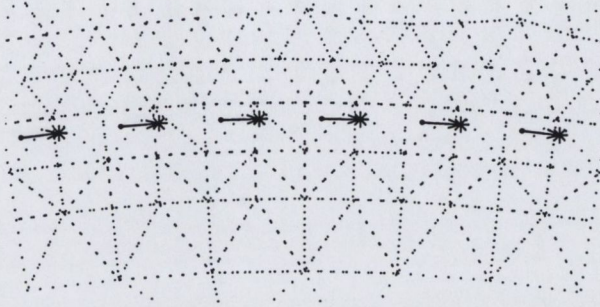


Figure 6.4: Convecting point vortices.

6.4 Create new vortices at the surface

New point vortices will be created at the midpoints of the surface panels to enforce the no-slip boundary condition, with strengths set from midpoints of the control panels slightly inside the surface. A circulation cut-off value is used to limit the maximum value of circulation which any one point vortex may have. If the circulation which needs to be created at a surface panel is greater than the circulation cut-off, then a number of point vortices will be created until the required circulation has been introduced. As the cut-off value is decreased, more point vortices will be introduced into the simulation. This may give a more accurate representation of the flow field, but will also incur the penalty of higher computational cost. This cut-off value is for the circulation in the simulation, which, being based on non-dimensional velocity, has units (m). A cut-off value of $0.001m$ was used for all simulations in the present work, and was found to give a good balance of resolution of the flow field and computational efficiency.

Figure 6.5(a) shows the velocity field near the surface before the new vortices have been created. The surface nodes are represented by the black stars. In Figure 6.5(b) the velocity field has been calculated including the effect of the newly created point vortices at the surface. It can be seen that the velocity inside the surface has been effectively reduced to zero, as required by the no-slip condition. This velocity calculation has only been performed here to show the effect of the newly created vortices. In the simulation, all point vortices are diffused before a new velocity field is calculated.

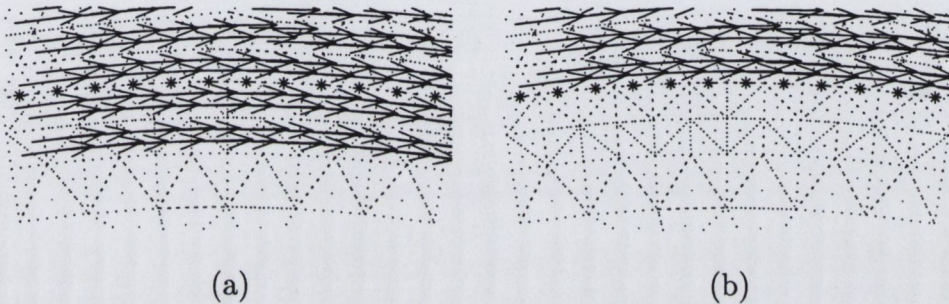


Figure 6.5: Enforcing the no-slip condition.

6.5 Diffuse all vortices

The next step in the simulation involves diffusing all point vortices. Point vortices in the flow are diffused by adding a random walk to their position (Figure 6.6(a)), while point vortices newly created at the surface are diffused along the outward normal from the surface, to avoid them walking inside the solid boundary (Figure 6.6(b)).

6.6 Locate the home elements

(Figure 6.7) shows a typical region for which a graphical look-up matrix has been generated. When finding the home elements of the point vortices, those point vortices whose coordinates lie within the boundaries of this region will have their

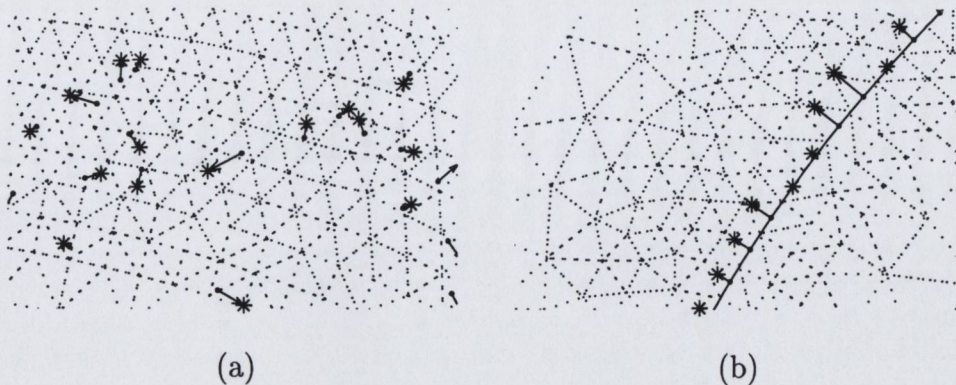


Figure 6.6: Diffuse all vortices.

home element estimated by the graphical look-up matrix. If the estimate is incorrect, the neighbours matrix will be used to find the correct home element. Those point vortices which lie outside the region covered by the graphical look-up matrix will have their previous home element used as an initial guess, and if incorrect, the neighbours matrix can be used to find the correct home element.

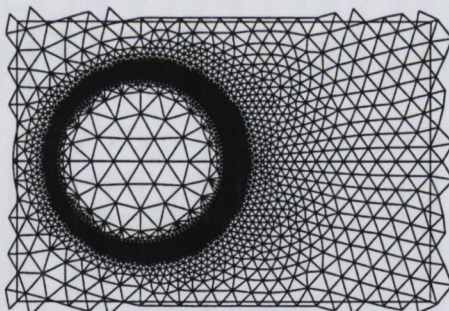


Figure 6.7: Typical region covered by graphical look-up matrix.

6.7 Spread circulation onto the mesh

Point vortices whose home elements are listed as being inside the solid boundary are deleted, and their circulation is noted. This internal circulation is then spread onto the surface nodes of the mesh, so that it is included in the next solution of the Poisson equation. When creating new point vortices at the next time step,

this circulation will be reintroduced into the flow. All other point vortices have their circulation spread onto the nodes of their home elements. The program is now ready to solve the Poisson equation for the new stream function values on the mesh, and returns to the first operation in the algorithm.

6.8 Profiling the program

Time accounting information was recorded during a typical simulation to highlight the relative amount of time spent calculating different sections of the simulation. The time information was generated from consecutive steps of the algorithm, and the random component present may be due to background processes, and intermittent reporting lags. This time profiling is shown in Figure 6.8.

The processes of solving the Poisson equation, and the calculation of the velocity for each element are seen to take the same amount of time throughout the simulation. This is to be expected, as these operations are carried out on the unstructured mesh. Thus, the calculation of the velocity field is independent of the number of vortices in the simulation.

The remaining processes are split into three groups; creating, convecting and diffusing point vortices, locating point vortices in the mesh, and spreading circulation onto the mesh. All of these operations take a greater amount of time as the simulation progresses, and the overall number of point vortices involved increases. The most efficient of these operations is that of locating the point vortices in the mesh, which increases in a linear manner with the number of vortices present. The process of creating, convecting and diffusing point vortices follows a similar trend, but consistently takes longer to perform than locating the point vortices. The process which takes the largest share of the cycle time is seen to be that of spreading the circulation of the point vortices onto the nodes of their home elements. The time required to perform this process also increases with the number of vortices present, although it does not increase as sharply as the previous two processes.

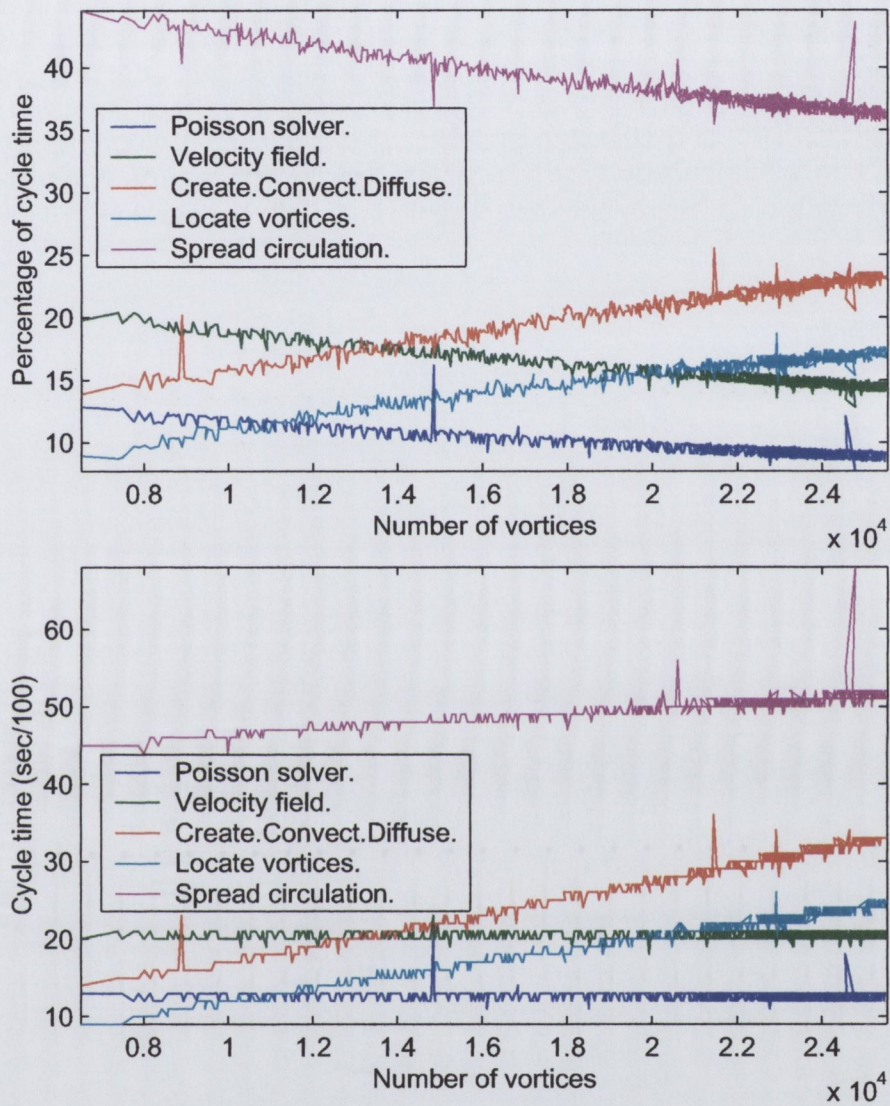


Figure 6.8: Typical simulation time profile.

The time profile information shows that all processes involved are either independent of N , or vary linearly with N . Hence the entire process is linear in N , as was seen by the total process time plot presented in Figure 5.8.

6.9 Summary

In this chapter, the algorithm to be used in the simulations was presented. The main operations were discussed sequentially, illustrating how the theory of the present method is applied in practice. Information about the process time was presented for a typical simulation run, showing overall process time to be $O(N)$. In the next chapter, simulations will be run to investigate the agreement between the present method and other computational and experimental results.

Chapter 7

Results

In order to validate the computer code, simulations have been run over a range of Reynolds numbers from 40 to 9,500. The results obtained by this method are compared to other published results. Although many papers have been published for the case of impulsively started flow over a circular cylinder, the benchmark paper for high-resolution simulations is that of Koumoutsakos & Leonard [13], and results generated here are compared with their computed results. Coefficients of lift and drag, surface circulation, vorticity and stream function contours were generated. Plots of the computed Strouhal number and mean drag variation with Reynolds number are compared with experimental results obtained from the comprehensive review on flow around circular cylinders by Zdravkovich [29]. Finally, results of a simulation showing the application of this method to flow around a multiply connected domain are presented.

7.1 Reynolds number = 40

For $Re = 40$ the time step used (all times are non-dimensionalised) was $\Delta t = 0.02$, and the mesh spacing ratio (given by dividing the normal mesh spacing at the surface by the standard deviation of the random walk) was $MR = 0.5$. Point vortices which are created at the surface move towards the rear stagnation point and induce a reversal of flow direction (Figure 7.1(a)). The flow separates from the surface of the cylinder, and forms two primary recirculating regions (Figure 7.1(b)). These recirculating regions grow with time (Figure 7.1(c)), until the flow achieves an almost steady symmetrical closed near-wake. The free shear layers meet at the end of the near-wake at the *confluence point*. The elongated closed near-wake eventually becomes slightly unstable, with a sinusoidal oscillation of shear layers commencing at the confluence point (Figure 7.1(d)).

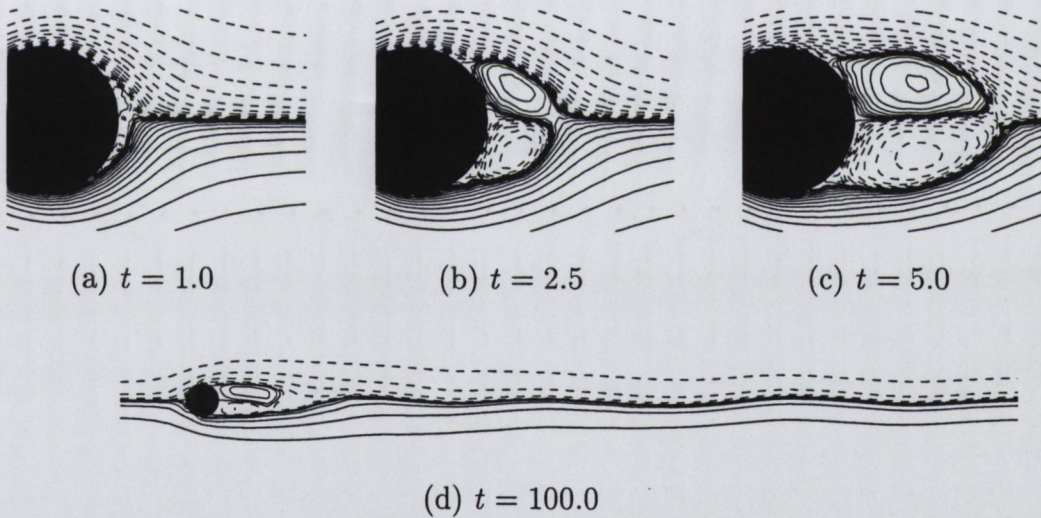


Figure 7.1: Streamlines for $Re = 40$.

The drag coefficient has been calculated from the locations and strengths of the point vortices in the wake, following the method described by Koumoutsakos & Leonard [13] (hereafter referred to as KL). Figure 7.2 shows the computed drag coefficient for the present calculations and those of KL. The drag coefficient was ensemble averaged over 10 runs to reduce the random component. The response to

the impulsively started flow is slightly slower than KL, but agreement after $t = 2.0$ is within 5%.

$$C_D = - \frac{d}{dt} \sum_{i=1}^N \Gamma_i y_i$$

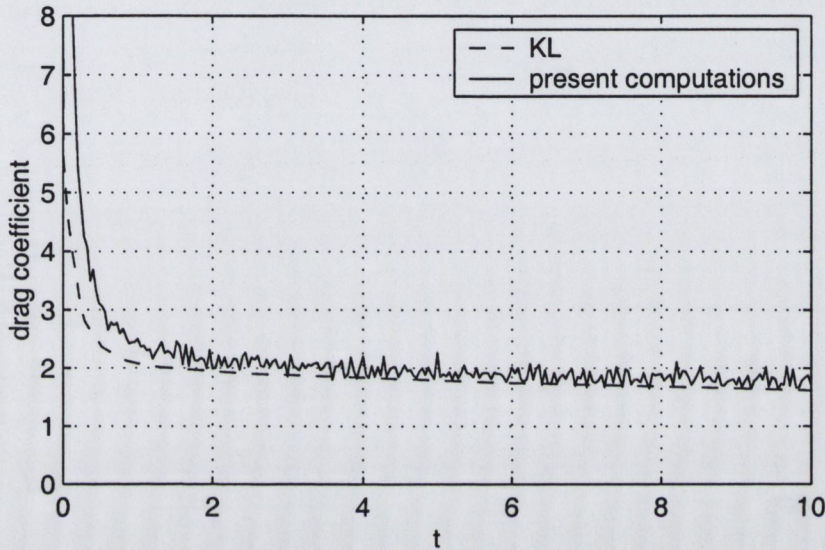


Figure 7.2: Drag coefficient, $Re = 40$.

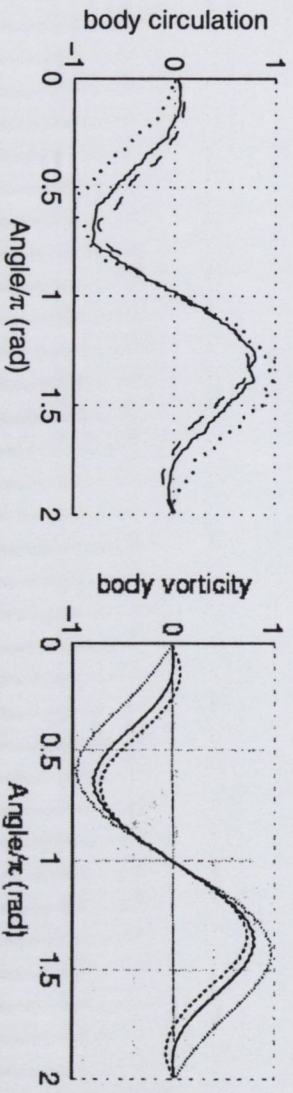
The creation of vorticity at the surface of the body to satisfy the no-slip condition is a crucial step in vortex methods. In the present work, vortices are created with a defined value of circulation to enforce the no-slip condition. The magnitude of the circulation created at the surface depends on the tangential velocity at the control nodes, which in turn is related to the magnitude of the time step. If a small time step is used, the point vortices created at the previous time step will not have moved far from the surface, and the magnitude of the tangential velocity calculated, and hence circulation created, will be relatively small. For larger time steps, the point vortices may have moved further from the surface and larger tangential velocities may be calculated, and hence larger values of circulation will be required. For this reason, plots of circulation at the surface of the body have

been normalised about the maximum values. In KL, plots of body *vorticity* are used, rather than circulation. However, as surface circulation and vorticity can be related by a constant term, it is the comparison of the trends of the present results and those of KL that are of interest. Later plots of body circulation are on different axes to avoid cluttered graphs, and to allow easy comparison with the plots of KL.

Figure 7.3 shows the calculated body circulation of the present work to be in good agreement with the vorticity calculated in KL. The angle is measured in a counterclockwise direction from the rear stagnation point of the cylinder. When the recirculating zone has developed, the body circulation is initially positive due to the counterclockwise tangential velocity at the surface induced by the primary eddy. As the angle approaches the separation point, the circulation reduces to zero, and becomes negative when moving from the recirculation region into the onset flow. The circulation reduces to zero again at the front stagnation point. The plot of body circulation is symmetric for the second half of the cylinder surface.

The vorticity in the wake of the cylinder was also recorded during the simulation, and is compared to the equi-vorticity contours of KL in Figure 7.4. The vorticity in the recirculating zone is seen to remain symmetrical and attached to the cylinder.

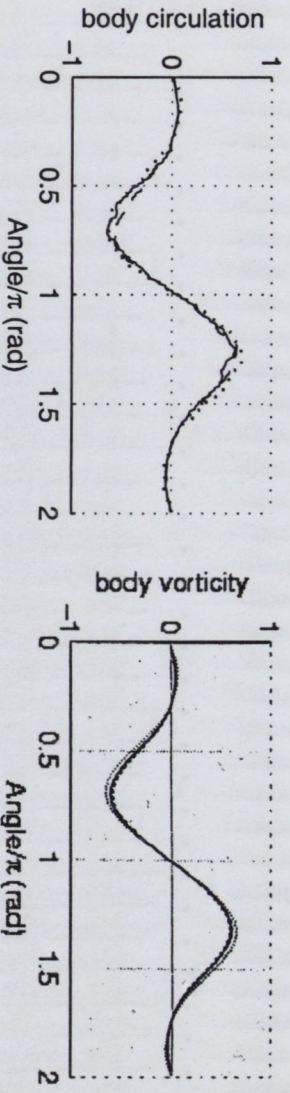
($\cdots$) $t = 0.4$, ($—$) $t = 1.0$, ($- -$) $t = 2.0$



(a) Present work.

(b) KL [13].

($\cdots$) $t = 5.0$, ($—$) $t = 10.0$, ($- -$) $t = 15.0$



(c) Present work.

(d) KL [13].

Figure 7.3: $Re = 40$, body circulation / body vorticity.

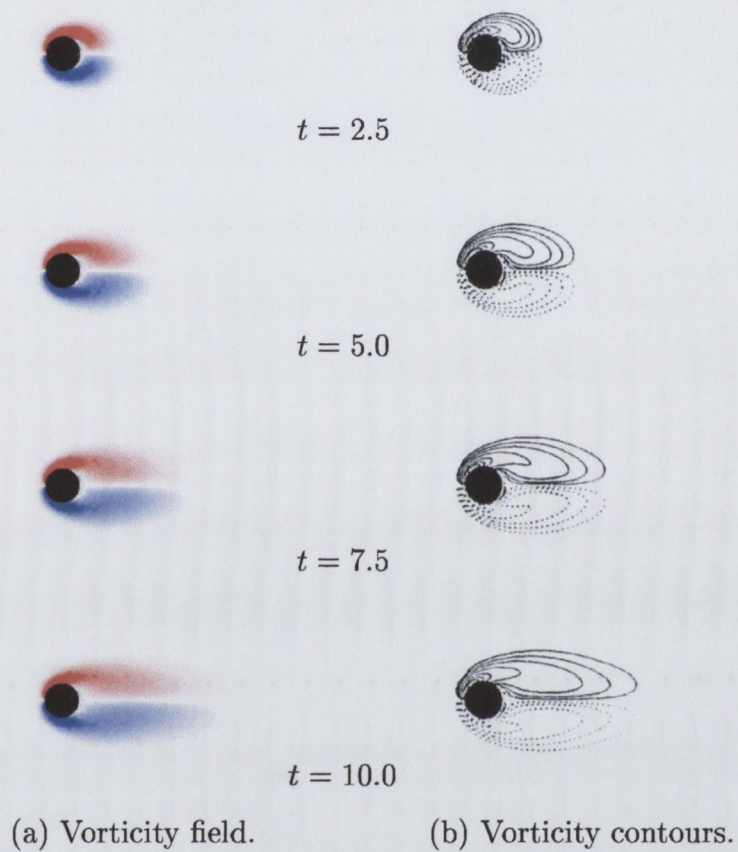


Figure 7.4: $Re = 40$: (a) Present work. (b) KL [13].

7.2 Reynolds number = 550

For $Re = 550$ the time step used was $\Delta t = 0.01$, and the mesh spacing ratio was $MR = 0.7$. At this Reynolds number, a bulge which is initiated at the surface of the cylinder slightly behind the separation point, Figure 7.5(a), will separate from the surface of the cylinder as the magnitude of the back flow increases. This will give rise to a small secondary eddy, which has a rotation opposite to that of the primary eddy, Figure 7.5(b).



(a) Appearance of bulge. (b) Secondary eddy.

Figure 7.5: $Re = 550$, secondary eddy formation.

The presence of this secondary eddy causes a change in the trend of the body circulation, shown in Figure 7.6(c). The circulation is positive in the region of the primary eddy, then reduces to zero and becomes negative for the small region of the secondary eddy. The circulation again becomes positive when leaving the secondary eddy, as the primary eddy still causes counterclockwise flow between the secondary eddy and the separation point. From the separation point, where the circulation is zero, the circulation follows the same trend as before, becoming negative due to the onset flow and returning to zero at the front stagnation point.

The calculated coefficient of drag is compared to that of KL in Figure 7.7. Again, the initial time lag of the present computations is followed by a good agreement of predicted drag, within 5% of KL.

For this Reynolds number, a plot of the instantaneous streamlines at $t = 5.0$ predicted by the present work (Figure 7.8(a)) is compared to the streamline plots of KL (Figure 7.8(b)) and that of Smith & Stansby [23] (Figure 7.8(c)), and the

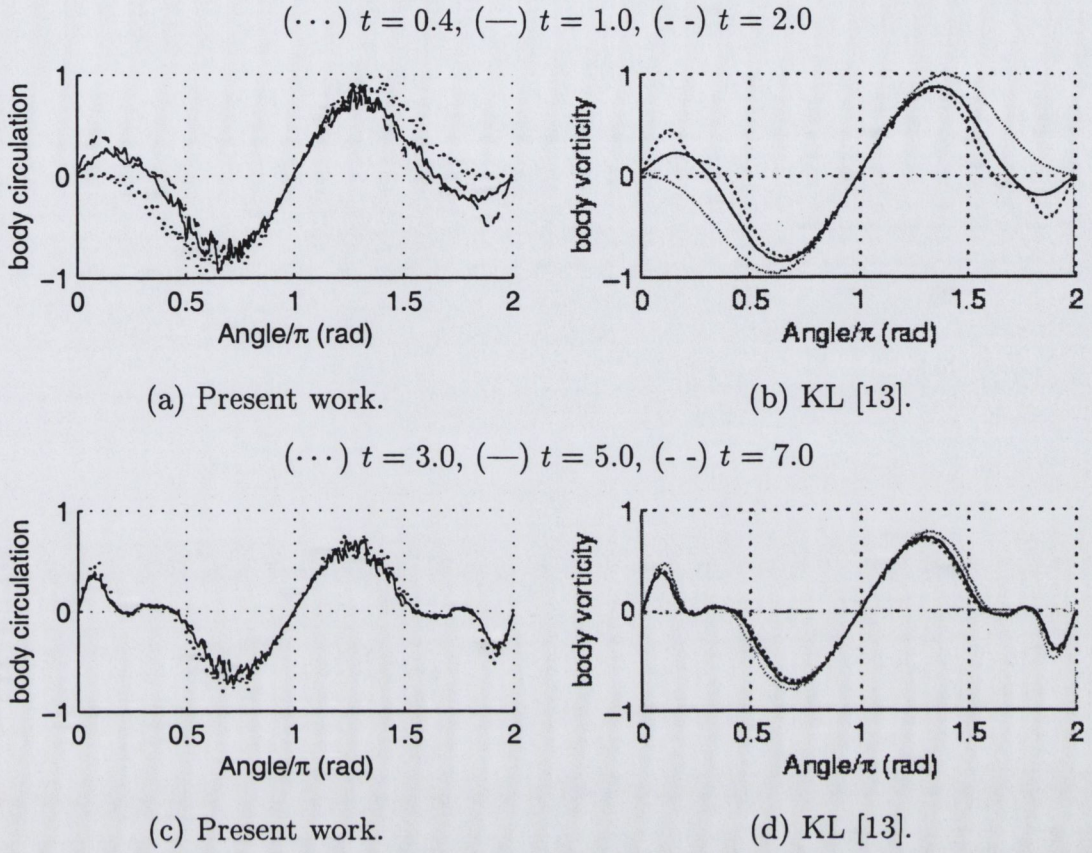


Figure 7.6: $Re = 550$, body circulation / body vorticity.

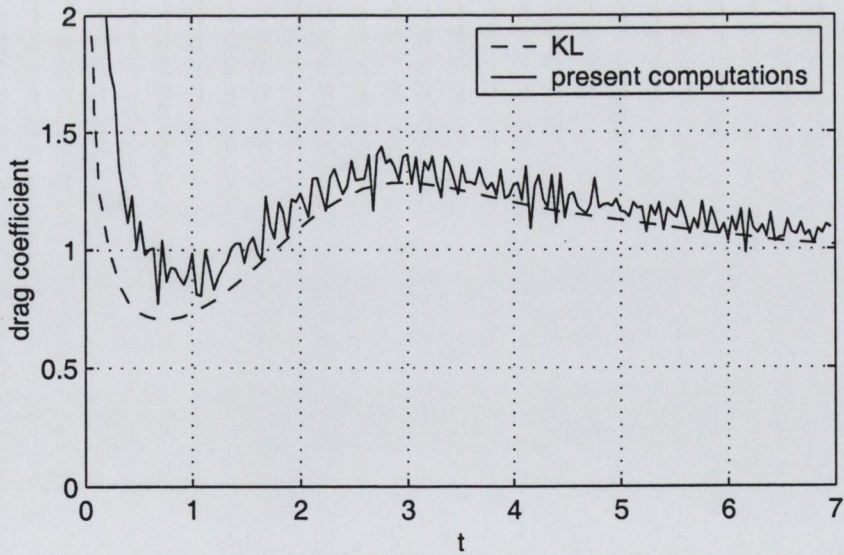
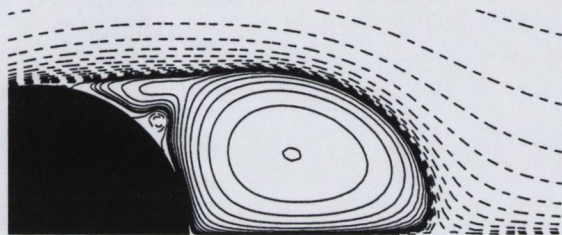
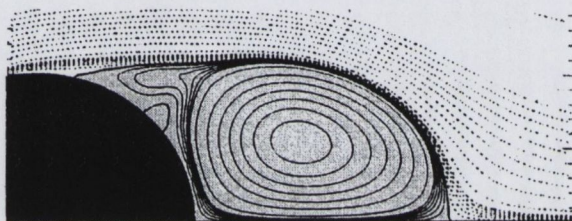


Figure 7.7: Drag coefficient, $Re = 550$.

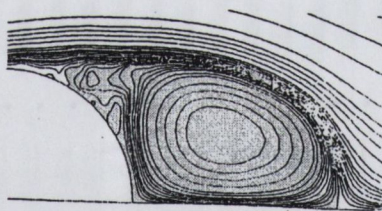
experimental flow visualisation of Bouard & Coutanceau [4] (Figure 7.8(d)). The images have been scaled to give the same cylinder size. Note that the outline of the cylinder is not wholly visible on the photographs of Bouard & Coutanceau because of shadow effects. The results are seen to be in good qualitative agreement.



(a) Present computations.



(b) KL [13].



(c) Smith & Stansby [23].



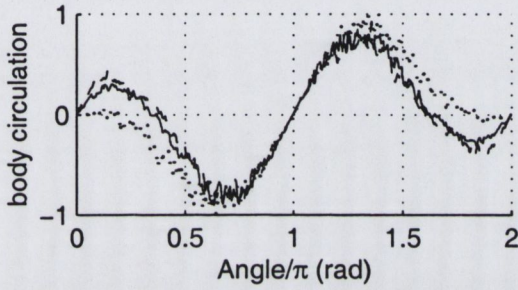
(d) Streaklines of Bouard & Coutanceau [4].

Figure 7.8: $Re = 550$, $t = 5.0$, computed streamlines and experimental streaklines.

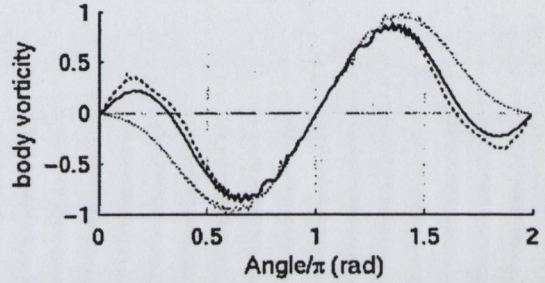
7.3 Reynolds number = 1000

At $Re = 1000$ the time step used was $\Delta t = 0.01$, and the mesh spacing ratio was $MR = 0.95$. The flow develops in the same manner as for $Re = 550$, with more marked effects of the recirculating regions. Figure 7.9(c) shows that the body circulation due to the primary eddy peaks at a higher value than for $Re = 550$. The section of negative circulation due to the secondary eddy has also increased in magnitude, showing good agreement with the predictions of KL.

($\cdots$) $t = 0.4$, ($—$) $t = 1.0$, ($- -$) $t = 1.4$

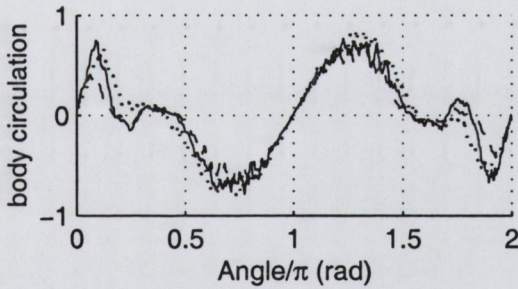


(a) Present work.

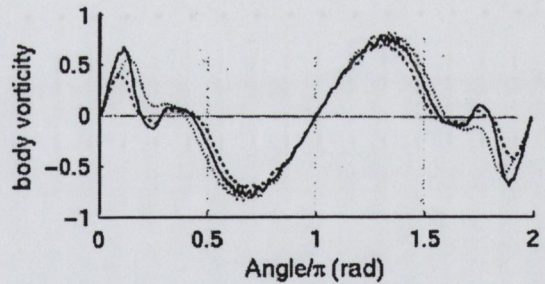


(b) KL [13].

($\cdots$) $t = 2.0$, ($—$) $t = 3.0$, ($- -$) $t = 5.0$



(c) Present work.



(d) KL [13].

Figure 7.9: $Re = 1000$, body circulation / body vorticity.

The calculated drag coefficient is compared to that of Smith & Stansby [23] as well as KL, Figure 7.10. The present work follows the predictions of KL within 7%, after the initial drop in drag. The predictions of Smith & Stansby, using the cloud-in-cell method, show an oscillation of the drag which is not found in KL or in the present work.

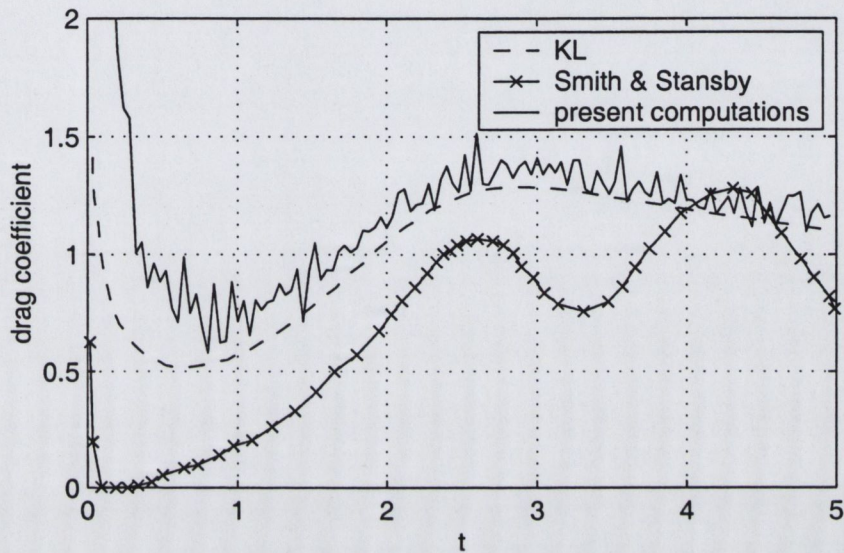


Figure 7.10: Drag coefficient, $Re = 1000$.

7.4 Reynolds number = 3000

Simulations at $Re = 3000$ had a time step of $\Delta t = 0.01$, and the mesh spacing ratio was $MR = 1.6$. At this Reynolds number, the secondary eddy has a larger effect on the flow than for smaller Reynolds numbers. There is a faster roll up of the primary eddy, with stronger circulation, which in turn causes stronger circulation in the secondary eddy, Figure 7.11. The secondary eddy then attempts to penetrate the boundary of the main recirculation zone, restricting the flow into the primary eddy and thus weakening it. This causes a plateau in the drag coefficient, Figure 7.12. The appearance of a tertiary eddy just upstream of the secondary eddy acts to restrain the growth of the secondary eddy, allowing flow to the primary eddy to continue, and the drag increases once more. The growth of the primary eddy is accompanied by a growth in the secondary eddy, causing a reduction in drag towards the end of the plot.

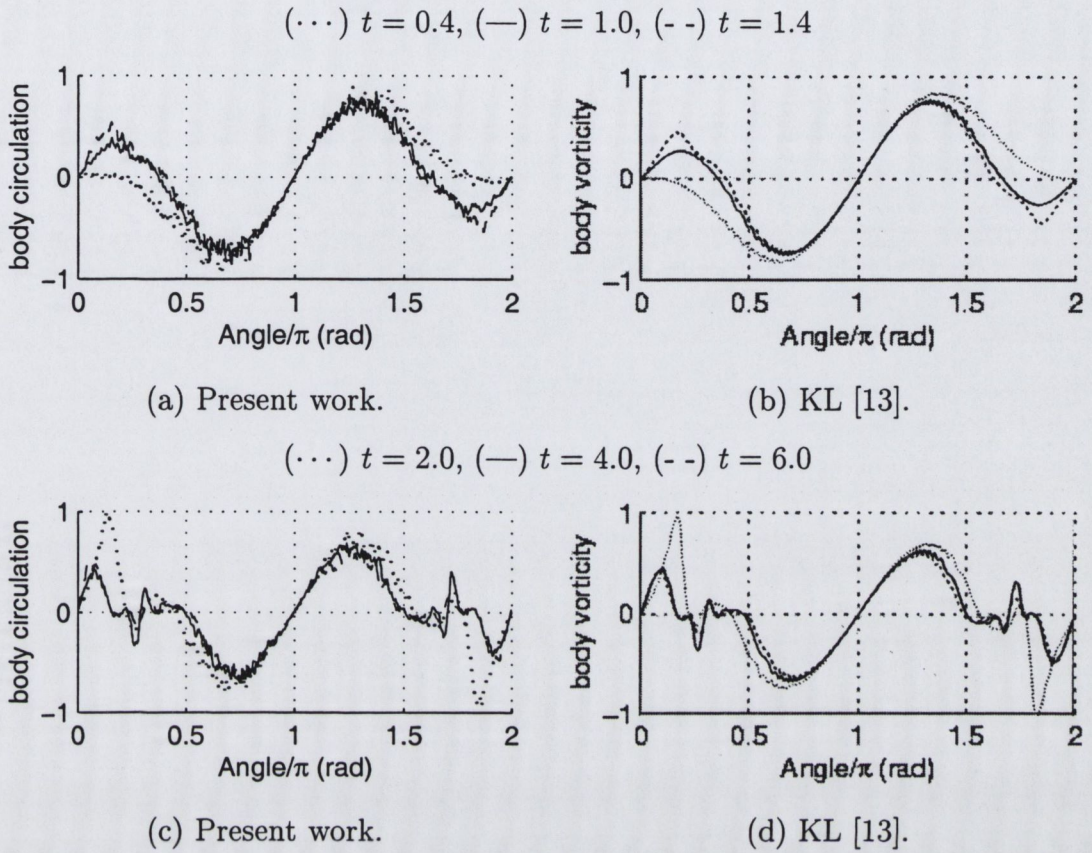


Figure 7.11: $Re = 3000$, body circulation / body vorticity.

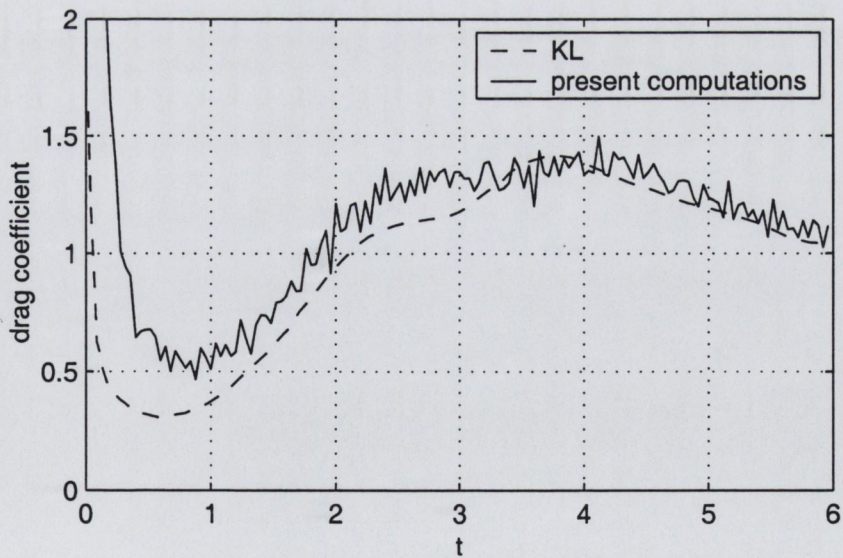


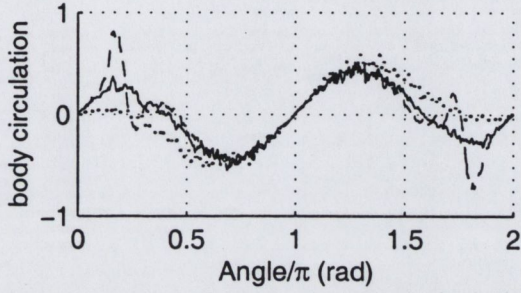
Figure 7.12: Drag coefficient, $Re = 3000$.

7.5 Reynolds number = 9500

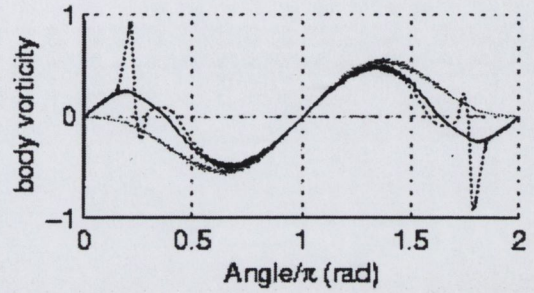
At $Re = 9500$ the time step used was $\Delta t = 0.01$, and the mesh spacing ratio was $MR = 2.9$. The development of the flow at this Reynolds number becomes more complex. There is an early and strong formation of the primary and secondary eddies, shown by the peaks in Figures 7.13(a)&(b). These regions interact and start to move away from the surface, resulting in multiple areas of opposing recirculation appearing at the surface, Figures 7.13(c)&(d). At later times, the present work shows these areas to merge, forming a more stable system of recirculating regions in the wake of the cylinder (Figure 7.13(e)). The simulation of KL, however, shows a greater amount of activity of the body vorticity (Figure 7.13(f)).

Figure 7.14 shows that the increased activity of body circulation is reflected in the behaviour of the drag coefficient. Following the discussions of KL, the rapid formation of a strong secondary eddy which penetrates the feeding vortex sheet of the primary eddy causes the drag to decrease around $t = 2.5$. The drag increases once more when the secondary eddy reaches the outer flow, and the feeding of the primary eddy is re-established. The primary eddy subsequently rolls up and detaches from the body, carrying along part of the secondary eddy. The interplay between the primary and secondary eddies, with the formation of smaller regions of recirculation, causes peaks in the computed drag coefficient at later times in the simulation of KL. The present work, however, shows a less complex recirculating region at later times, and the peaks in the drag coefficient at later times are not predicted.

($\cdots$) $t = 0.4$, ($—$) $t = 1.2$, ($- -$) $t = 1.8$

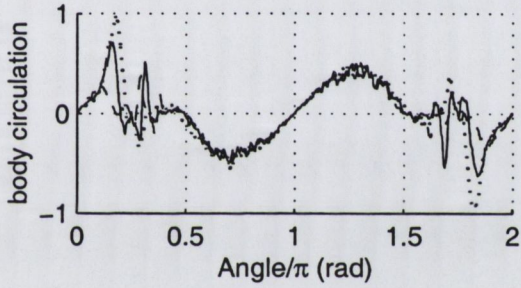


(a) Present work.

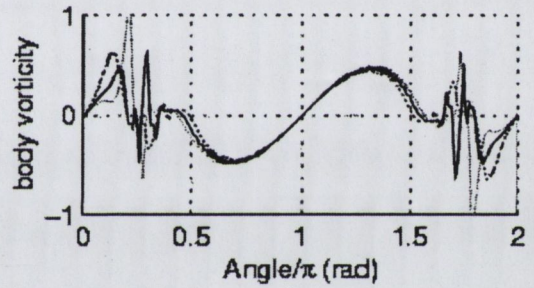


(b) KL [13].

($\cdots$) $t = 2.0$, ($—$) $t = 2.6$, ($- -$) $t = 3.2$

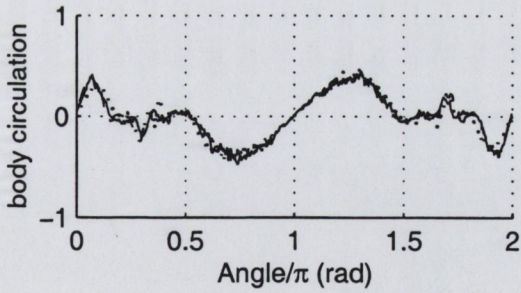


(c) Present work.

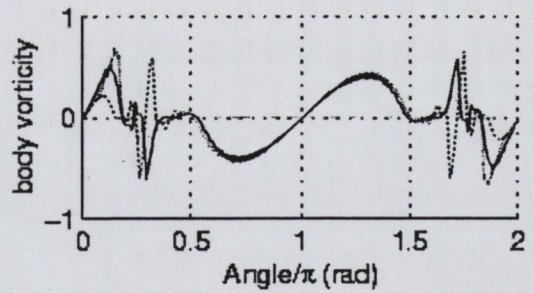


(d) KL [13].

($\cdots$) $t = 3.0$, ($—$) $t = 4.2$, ($- -$) $t = 5.0$



(e) Present work.



(f) KL [13].

Figure 7.13: $Re = 9500$, body circulation / body vorticity.

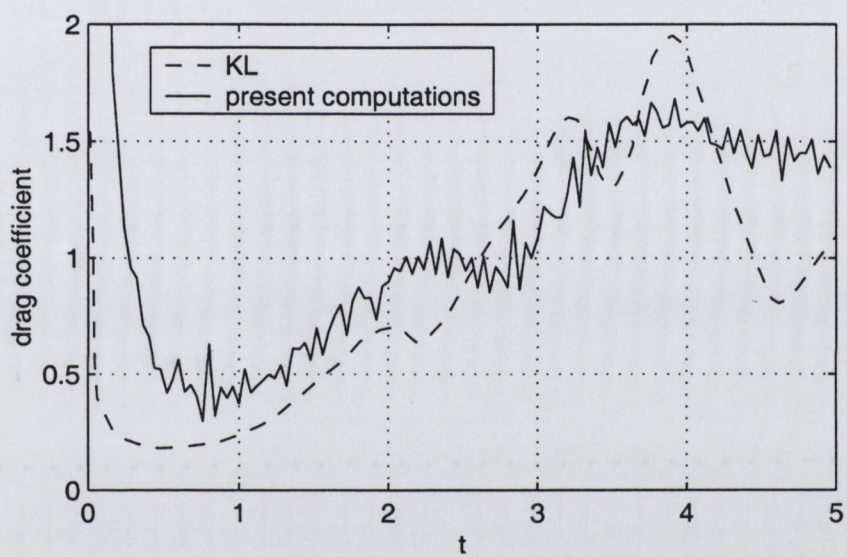
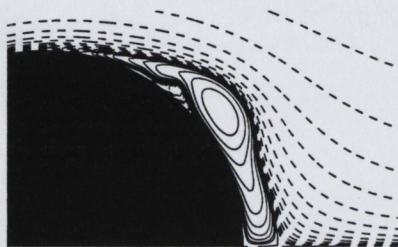
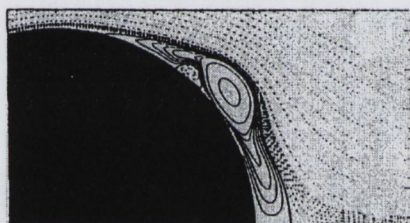


Figure 7.14: Drag coefficient, $Re = 9500$.

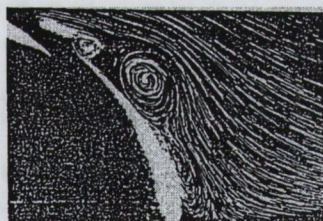
The instantaneous streamlines of the present work, and KL, are compared to the flow visualisation of Bouard & Coutanceau for this Reynolds number in Figure 7.15. A good agreement is seen between the three, with a compact secondary eddy existing near the primary recirculating region, which is found further upstream on the cylinder surface than for lower Reynolds numbers.



(a) Present work.



(b) KL [13].



(c) Streaklines of Bouard & Coutanceau [4].

Figure 7.15: $Re = 9500$, $t = 2.0$, computed streamlines and experimental streaklines.

The vorticity field of the present work is compared to that of KL in Figures 7.16 and 7.17. The overall flow structure is seen to compare well. An initial eruption of the secondary eddy from the surface at early times is followed by a re-established feeding of the primary eddy, with intermittent leaching of circulation from the secondary eddy.

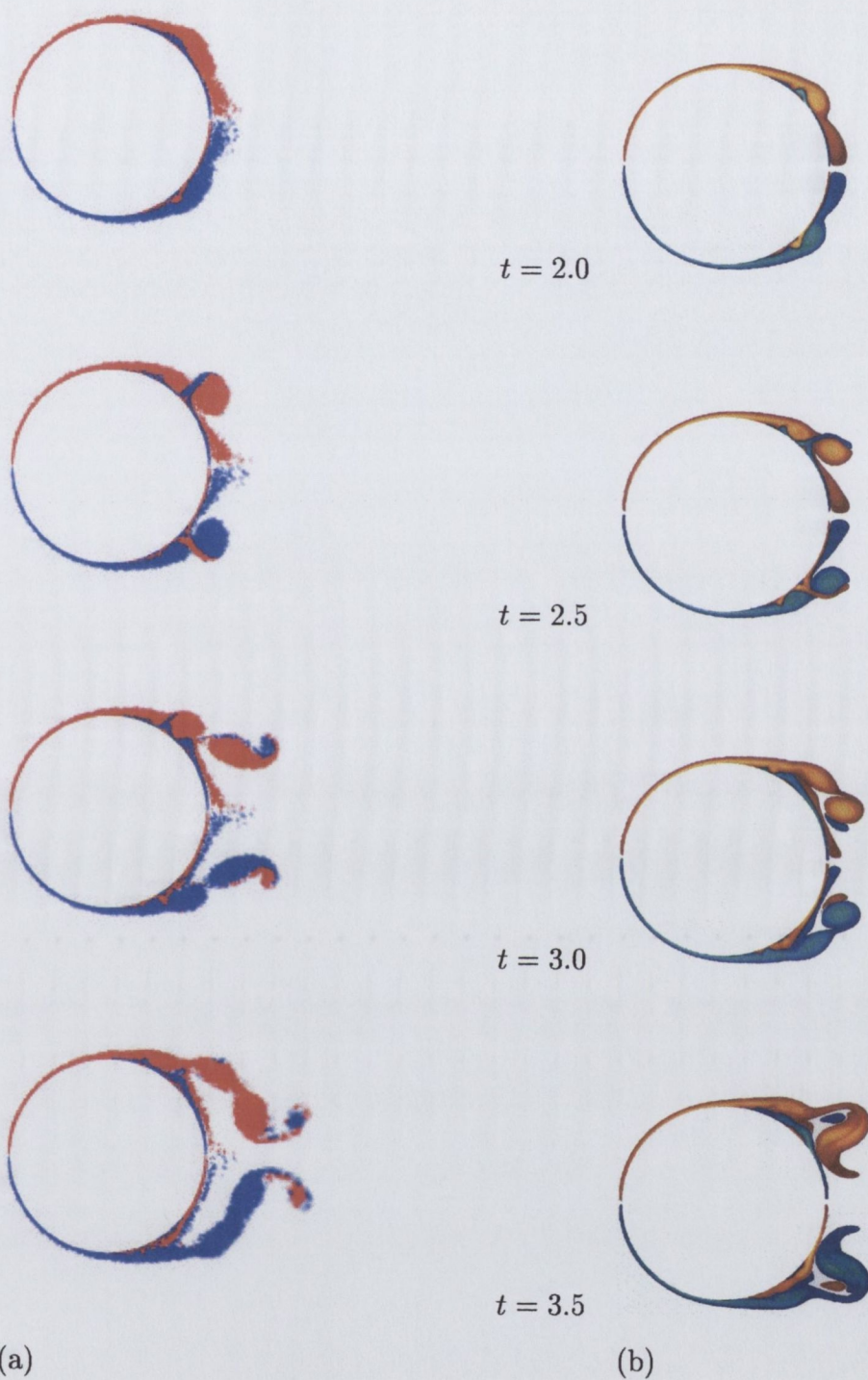


Figure 7.16: $Re = 9500$, $t = 2.0 - 3.5$: (a) Present work. (b) KL [13].

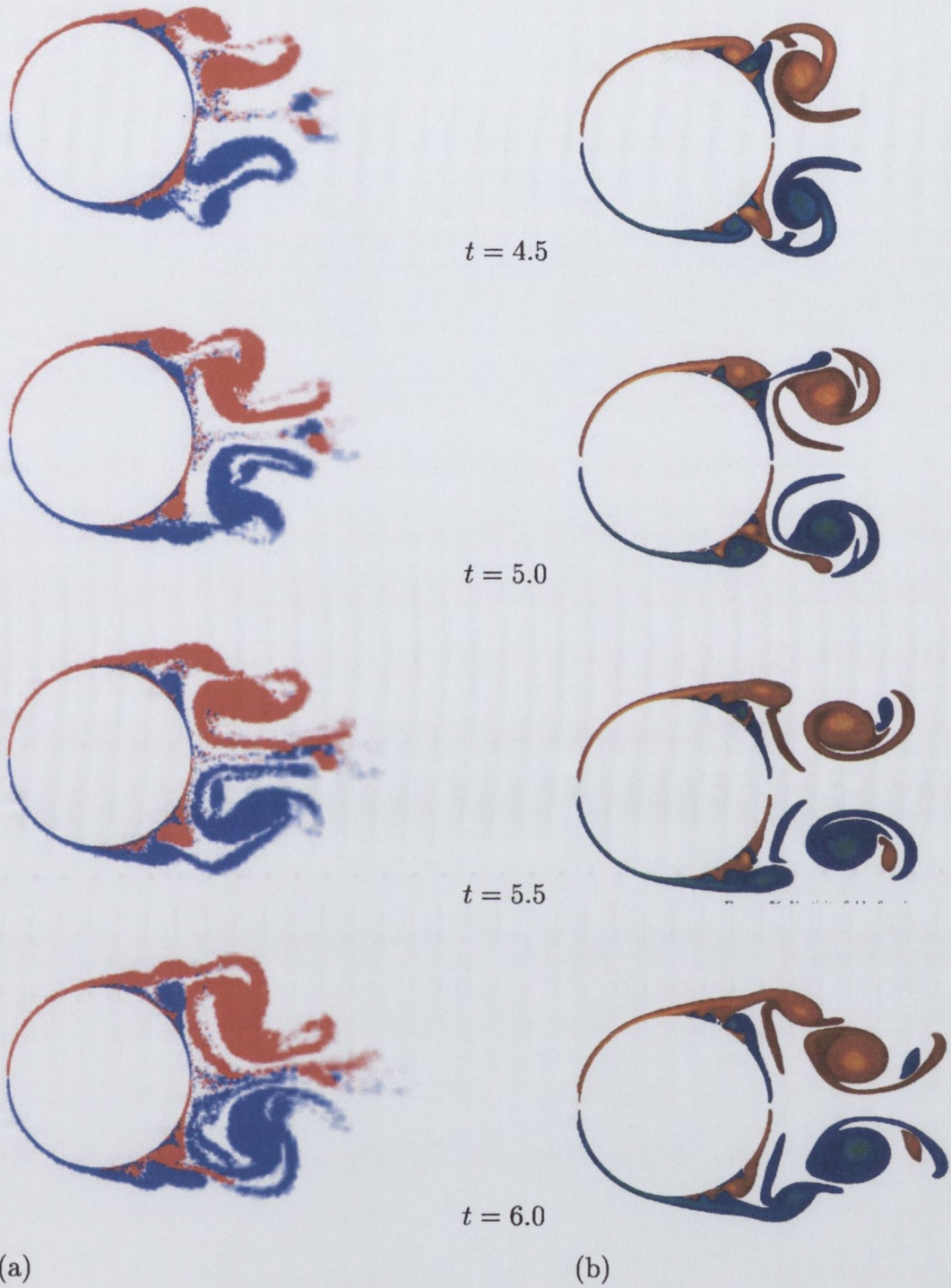


Figure 7.17: $Re = 9500$, $t = 4.5 - 6.0$: (a) Present work. (b) KL [13].

7.6 Strouhal number

Simulations were run over the range of Reynolds numbers from 40 to 9500 in order to obtain values of the Strouhal number and the mean drag. The Strouhal number is a dimensionless parameter which is obtained from the frequency of eddy shedding, f , the free-stream velocity, U_∞ , and the diameter of the cylinder, D :

$$St = \frac{fD}{U_\infty}$$

The Strouhal number was obtained by extending the mesh further into the wake, and running simulations up to $t = 100$. To avoid the large numbers of point vortices which would exist in the simulation after a large number of time steps, point vortices were combined at a distance of 4 diameters downstream. This was done by periodically creating new point vortices at the weighted centre of the uncombined point vortices in that element. The newly created vortices had a circulation cut-off equal to four times that used at the surface. In this way, the total number of vortices in the simulation could be kept low.

The frequency of eddy shedding could be obtained from these simulations, and the Strouhal number calculated. Figure 7.18 shows the comparison of the Strouhal number calculated in the present work to experimental results detailed in the book by Zdravkovich [29]. The present results agree well with the experimental results. The Strouhal number increases from a low value for low Reynolds numbers, to reach a reasonably level value from $Re = 550 - 9500$.

7.7 Mean drag

The same simulations which were used to obtain the Strouhal number yielded values of the mean drag coefficient. Figure 7.19 shows a comparison between the mean drag of the present work, and experimental results [29]. The simulations show a good initial trend, decreasing with increasing Reynolds number, and pre-

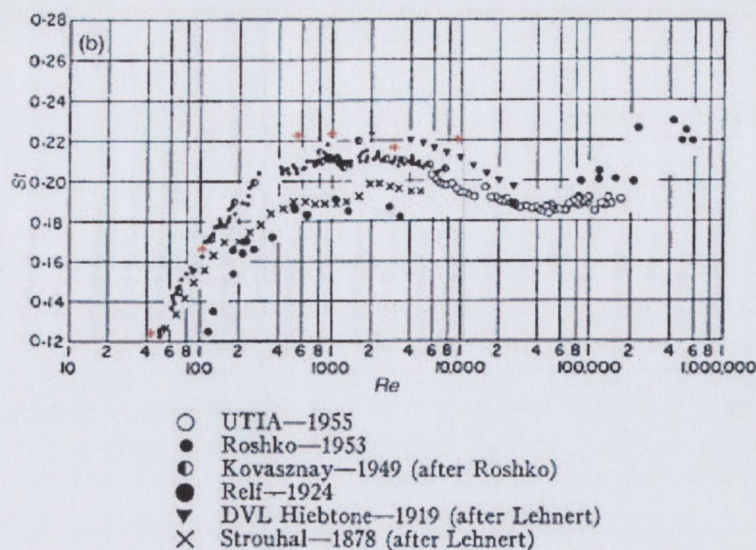


Figure 7.18: Strouhal number: Red stars - present work. Other symbols - experimental [29].

dicting value close to those found experimentally. At higher Reynolds numbers the predictions do not follow the experimental results as closely, but always remain correct within an order of magnitude

7.8 Fluctuating velocity

One of the advantages of computer simulations is the ability to easily define the location at which flow parameters are to be recorded. As an illustration of the type of data which may be required from a simulation, Figure 7.20 shows the magnitude of the velocity in the wake of a circular cylinder recorded at a position 3.5 diameters downstream, and 1 diameter from the centre-line of the flow at $Re = 100$. The periodic nature of the flow is clearly visible, showing the effects of the eddy structures shed from the cylinder surface as they pass this point. The flow is initially unsteady, but later becomes periodic with a fixed amplitude.

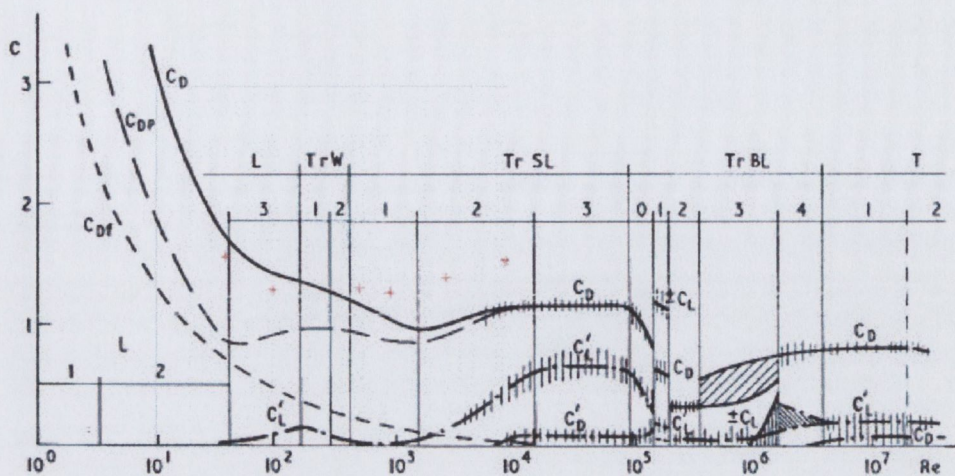


Figure 7.19: Mean drag coefficient: Red stars - present work. Other symbols - experimental [29].

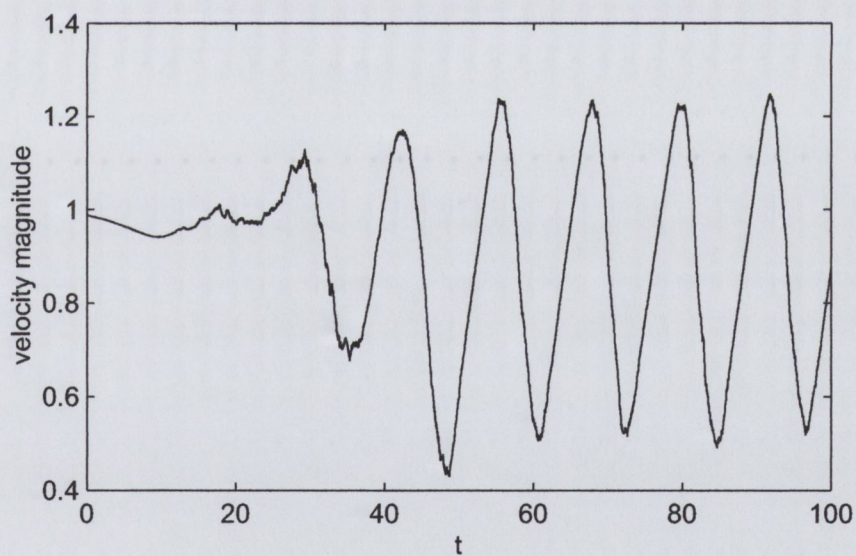


Figure 7.20: Velocity magnitude, $Re = 100$.

7.9 Flow over two cylinders

The motivation for this work was to generalise the application of the cloud-in-cell technique for complex geometries and multiply connected domains by making use of a single unstructured mesh to cover the flow field. To demonstrate this, a simulation was run for the flow over two circular cylinders, with a centre-to-centre longitudinal spacing of 4 diameters, at a Reynolds number of 550.

Figure 7.21 shows the development of the vorticity field for this flow. The flow behind each cylinder develops in a regular manner with time, until the recirculating region of the upstream cylinder approaches the surface of the downstream cylinder. The downstream cylinder then starts to shed a recirculating region of negative vorticity, and in doing so leeches vorticity from the negative recirculating region of the upstream cylinder. This results in the upstream positive recirculating region growing, and subsequently being shed.

At later times, vortex shedding occurs regularly from both cylinders. Vortices which are shed from the upstream cylinder impinge on the downstream cylinder, which sheds vortices which are a combination of its own vorticity and that of the upstream cylinder.

The flow structure is in good qualitative agreement with other studies of cylinders in tandem [11] [22].

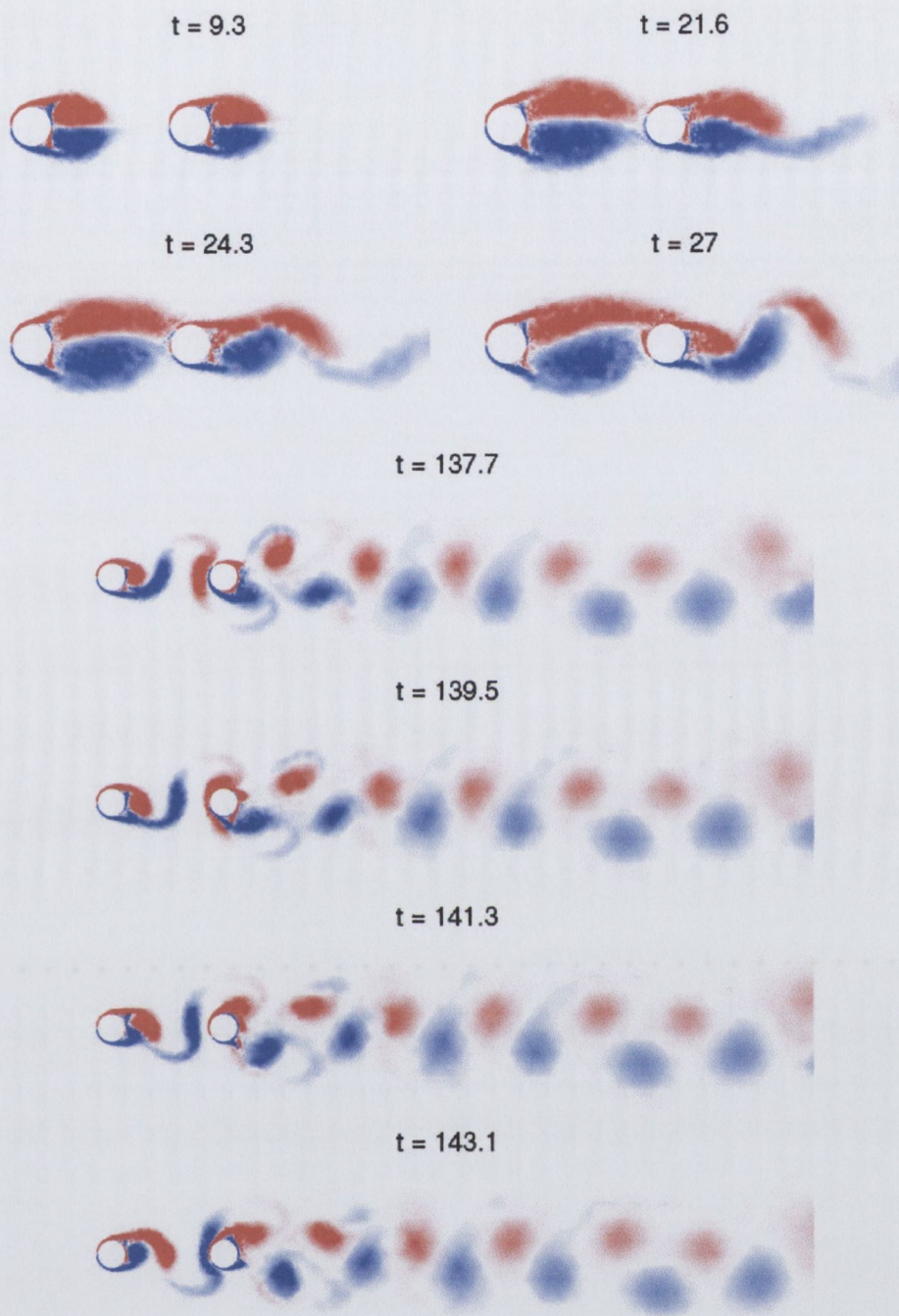


Figure 7.21: Vorticity field for flow over two circular cylinders, $Re = 550$.

Chapter 8

Conclusions

8.1 Cloud-in-element

The present work allows an unstructured mesh to be used with the random vortex method, offering a mixed Lagrangian-Eulerian simulation of viscous flow over multiple bodies of arbitrary geometry.

The velocity field due to the point vortices present is calculated by using a finite element method to solve the Poisson equation on a single unstructured mesh. This requires that the circulation of the point vortices be assigned to the unstructured mesh. The first step, that of locating the point vortices in the mesh, is carried out efficiently by a novel technique which makes use of reference matrices. The circulation may then be spread onto the mesh using area coordinates.

Resolution of the velocity field depends on the size of the elements in the unstructured mesh. The mesh density may be increased in areas of interest, such as near the boundary and in the near wake, to better capture the velocity field, while larger element sizes are sufficient for areas where complex flow patterns do not occur.

Numerical stability and accuracy were examined by varying the time step and mesh density. The simulations remain stable for all parameters, showing better

agreement with other results as the time step became smaller and mesh density became finer. The process time of the simulation was found to increase *linearly* with the number of vortices present in the calculation, rather than $O(N^2)$, or $O(N\log N)$.

8.2 Results obtained

The cloud-in-element method was applied to the standard case of impulsively started flow over a circular cylinder, for a range of Reynolds numbers from 40 – 9500. Streamline plots, vorticity field plots, body circulation and drag coefficient were recorded and compared to other computational [13], and experimental [29] results. Agreement was found to be good across a large range of Reynolds numbers.

At low Reynolds numbers, two large recirculating regions developed behind the cylinder, giving a steady drag with a slight sinusoidal oscillation in the wake after long time runs. As the Reynolds number increased, secondary and tertiary recirculating regions appeared, resulting in more complex behaviour of the body circulation and vorticity field. The Strouhal number increased with Reynolds number until reaching a plateau around 0.22, showing good agreement with experimental results. The values of mean drag showed an initial decrease with increasing Reynolds number before leveling off, in good agreement with experimental results. At higher Reynolds numbers, the mean drag increased again but was always within an order of magnitude of the experimental results.

The use of an unstructured mesh allows this method to be easily applied to multiple bodies and arbitrary geometries. As an example, a simulation was run for the flow over two longitudinally spaced cylinders, and the vorticity field presented at different stages of the flow.

It is concluded that:

- The random vortex method has successfully been extended to use on a single unstructured mesh
- Points can efficiently be located within an unstructured mesh using reference matrices
- The method remains numerically stable, with increasing accuracy as time step and mesh spacing are decreased
- The method offers results which are in good qualitative and quantitative agreement with other computational and experimental results
- The new technique, referred to as the cloud-in-element method, offers an efficient way of simulating two-dimensional bluff body flow, $O(N)$, and can easily be applied to multiple bodies and arbitrary geometries

8.3 Future work

Further study into the behaviour of the simulation with different parameters may offer a guideline on the requirements on time and space discretisation for different Reynolds numbers.

The present work used first-order elements in the unstructured mesh, resulting in the calculated velocity being constant across each element. The stream function in this case is C_0 continuous. Higher-order elements could allow a C_1 continuity, and the calculated velocity would be able to vary across each element face. Therefore, less elements may be required in a higher-order mesh than would be with first-order elements. However, higher-order elements require more nodes per element in the unstructured mesh, and the efficiency gained by using less elements would have to be judged against the efficiency lost by requiring more nodes per element.

Aerodynamic sound can be considered to be generated as a result of the movement of vortices in an unsteady fluid flow [17]. As point vortices form the basis of the present flow simulation, it would be interesting to investigate how the current method could be extended to offer predictions of the acoustic field generated by flow over multiple bodies.

The random vortex method offers a way of solving for viscous diffusion which is easy to implement, but it does require the vorticity transport equation to be split into two terms which are solved separately at each time step. The method of particle strength exchange allows viscous diffusion to be solved at the same time as solving for the velocity field. The only process remaining for each time step would then be the convection of the point vortices. Using a method like particle strength exchange would allow a clearer study of the dependency of the simulation on time step and mesh spacing, which may result in higher resolution simulations across a range of Reynolds numbers for multiple bodies and arbitrary geometries.

Vortex methods offer a Lagrangian approach to the solution of the governing equations. No separation into mean and fluctuating components has been done, and so a full solution to the flow is being sought. The present work allows vortex methods to be applied to multiple bodies and arbitrary geometries, however if flow at high Reynolds numbers is to be considered, three-dimensional effects must be taken into account. If the present method could be extended to three dimensions, then scales which are not caught by the discretisation of time and space used in the simulation could be modelled for the effects of turbulence, similar to the sub-grid scale modelling done in LES. This may then offer a powerful method of CFD analysis, combining Lagrangian and Eulerian methods, direct solutions and sub-grid modelling.

Bibliography

- [1] *Using MATLAB*. The MathWorks, Inc., 1998.
- [2] G. K. Batchelor. *An introduction to fluid dynamics*. Cambridge University Press, 1967.
- [3] G.D. Birkhoff and J. Fisher. Do vortex sheets roll up? *Rendiconti del Circolo Matematico di Palermo*, 2(8):77–90, 1959.
- [4] R. Bouard and M. Coutanceau. The early stage of development of the wake behind an impulsively started cylinder for $40 < Re < 10^4$. *Journal of Fluid Mechanics*, 101(3):583–615, 1980.
- [5] A. J. Chorin. Numerical study of slightly viscous flow. *Journal of Fluid Mechanics*, 57(4):785–796, 1973.
- [6] A. J. Chorin. Vortex sheet approximation of boundary layers. *Journal of Computational Physics*, 27:428–442, 1978.
- [7] A. J. Chorin and P. S. Bernard. Discretization of a vortex sheet with an example of roll-up. *Journal of Computational Physics*, 13:423–429, 1973.
- [8] J. P. Christiansen. Numerical simulation of hydromechanics by the method of point vortices. *Journal of Computational Physics*, 13:363–379, 1973.
- [9] G-H Cottet and P. D. Koumoutsakos. *Vortex methods: theory and practice*. Cambridge University Press, 2000.

- [10] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73:325–348, 1987.
- [11] C. L. R. Siqueira J. R. Meneghini, F. Saltara and J. A. Ferrari Jr. Numerical simulation of flow interference between two circular cylinders in tandem and side-by-side arrangements. *Journal of Fluids and Structures*, 15:327–350, 2001.
- [12] O. D. Kellogg. *Foundations of potential theory*. New York: Ungar, 1929.
- [13] P. Koumoutsakos and A. Leonard. High-resolution simulations of the flow around an impulsively started cylinder using vortex methods. *Journal of Fluid Mechanics*, 296:1–38, 1995.
- [14] H. Lamb. *Hydrodynamics*. Cambridge University Press, sixth edition, 1993.
- [15] F. Milinazzo and P. G. Saffman. The calculation of large reynolds number two-dimensional flow using discrete vortices with random walk. *Journal of Computational Physics*, 23:380–392, 1977.
- [16] D. W. Moore. A numerical study of the roll-up of a finite vortex sheet. *Journal of Fluid Mechanics*, 63(2):225–235, 1974.
- [17] A. Powell. Theory of vortex sound. *Journal of the Acoustical Society of America*, 36(1):177–195, 1964.
- [18] L. Rosenhead. The formation of vortices from a surface of discontinuity. *Proceedings of the Royal Society. Series A: Mathematical & Physical Sciences*, 134:170–192, 1931.
- [19] W. D. Gropp D. Kaushik L. Curfman McInnes S. Balay, K. Buschelman and B. F. Smith. Petsc home page. <http://www.mcs.anl.gov/petsc>, 2001.
- [20] J. A. Sethian. *Vortex methods and vortex motion*, chapter 1. A brief overview of vortex methods, pages 1–32. Philadelphia SIAM, 1991.

- [21] I. H. Shames. *Mechanics of fluids*. McGraw-Hill, international student edition, 1982.
- [22] A. Slaouti and P. K. Stansby. Flow around two circular cylinders by the random-vortex method. *Journal of Fluids and Structures*, 6:641–670, 1992.
- [23] P. A. Smith and P. K. Stansby. Impulsively started flow around a circular cylinder by the vortex method. *J. Fluid Mech.*, 194:45–77, 1988.
- [24] P. A. Smith and P. K. Stansby. An efficient surface algorithm for random-particle simulation of vorticity and heat transport. *Journal of Computational Physics*, 81:349–371, 1989.
- [25] P. A. Smith and P. K. Stansby. Viscous oscillatory flows around cylindrical bodies at low keulegan-carpenter numbers using the vortex method. *Journal of Fluids and Structures*, 5:339–361, 1991.
- [26] P. K. Stansby and A. G. Dixon. Simulations of flows around cylinders by a Lagrangian vortex scheme. *Applied Ocean Research*, 5(3):167–178, 1983.
- [27] P. K. Stansby and A. Slaouti. Simulation of vortex shedding including blockage by the random-vortex and other methods. *International Journal for Numerical Methods in Fluids*, 17:1003–1013, 1993.
- [28] N. Wax. *Selected papers on noise and stochastic processes*. Dover, 1954.
- [29] M. M. Zdravkovich. *Flow around circular cylinders*, volume 1. Oxford University Press, 1997.
- [30] O. C. Zienkiewicz and R. L. Taylor. *The finite element method*, volume 1: Basic formulation and linear problems. McGraw-Hill Book Company Europe, 4th edition, 1989.