Development of a Fast Vortex Method for Fluid Flow Simulation using Special-Purpose Computers

Tarun Kumar Sheel

School of Science for Open and Environmental Systems

Graduate School of Science and Technology

Keio University, Japan

A thesis submitted for the degree of

Ph.D. in Engineering

March 2008

I would like to dedicate this thesis to my family

Acknowledgements

It is my great pleasure to acknowledge my teachers, colleagues, and friends in my lab and mechanical engineering department of Keio University for their contribution to this work and their friendship over the past years.

The time I have spent at Keio was certainly fruitful and enjoyable. Keio is not only renowened for its scientific research but also for its other curriculum activities in Japan as well as in abroad.

I would like to express my heartiest thank to my supervisor Professor Shinnosuke Obi for his excellent guidence in bringing this work to its present form through the years. He has always been available for discussion, consultations and inspirations.

I am grately appreciate to Prof. Shigeki Masuda for his continuous advise throughout his time.

I am much obliged to the members of my examining committee: Prof. Tatsuo Sawada, Prof. Hideharu Amano and Prof. Kenji Yasuoka. My special gratitude to Prof. Sawada and Prof. Amano to review my work and make valuable comments and suggestions on it. Their inspirations encouraged me to prduce a good dissertation.

I would like to give my sincere thanks to Prof. Kenji Yasuoka for his fruitful suggestions during the period and provided the computer facilities. It would not be possible to finish this work without the computational resources from his lab. I had an open access to his computer lab for computation. He always encouraged me to progress this work in present form.

I am really grateful to Dr. Tetsu Narumi for his suggestions and fruitful discussions to prepare the figures of MDGRAPE-2 and MDGRAPE-3. He provided me some technical information to use them.

My heartiest thanks to Dr. Koji Fukagata for his sincere suggestion in the final year of my program.

The special thanks to Mr. Rio Yokota who has been helped me from the first day in Japan and continuous discussion regarding the research. He provided me some vortex codes to progress my computation. I am always grateful for his endless support and fruitful suggestions whole the period.

I thank Mr. Shan Kameoka for his technical support to run the MDGRAPE machines uninterruptedly.

I appreciate the administrative assistance given to me by all of our personnel at International Center and the Mechanical Engineering Department of Faculty of Science and Technology.

This effort supported by the Yoshida Scholarship Foundation and Amano Scholarship Foundation, Japan. It was also supported by KLL doctor students grant from Keio University. Partial research fund was received from the students affairs office of Faculty of Science and Technology. These contributions are gratefully acknowledged.

Abstract

A fast vortex method has been developed by using special-purpose computers, MDGRAPEs, those were exclusively designed for molecular dynamics simulations. This is an attempt to make a bridge between vortex method and molecular dynamics calculations. The three main issues have been solved regarding the implementation of the MDGRAPE on vortex methods those are the efficient calculation of the Biot-Savart and stretching equation, the optimization of the table domain, and the round-off error caused by the partially single precision calculation in the MDGRAPE.

A mathematical formulation for the 3D vortex method has been developed for calculation using a special-purpose computer MDGRAPE-2. A rigorous assessment of this hardware has been made for a few representative problems and compared the results with and without it. It is found that the generation of appropriate function tables, which are used to call libraries, embedded in MDGRAPE-2 is of primary importance in order to retain accuracy. The error arising from the approximation is evaluated by calculating three pairs of vortex rings impinging to themselves. Consequently, acceleration of about 100 times is achieved by MDGRAPE-2 while the error in the statistical quantities such as kinetic energy and enstrophy remain negligible.

MDGRAPE-3, successor of MDGRAPE-2, has been applied to the same calculations and the improvement in speed was 1000 times faster when compared with the host PC and 25 times compared with MDGRAPE-2 for $N = 10^6$. Some issues regarding the comparative study between MDGRAPE-2 and MDGRAPE-3 have been investigated carefully.

The simultaneous use of the fast multipole method (FMM) with MDGRAPE-2 and MDGRAPE-3 has been successfully applied to the same calculations to

investigate the possibility of further accelerations. The various forms of FMM and their performance on MDGRAPE-2 and MDGRAPE-3 have been investigated. With the help of these acceleration techniques the dynamics of two colliding vortex rings have been studied and the computation time has been reduced by a factor of 2000 compared to a direct calculation on a standard PC. The global kinetic energy and enstrophy have been investigated to address the numerical accuracy. The results have good agreement when compared with the previous and referenced work. The reconnection of the vortex rings was clearly observed, and the discretization error became nearly negligible for the calculation using 10^7 elements.

Nomenclature

Roman Symbols

u	Velocity
u	Velocity component in i-direction
v	Velocity component in j-direction
w	Velocity component in k-direction
V	Volume
x, x′	Position of vortex particles
t	Time
d	Dimension
r	Distance between position vector
p	Order of multipole moment
N	Number of particles
O	Order of calculation cost
G	Green's Function
Y	Spherical harmonics
S	Distance between two rings
R	Radius of vortex ring
r	Radius of cross-section
L	Level of box division
g	Arbitrary function of a function table
f_i	Pairwise force
A, a	Scaling factor
B, b	Scaling factor
K, E	Kinetic Energy
k	Wave number space

D	Material derivative

Greek Symbols

ω	Vorticity
γ	Vortex strength
Γ	Circulation
∇	Vector operator
Δ	Differential operator
δ	Partial differential operator, Difference operator
ν	Kinemetic viscosity
$\pi \approx$	3.1415926...
σ	Core radius
ζ	Cutoff Function, Enstrophy
Ω	Enstrophy
α	Spherical coordinate
β	Spherical coordinate
θ	Spherical coordinate
ϕ	Spherical coordinate
ρ	Spherical coordinate
Φ_i	Potential
ϵ	Softening parameter
η	Efficiency

Other Symbols

erf	Error function
exp	Exponential Function

Acronyms

VM	Vortex Method
FMM	Fast Multipole Method
MD	Molecular Dynamics
MDG2	MDGRAPE-2
MDG3	MDGRAPE-3
PCI	Peripheral Component Interconnect
API	Application Programming Interface

FPGA	Field Programmable Gate Array
LSI	Large-scale Integrated
SIMD	Single Instruction Multipole Data
GPU	Graphics Processing Units
FLOPS	Floating Point Operations
GRAPE	GRAvity PipE
MD-GRAPE	Molecular Dynamics GRAvity PipE
ASIC	Application-Specific Integrated Circuit
M2M	Multipole to Multipole expansion
M2L	Multipole to Local translation
L2L	Local to Local expansion
P^2M^2	Pseudo-Particle Multipole Method
stx	Stretching term part 1
tx	Stretching term part 2
FE	Function Evaluator
VIC	Vortx-in-Cell Method
MLS	Moving Least Squares
RVM	Random Vortex Method
CSM	Core Spreading Method
PSE	Particle Strength Exchange
VRM	Vortex Redistribution Method
SPH	Smooth Particle Hydrodynamics
MPS	Moving Particle Semi-implicit Methods

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

N-Body simulations were devised in the 1950s and have been widely used since the 1970s when digital computers became powerful enough and affordable. Today it is considered to be an orthodox method for studying particle systems. The classical N-body problem simulates the evolution of a system of N bodies, where the force exerted on each body arises due to its interaction with all other bodies in the system. N-Body algorithms have numerous applications in areas such as astrophysics, molecular dynamics, plasma physics and computational fluid dynamics using the vortex method. For each of these computational problems the calculation takes on a slightly different form but each share common features.

The vortex methods have been developed and applied for analysis of complicated, unsteady and vortical flows related with problems in a wide range of industries, because they consist of simple algorithm based on physics of flow (Anderson and Greengard [1986]; Koumoutsakos and Leonard [1995]; Shankar [1996a]; Mansfield et al. [1999]; Cottet et al. [2000]; Barba et al. [2004]). Leonard [1980] summarized the basic algorithm and example of its applications.

The emergence of the vortex method can be traced to the 1930s, with the Rosenhead [1931] calculations of the Kelvin-Helmohltz instabilities. The use of the fractional-step, Lagrangian vortex scheme for incompressible viscous flow was originally proposed by Chorin [1973]. The scheme was named random walk

method and later, Chorin [1978] modified his method for wall boundary conditions. The method has been proved to converge for unbounded domains by Beale and Majada [1981, 1982].

During the following decades, different vortex methods were proposed to solve the Navier-Stokes equations. There are several detailed reviews vortex methods in incompressible flow (Leonard [1980]; Anderson and Greengard [1986]). The development and applications of vortex methods were described by Cottet et al. [2000].

The vortex methods have made remarkable advancements in the past decade, but still face numerous challenges, especially involving viscous diffusion schemes and the high computation cost.

The evolution of viscous vortex methods in the last two decades mainly includes the particle strength exchange(PSE) method developed by Degond and Mas-Gallic [1989], core-spreading method of Kuwahara and Takami [1973]; Leonard [1980], a deterministic particle method proposed by Fishelov [1990] and several hybrid vortex methods thereafter.

The difficulty of calculating diffusion in Lagrangian methods stands out when I look back at the history of Vortex Methods. Early studies were based on inviscid flows, e.g. the filament methods and panel methods. The theorems of Helmholtz and Kelvin state that for an inviscid fluid, vortex lines move as material fluid elements, and the Lagrangian treatment of vortex filaments is a straightforward solution of the vorticity equation without diffusion. An excellent review by Leonard [1985] gives a detailed description of such methods. Another approach for inviscid flows is the solution of the potential equations. Panel methods solve the potential equations using the boundary integral approach, and have been applied to numerous aerodynamic problems. For extremely high Reynolds number flows the viscosity is confined to a very thin region near the solid boundaries, thus justifies the use of such idealized equations to some extent. The review by Hess [1990] summarizes nicely the most important achievements in this field.

The vortex method solves time-dependent incompressible flow problems by discretizing the vorticity into vortex elements and following these elements in time. This results in a volume mesh-free algorithm and saves significant time in preprocessing when compared to the conventional Navier-Stokes approach where

grids need to be generated. The significant advantages of vortex methods is of its grid free nature and the calculation takes only the region of entire domain where vorticity exist. It has an advantage in solving high Reynolds number flows with complex geometries. The computational elements are automatically concentrated and thus no need to distribute elements in critical regions compared with grid-based methods. In the computation of the velocity field from the vorticity, far-field boundary conditions are satisfied explicitly (Stock [2007]).

One of the main difficulties with vortex methods is that the cost of the evaluation of the velocity field induced by N vortices is of $O(N^2)$. In these calculations, the largest computational load occurs in the routine that calculates the Biot-Savart law and the stretching term in the vorticity equation. This is expensive, particularly in three dimensions where a large number of elements are computed simultaneously, and calculation load becomes highly expensive. The large number of elements are required to accurate calculation for high Reynolds number flows which consumed very high computation cost.

1.2 The need for acceleration techniques

There has always been a strong relationship between progress in vortex methods and advancements in acceleration techniques that utilize this method. When the classical vortex methods became popular nearly 30 years ago, the calculation cost of the N-body solver was $O(N^2)$ for N particles. Due to this enormous calculation cost, the intention at that time was not to fully resolve the high Reynolds number fluid flow, but to somewhat mimic the dominant vortex dynamics using discrete vortex elements.

The advance of fast algorithms made it possible to achieve a scaling of $O(N)$ (Warren and Salmon [1994]; Cheng et al. [1999]), and with the help of the rapid development in computational hardware, a calculation involving millions of vortex elements became possible (Ploumhans et al. [2002]). The application of fast N-body solvers to vortex methods enabled the calculation of millions of vortex elements (Salmon et al. [1994]). The fast algorithms were also applied to the boundary integral equations (Winckelmans et al. [1996]). Furthermore, fast N-body solvers were not only used for vortex particle methods, but also vortex tube

methods (Collins et al. [1999]). These efforts led to a new paradigm, i.e. solving flows of moderate Reynolds numbers and fully resolving these flows. The vortex method was recognized as a discretization method rather than an attempt to model vortex dynamics, because the computational power that is neccessary to prove these claims became available.

However, the high proportionality constant of the fast N-body solvers prevented them from matching the speed of fast Poisson solvers. Since the mainstream methods in computational fluid dynamics use fast Poisson solvers, it was still difficult for vortex methods to be considered as an alternative to conventional grid based methods.

The shortcomings of the fast N-body solvers can partly be circumvented by the use of hybrid methods, while the Lagrangian nature of the convection calculation is retained. The vortex-in-cell (VIC) is a typical hybrid method, and its accuracy and speed are quite close to that of the spectral method (Cottet et al. [2002]). The particle-mesh method is another hybrid approach(Hockney et al. [1973, 1981]). Sbalzarini et al. [2006] developed a particle-mesh library that calculates one vortex method iteration for 268 million particles took 85 s on 128 processors. This is comparable to the performance of the state-of-the-art finite difference methods using processors of comparable performance.

Based on the above discussions, significant acceleration techniques are necessary to reduce the computation cost of direct interaction calculation for millions of particles.

1.3 Motivation

In order to accelerate the $N-$body simulations, various methods have been explored from the point of view of both software and hardware.

With respect to software, Barnes and Hut [1986] developed the tree code for reducing cost of gravitational N-body problems. The tree code is an $O(N \log N)$ algorithm based on a hierarchical octree representation of space in three dimensions. It computes interactions between distant particles and reduces the number of operations by means of a first order approximation. Many existing implementations of tree code algorithms only use up to quadropole moments and calculation

costs rise quickly when high accuracy is required. Ploumhans et al. [2002] has been used treecode for vortex method calculation with parallel computers to reduce the excessive calculation cost. Greengard and Rokhlin [1987] invented the fast multipole method (FMM) which reduced the calculation cost of particle simulations. The FMM is an approximate algorithm of which the calculation cost scales proportional to $O(N)$. In the FMM, the long-range forces are approximated by multipole expansion truncated at a certain degree, while the contributions from particles within nearby regions are calculated directly in a usual manner without approximation. Including higher order terms in multipole approximations and/or increasing the size of a nearby region can improve the computational accuracy. However, either effort substantially increases the computation time. In particular, the computation of a high-order term is very expensive.

With respect to hardware, the use of both vector and parallel computers have been investigated. Susukita et al. [2003] has been developed a hardware accelerator MDGRAPE-2 to execute the N-body simulation. MDGRAPE-2, the GRAPE (Gravity Pipe, developed by Sugimoto et al. [1990]) series machine, is a special-purpose computer exclusively dedicated for molecular dynamics calculations between point-charge or point-mass particles. MDGRAPE-3 was developed as a successor of MDGRAPE-2 with the speed up 12.5 times higher than that of MDGRAPE-2 (Taiji et al. [2003]). Both of the hardwares can be used for vortex method calculations as of its mathematical architectures are similar as molecular dynamics calculations. Its performance are much higher than that of ordinary computers. Its can speed up force calculations about 10-1000 times when compared to general-purpose (defined as 'host' or 'direct' hereafter) computers of the same cost.

Parallel computation is another hardware technique rather than special-purpose hardware. There are several articles have been published based on the parallel computers to accelerate vortex method calculations. Warren and Salmon [1994] have been developed a fast tree code using 1024 parallel processors for many body problems and the calculation time has been reduced significantly. The authors also mentioned the three difficulties to use parallel computers for large N calculation. In later year, Warren and Salmon [1995] have been developed a portable particle program by using high performance cluster computers and applied to

vortex particle methods to reduce the calculation cost. Winckelmans et al. [1996] has been calculated 3D viscous flows without boundaries on parallel computers and computation cost has been reduced from $O(N^2)$ to $O(NlogN)$. Ploumhans et al. [2002] has been used HP V-class 64 CPU and accelerated 3D vortex method calculation for a million of particles. A massively parallel single instruction multipole data stream (SIMD) processor has been desigend and applied to the RC4 keysearch problem and able to achieve a high level of parallelism as well as utilize the higher memory bandwidth available on the device (Stanley et al. [2003]). This approach can be applied to other applications to accelerate the parallel based computations.

Considered the above-mentioned discussions regarding the different hardware acceleration techniques, my approach is to use special-purpose computers in favour of following reasons.

It is important to note that fast Poisson solvers cannot be processed on such special hardware, and only N-body solvers enjoy the benefit of these special-purpose computers. Liu et al. [2007] used the Graphics Processing Unit(GPU) to speed up the scientific calculations of tornado. Recently Stock and Gharakhani [2008] have been used general-purpose graphics processing unit(GPGPU) for parallel vortex particle methods and speedup the calculation significantly. The referenced paper reported the speedup nearly 1000 times over the direct method for practical problems ($N = 10^6$) on an AMD Opteron 246 processor with version 5.2 of the pgf90 compiler. Further details of GPGPU and its related software can be found in Bedorf [2007]; Belleman et al. [2008]; Elsen et al. [2007]. Ever since the GRAPE (Sugimoto et al. [1990]) was first introduced, these special-purpose computers have constantly outperformed the leading general purpose computers of the same price (Makino and Taiji [1998]; Narumi et al. [2006]). The special-purpose computers can also be used for the boundary integral calculation (Takahashi et al. [2006]). At this point, it is not yet evident which will prevail; fast Poisson solvers on parallel general purpose architecture, or fast N-body solvers on parallel special-purpose processors.

The long computation time due to the above-mentioned $O(N^2)$ problem may be reduced when applied to a special-purpose computer. Nevertheless, the vortex methods calculations have the same mathematical architecture as a multibody

problem, thus permitting the use of special-purpose computers of multibody problems.

The present research aims to solve the 3D complex flows by investigating the high Reynolds number effects and the error is being isolated from the boundary effect(Orszag and Patterson [1972]; Rogallo [1981]). Vortex ring is an important element in turbulent flows, and the studies of their interaction may be applicable to the complex fluid flows. The study of vortex rings collisions, both with a solid boundary and with other rings, have provided a plenty of information about vortex dynamics which is one of the most fundamental means of understanding fluid motion, especially at high Reynolds number flows.

The collision of vortex rings contain millions of particles has been chosen as a test case for the present calculations. The following characteristics of this flow allow to focus on the assessment of the proposed acceleration technique. The flow does not involve solid or periodic boundaries, thus causes minimum complication in the implementation of the FMM itself. Also, the initial condition is simple to generate using vortex methods. Furthermore, although the initial flow field is quite simple, the collision of the rings results in a highly turbulent state, and is greatly affected by the Reynolds number of the initial situation. This allows me to demonstrate the ability to handle high Reynolds number flows by using a large number of particles, which becomes possible with the use of the proposed acceleration method.

1.4 Previous Studies

Vortex dominated flows are often very complex such as free jets (Liu et al. [2000]). The jet development is characterized by the dynamics of large scale vortex rings. In past decade, vortex rings have been studied in the broader arena of vortex interactions (Shariff and Leonard [1992]). The strong effects have been devoted to study of the interaction of vortex rings in various configurations.

It is evident from previous studies, a large number of particles is necessary to capture the essential characteristics of the vortex ring collision. Winckelmans and Leonard [1993] performed a vortex method calculation of the impingement of two identical inclined vortex rings, and investigated the physical properties at

$Re = 400$. Authors suggested that a large number of particles is necessary wherever stretching of vortex is intense. Mammetti et al. [1999] studied the collision of vortex rings with a solid boundary using vortex methods and suggested that for low $Re < 350$ the vorticity is too weak to generate a secondary vortex and for higher Re it generates a secondary vortex ring. Chatelain et al. [2003] calculated the reconnection of two offset vortex rings at $Re = 250$ and suggested that higher Re is necessary to produce the fast mechanism of energy transfer. Authors used tree code to accelerate the calculation and observed that the computational resources are important for large numbers calculation for entire time. Cottet et al. [2002] has used the Vortex-in-Cell(VIC) method for acceleration and found that the cost is still larger than a grid-based poisson solver. Even though high computation cost, the vortex method would become advantages for high Reynolds number external flows, where the vorticity concentrated to a finite region.

It can be clearly observed from previous studies that the simultaneous use of fast algorithms with fast computers led to a new paradigm to accelerate the calculation of multibody problems retained the accuracy in an acceptable level. The first implementation of fast algorithms on GRAPE architecture was presented by Makino [1991], and showed a 30-50 times increase in computational speed compared to the treecode without GRAPE for the simulation of astrophysical problems. The implementation of the FMM on MDGRAPE-2 was presented by Chau et al. [2002a,b] and similar results were obtained. The application of P^2M^2 tree code on MDGRAPE-2 presented by Kawai et al. [2004] and accelerates the calculation by a factor of 20-200 compared with conventional PCs. Yatsuyanagi et al. [2003a] has used MDGRAPE-2 to accelerate the velocity calculation of Biot-Savart integral equation for the simulation of 2D magnetohydrodynamics problems using the current vortex method.

1.5 Purpose of the present study

The main focus of the present study is to accelerate the vortex method calcaultions for high Reynolds number flows without the loss of numerical accuracy. This is an attempt to make an interface between vortex method and molecular dynamics calculations. To be succeeded, the special-purpose computers

MDGRAPE-2 and MDGRAPE-3 have been used for entire calculations. The three main issues have been solved regarding the implementation of the MDGRAPE on vortex methods are the efficient calculation of the Biot-Savart and stretching equation, the optimization of the table domain, and the round-off error caused by the partially single precision calculation in the MDGRAPE. Furthermore, the FMM will be implemented on MDGRAPE-2 and MDGRAPE-3 for further acceleration of the present calculations. The simultaneous use of the above two methods is possible since the MDGRAPE still has a scaling of $O(N^2)$ and the FMM can be used to reduce it.

In this study, a number of techniques to make the vortex methods calculation manageable are implemented.

Firstly, a mathematical formulation for the 3D vortex method has been developed for calculation using a special-purpose computer MDGRAPE-2 that was originally designed for molecular dynamics simulations. A rigorous assessment of this hardware has been made for a few representative problems and compared the results with and without it. It is found that the generation of appropriate function tables, which are used to call libraries, embedded in MDGRAPE-2 is of primary importance in order to retain accuracy. The cross product calculation which is not considered in the original command set must be handled in a proper manner, which is treated in some previous works, e.g., Yatsuyanagi et al. [2003a,b] and Elmegreen et al. [2002, 2004]. Consequently, acceleration about 50 times is achieved by MDGRAPE-2 while the error in the statistical quantities such as kinetic energy and enstrophy remain negligible(Sheel et al. [2007]).

Secondly, MDGRAPE-3, successor and 12.5 times faster than that of MDGRAPE-2, has been applied to the same calculations and the improvement in speed was 1000 times faster when compaerd with the host PC and 25 times compared with MDGRAPE-2 for $N = 10^6$. The only difference between the MDGRAPE-2 and MDGRAPE-3 is that the latter can simultaneously calculate along with the host machine, but can only handle a small number of source particles at once (Narumi et al. [2006]). Some issues regarding the comparative study between MDGRAPE-2 and MDGRAPE-3 will be investigated carefully.

Thirdly, the simultaneous use of the FMM and MDGRAPE-2 and -3 has been implemented to investigate the posiibility of further acceleration without

the loss of numerical accuracy. The effect of the order of multipole momentums have been investigated for FMM accuracy. Optimum level of box divisions has been determined for MDGRAPE calculation. The performance and accuracy have been investigated by calculating the collision of two identical inclined vortex rings. The time history of kinetic energy and enstrophy have been compared with the host calculation. The reconnection of the vortex rings was clearly observed, and the discretization error became nearly negligible for the calculation using 10^7 elements(Sheel et al. [2008]).

The thesis is organised as follows. In chapter 2, basic numerical methods are discussed, acceleration techniques are discussed in chapter 3, in chapter 4 a fast vortex method is developed by using MDGRAPE-2, the simultaneous use of the FMM with MDGRAPE-2 and MDGRAPE-3 are explained in chapter 5 and finally the concluding remarks are in chapter 6.

Chapter 2

Numerical Methods

2.1 Introduction

In this chapter I will briefly describe the basic elements of the vortex methods and the other fast methods have been used in the present calculations. Further details and applications of those methods can be found in Anderson and Greengard [1986]; Barnes and Hut [1986]; Barnes [1990]; Greengard and Rokhlin [1987], Leonard [1980, 1985]. The mathematical formulations can be found in Anderson and Greengard [1986]; Cheng et al. [1999]; Chorin [1993], Hernquist [1987]; Hernquist and Katz [1989], Puckett [1993], Raviart [1985], and Schmidt et al. [1991].

The main difficulty for vortex methods to be accepted in the mainstream of computational fluid dynamics are; *(a)* the numerical complexity of calculating the velocity using the Biot-Savart law, which is in fact analogous to an "N-body problem" and hence requires $O(N^2)$ operations for N vortex elements, *(b)* diffusion error; and *(c)* the loss of discretization accuracy due to the distortion of the particle distribution. The details reviewed can be found in Barba et al. [2004].

First I will discuss the vortex methods and its formulation for a fast vortex method, then viscous diffusion schemes will be discussed and finally the other fast methods will be discussed in subsequent sections.

2.2 Vortex Methods

Vortex methods are part of a wider class of methods: the Lagrangian methods used to simulate unsteady, convection-dominated, problems. Those are expressed by transport equations written in conservative form, often with a diffusion term, and eventually with a source/depletion term (Shankar [1996a]).

Despite its potential to become a totally grid-free and very accurate fluid dynamics solver, vortex methods suffer the reputation of being some obscure "model" to mimic vortex dynamics. This is certainly not the case since the governing equations are discretized without any modeling whatsoever. The incompressible vortex method solves the Poisson equation for velocity along with the vorticity equation, just like the Poisson equation for pressure and the Navier-Stokes equation are coupled in the primitive variable formulation. It is true that Lagrangian methods have a difficulty in achieving higher order spatial accuracy compared to Eulerian methods, but this is a relative matter and does not mean that they are inconsistent. It is strongly believe that, given a sufficient amount of consideration for the diffusion schemes and validating these schemes by progressively increasing the complexity of the flow field starting from the most simple ones, the vortex method will become an alternative method of computational fluid dynamics, especially advantageous for complex external flows.

2.2.1 Formulation of 3D Vortex Element Method (VEM)

Vortex element method have been growing in popularity in last three decades. As their name indicates, they are based on the discretization of vorticity-a quantity that has a compact support in many physical problems-thereby making this approach interesting (Chatelain et al. [2005]).

The three-dimensional incompressible flow of a viscous fluid has been studied here. The evolution equation for vorticity is

$$\frac{\mathrm{D}\boldsymbol{\omega}_i}{\mathrm{D}t} = (\boldsymbol{\omega}_i \cdot \nabla)\,\mathbf{u} + \nu\nabla^{\mathbf{2}}\boldsymbol{\omega}_{\mathbf{i}} \tag{2.1}$$

where $\boldsymbol{\omega}$ is the vorticity defined as $\boldsymbol{\omega} = \nabla \times \mathbf{u}$, $\mathbf{u}$ is the velocity of the vortex element, $(\boldsymbol{\omega} \cdot \nabla)\mathbf{u}$ is called the stretching term and represents the rate of change

of vorticity by deformation of vortex lines, ν is the kinematic viscosity, and the term $\nu\nabla^2\boldsymbol{\omega}$ represents the change of vorticity by viscous diffusion. The velocity field in a three-dimensional problem is,

$$\mathbf{u}(\mathbf{x}) = -\frac{1}{4\pi}\int \frac{(\mathbf{x}-\mathbf{x}')\times\boldsymbol{\omega}(\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}dV(\mathbf{x}') \tag{2.2}$$

where $\mathbf{x}$, and $\mathbf{x}'$ are the positions of vortex elements and dV is the volume of the element. Using the Winckelmans and Leonard [1993] model as a cut-off function (Eq. 2.10), the Biot-Savart law is formulated as follows

$$\mathbf{u}_i = -\frac{1}{4\pi}\sum_{j=1}^{N}\frac{\mathbf{r}_{ij}^2+(5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\mathbf{r}_{ij}\times\gamma_j \tag{2.3}$$

where $\mathbf{r}_{ij}=\mathbf{r}_i-\mathbf{r}_j$, σ_j and $\boldsymbol{\gamma}_j$ are the distances of the position vector, core radius and strength of element. The subscript i stands for the target elements, while j stands for the source elements. The stretching term of Eq. (2.1) can be discretized as follows:

$$\frac{d\boldsymbol{\omega}_i}{dt} = (\boldsymbol{\omega}_i\cdot\nabla)\,\mathbf{u} \tag{2.4}$$

If I put vortex strenght $\gamma_i = \boldsymbol{\omega}_i d^3\mathbf{x_i}$ in Eq. (2.4), then it becomes

$$\frac{d\gamma_i}{dt} = (\gamma_i\cdot\nabla)\,u_i \tag{2.5}$$

Hence, the vortex strength of an individual element is expressed by Eq. (2.3) in a discretized formulation as

$$\begin{aligned}\frac{d\gamma_i}{dt} \;=\; & \frac{1}{4\pi}\sum_{j=1}\left\{-\frac{|\mathbf{r}_{ij}|^2+(5/2)\sigma_j^2}{\left(|\mathbf{r}_{ij}|^2+\sigma_j^2\right)^{5/2}}\gamma_i\times\gamma_j\right.\\ & \left.+3\frac{|\mathbf{r}_{ij}|^2+(7/2)\sigma_j^2}{\left(|\mathbf{r}_{ij}|^2+\sigma_j^2\right)^{7/2}}(\gamma_i\cdot\mathbf{r}_{ij})(\mathbf{r}_{ij}\times\gamma_j)\right\}\end{aligned} \tag{2.6}$$

where all notations carry the same meaning as in Eq. (2.3). For details mathematical formualtions, referred Appendix A.

2.2.2 Viscous Diffusion

Vortex methods originate in inviscid methods and recent studies mostly focus on their extension to viscous flows. Though, this has not been a straightforward task and the diversity of methods has become quite large. The random vortex method (RVM) by Chorin [1973] uses a stochastic interpretation of the diffusion equation. It has served an important role in the early development of viscous diffusion schemes, but its slow convergence rate prompted the development of alternative methods. The core spreading method (CSM) by Kuwahara and Takami [1973] and Leonard [1980] uses a deterministic approach, which changes the standard deviation of the Gaussian distribution to match the fundamental solution of the diffusion equation. A straightforward implementation of this method lacks convergence due to the fact that the ever-expanding Gaussian distribution moves with the velocity at its center. Local spatial refinement (Rossi [1996]) can circumvent this problem, though this will introduce a large amount of error without careful consideration (Barba et al. [2005]; Barba [2006]; Huang [2005]).

The particle strength exchange (PSE) by Degond and Mas-Gallic [1989] redistributes the strength among vortex elements by solving the integral equation of the Laplacian operator. The location of elements are used as quadrature points, thus requires them to be nearly uniform for an accurate calculation(Chatelain et al. [2002]). The vortex redistribution method (VRM) by Shankar [1996b] also redistributes the strength of vortex elements but by solving an underdetermined system of equations to equate the truncated Taylor series of the new distribution with that of the exactly diffused vorticity. Although, restrictions of particle nonuniformity are not as severe as the PSE, it is obvious that a sufficient number must exist in the neighborhood. The insertion and merging of particles is still an open area of research, as is the case with CSMs.

In most cases a vortex element has three properties, circulation, core radius, and velocity. The CSM changes the core radius, PSE and RVM change the circulation to account for diffusion. The diffusion velocity method by Ogami and Akamatsu [1991] modifies the velocity instead, where the diffusion velocity becomes the product of $-\nu/\omega$ and the gradient of vorticity. For regions of zero vorticity the $-\nu/\omega$ becomes singular, so an algorithm which does not increase

the vorticity magnitude outside of the computational vorticity support (Grant and Marshall [2005]) is essential to this scheme. There exist many other ways to calculate the viscous diffusion of vorticity using a semi-Lagrangian discretization, such as the vortex in cell (VIC), free Lagrangian, triangulated, moving particle semi-implicit method(MPS)(Koshizuka et al. [1995]), and moving least squares (MLS). The present study focuses on pure Lagrangian schemes (with remeshing in some cases), thus semi-Lagrangian methods are out of scope.

In particular, I will focus only the scheme of core spreading method as follows.

The CSM is way to discretize the viscous diffusion equation

$$\frac{D\boldsymbol{\omega}_i}{Dt} = \nu\nabla^2\boldsymbol{\omega}_i, \tag{2.7}$$

where the Green's function solution is

$$\boldsymbol{\omega}_i = \frac{\boldsymbol{\gamma}_j}{(4\pi\nu t)^{d/2}} \exp\left(-\frac{|\mathbf{x}_j - \mathbf{x}_i|^2}{4\nu t}\right). \tag{2.8}$$

$\boldsymbol{\omega}$ is the vorticity, ν is the kinematic viscosity, $\boldsymbol{\gamma}$ is the circulation, $\mathbf{x}$ is the position vector, and d is the dimensionality of the problem. The subscript i stands for the target elements, while j stands for the source elements. The CSM uses a cutoff function ζ to discretize the diffusion equation. In this case the vorticity at an arbitrary point can be expressed as

$$\boldsymbol{\omega}_i = \sum_j \boldsymbol{\gamma}_j \zeta\left(|\mathbf{x}_j - \mathbf{x}_i|\right). \tag{2.9}$$

A common choice for the cutoff function is the Gaussian distribution

$$\zeta = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{|\mathbf{x}_j - \mathbf{x}_i|^2}{2\sigma^2}\right). \tag{2.10}$$

If I substitute (2.10) into (2.9), I can see that changing the variance of the Gaussian distribution according to

$$\sigma^2 = 2\nu t \tag{2.11}$$

will result in the heat kernel (2.8). σ is often referred to as the core radius of the vortex blob, and represents the physical length scale of the vortex elements.

Table 2.1: Examples of 3D cutoff functions for VM (Winckelmans [2004])

Function	g(ρ)	f(ρ)	$\zeta(\rho)$	r
Low-order algebraic	$\frac{\rho^3}{(\rho^2+1)^{3/2}}$	$\frac{1}{(\rho^2+1)^{1/2}}$	$\frac{3}{(\rho^2+1)^{5/2}}$	0
High-order algebraic	$\frac{\rho^3(\rho^2+5/2)}{(\rho^2+1)^{5/2}}$	$\frac{(\rho^2+3/2)}{(\rho^2+1)^{3/2}}$	$\frac{15/2}{(\rho^2+1)^{7/2}}$	2
Gaussian	$erf\left(\frac{\rho}{2^{1/2}}\right) - \rho\left(\frac{2}{\pi}\right)^{1/2} e^{-\rho^2/2}$	$\frac{1}{\rho} erf\left(\frac{\rho}{2^{1/2}}\right)$	$\left(\frac{2}{\pi}\right)^{1/2} e^{-\rho^2/2}$	2

The radial basis function interpolation(Barba et al. [2005]) is used every ten time steps to ensure the convergence of the core spreading method(Yokota et al. [2007]). The convection is solved by updating the position of vortex elements according to their velocity

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}_i \tag{2.12}$$

2.2.3 Cutoff Function

A number of 3D cutoff functions $\zeta(\rho)$ and their associted $f(\rho)$ and $g(\rho)$ have been used in vortex method calculations. Here I will introduce some of them which are mostly used summarized in Table 2.1. For the details and further explanations can be found in Hald [1987]; Winckelmans and Leonard [1993]; Winckelmans [2004]. The cutoff function is used for convergence of field discretization in vortex method calculation. It is an Gaussian smoothing function. In my calculation I have used hig-order algebraic function as of $\zeta(\rho)$.

2.3 Fast Methods

Vortex methods are a powerful tool for the simulation of incompressible flows at high Reynolds number. They rely on a discrete Lagrangian representation of the vorticity field to approximately satisfy the Kelvin and Helmholtz theorems which govern the dynamics of vorticity for inviscid flows. A time-splitting technique can be used to include viscous effects. The diffusion equation is considered separately after convecting the particles with an inviscid vortex method. In the present

work, the viscous effects are represented by the three dimensions core spreading method.

In order to accurately compute the viscous transport of vorticity, gradients of velocity need to be well resolved. As the Reynolds number is increased, these gradients get steeper and more particles are required to achieve the requisite resolution. In practice, the computing cost associated with the convection step dictates the number of vortex particles and puts an upper bound on the Reynolds number that can be simulated with confidence. That threshold can be increased by reducing the asymptotic time complexity of the convection step from $O(N^2)$ to $O(NlogN)$. The nearfield of every vortex particle is identified. Within that region, the velocity is computed by considering the pairwise interaction of vortices.

There are some well-known fast algorithms already been used to accelerate the vortex method calculations. Some of them are briefly introduced here as a general review. The details can be found in corresponding referenced papers.

2.3.1 The Tree Algorithm

Appel [1985] introduced a gridless method for many body simulations with a computational complexity estimated to be of the order $O(NlogN)$. Basically Appel introduced binary treecode for many-body simulations. A tree is a common hierarchical data structure used for many computer science applications. A tree is a recursive structure that usually maps an ordered set of data from an internal definition to some data space. Tree parts are often named after their contemporaries in family trees; trees contain nodes known as parent, child, and sibling. Trees are made of nodes, which can contain both data to be stored and always link to further levels in the tree. The tree code for two dimension flow has been developed by Barnes and Hut [1986].

An octree is a tree data-structure based on a cell with eight children. Each cell of an octree represents a cube in physical space. Each child represents one octant of its parent. At the leaves of the tree are the computational cells of the grid, and these cells are classified as either Cartesian or cut. These computational

cells hold the state vectors, spatial-derivatives of the states, centroid, volume, time-step, and any other information needed by the flow solver.

In tree code, the forces from a group of distant particles are approximated by multipole expansions. Hierarchical tree structure is used for grouping of the particles. The calculation procedures are available in Hernquist [1987]; Hernquist and Katz [1989]; Pfalzner and Gibbon [1996].

The calculation cost of tree code is $O(NlogN)$, since the cost is proportional to the number of levels of the tree. For the implementation on GRAPE hardwares, see Makino [1991] and Athanassoula et al. [1998]. Most of existing implementations of tree code use only up to a quadrupole moment and calculation cost rises quickly when high accuracy is required. For the implementation on distributed-memory parallel computers, see Salmon et al. [1994]; Warren and Salmon [1994].

2.3.2 Fast Multipole Method (FMM)

FMM was initially introduced by Rokhlin [1985] as a fast solution method for integral equations for two dimensional Laplace's equation. In his paper the term "FMM" did not appear but the main framework of FMM was constructed. After Rokhlin's work, Greengard and Rokhlin [1987, 1988] refined the algorithm, applied FMM to two and three-dimensional N-body problems with Coulombic potential and showed the applicability of FMM to various fields.

Rokhlin uses FMM in conjunction with an iterative solver to reduce the computational complexity for the matrix-vector multiplication from $O(N^2)$ to $O(N)$. In FMM Rokhlin uses multipole moments to represent distant particle groups and introduces a local expansion to evaluate the contribution from distant particles in the form of a series. The multipole moment associated with a distant group can be translated into the coefficient of the local expansion associated with a local group. In addition to Rokhlin's work, Greengard introduces the hierarchical decomposition of a spatial domain with a quad-tree in two dimensions and oct-tree in three dimensions to carry out efficient and systematic grouping of particles with tree structures. Greengard uses FMM to reduce the computational cost for the pairwise force calculation from $O(N^2)$ to $O(N)$. A new version of the FMM for the evaluation of potential fields in three dimensions was introduced

by Greengard and Rokhlin [1997]. In the last decades, FMM has been applied to various fields such as molecular dynamics (MD), vortex methods (Hu et al. [1997]; Cheng et al. [1999]). The fact that FMM was nominated as one of the top 10 algorithms of the century along with Fast Fourier Transform (FFT), QR algorithm, etc. in Board et al. [1995]; Board and Schulten [2000] show how much influence it had on numerical analyses.

Mathematical Formulations

Mathematical formulations of FMM to compatible with vortex method calculation have been introduced here briefly. The detail formulations and its implementation for three dimensional case is given in Schmidt et al. [1991]; Cheng et al. [1999]; Gumerov and Duraiswami [2004]; Yokota et al. [2007] and in appendix D.

The deduced form of original Biot-Savart equation (Eq. 2.3) can be written as

$$\mathbf{u}_i \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \left\{ \sum_{j=1}^{N} \boldsymbol{\gamma}_j M_j \right\} \times \nabla S_i \tag{2.13}$$

$$\mathbf{u}_i \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \left\{ \sum_{j=1}^{N} \boldsymbol{\gamma}_j L_j \right\} \times \nabla R_i. \tag{2.14}$$

Where the operators S, M, R, L are defined in appendix D.

Similarly, the stretching term of Eq. (2.4) can be written as

$$\frac{D\boldsymbol{\gamma}_i}{Dt} \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \left\{ \sum_{j=1}^{N} \boldsymbol{\gamma}_j \times \nabla M_j \right\} (\boldsymbol{\gamma}_i \cdot \nabla S_i) \tag{2.15}$$

$$\frac{D\boldsymbol{\gamma}_i}{Dt} \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \left\{ \sum_{j=1}^{N} \boldsymbol{\gamma}_j \times \nabla L_j \right\} (\boldsymbol{\gamma}_i \cdot \nabla R_i)\,. \tag{2.16}$$

The cutoff function does not appear in these equations since they are used to calculate the effect of the far field, for which it would have negligible effect.

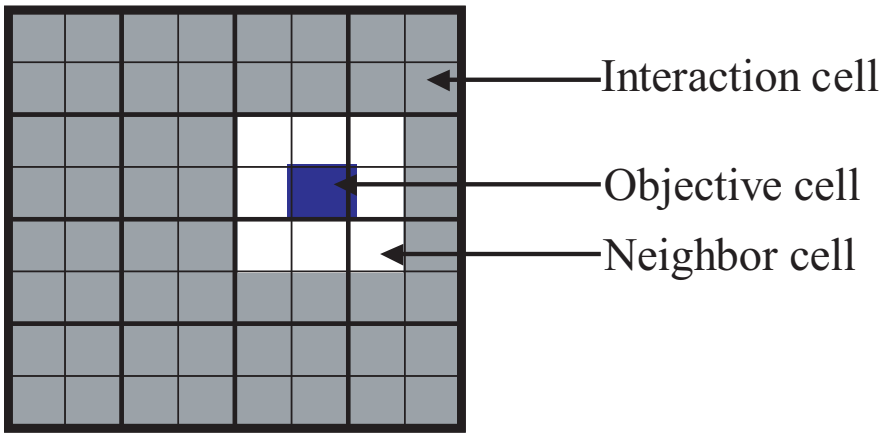


Figure 2.1: Relation between interaction cells (shaded) and neighbor cells of a hatched cell.

Brief Algorithm

Here the FMM for three dimensional case has been described briefly. It is assumed that all particles are uniformly distributed in a unit cube.

First, an oct-tree structure has been constructed by hierarchical subdivision of the cube. The division procedure starts from the root cell at refinement level $l = 0$, which covers the entire system. Here the refinement level l is defiend as the depth of the tree. The procedure has been repeated until a given refinement level l_{max} is reached. The level l_{max} is chosen so that the average number of particles in one leaf cell roughly equals the prescribed number which is determined to optimize the calculation speed.

In the next step, the multipole expansions are calculated for all leaf cells, then form multipole expansions for all non-leaf cells in coarser level by shifting and adding up expansions of their children. This step is called as upward pass as the tree traverse from leaf to root. In this step multipole expansion translations M2M has been performed.

The last step is downward in which the tree traverse from root to leaf. First the local expansion is formed at geometric center of each cell due to potential fieldof its interaction list.

The interaction list of a cell is the set of cells which are children of the nearest neighbors of the cell's parents and which are not neighbors of the cell itself.

Neighbor list of a cell is set of cells in the same level with the cell in question and have contact with the cell. Figure 2.1 shows the relation between neighbor and interaction cells of an object cell.

The potential field due to interaction list of the cell are calculated by converting their multipole expansion to local expansion of the cell and adding them up (M2L conversion). Then the local expansions at different refinement levels are summed up to obtain total potential field at all leaf cells.

Finally, the total force is being calculated on each particle. The total force is calculated as a sum of the distant and neighbor contributions. The distant part is calculated by evaluating the local expansion of the leaf cell at the position of the particle. The neighbor part is directly calculated by evaluating the particle-particle forces. The time complexity of FMM is $O(N)$. Therefore, the scaling of FMM is better than that of tree code. The most time consuming parts of FMM are multipole to local (M2L) translations and the direct interaction of particles.

The detail mathematical formulations used of FMM according to Cheng et al. [1999]; Gumerov and Duraiswami [2004]. Further details can be found in Appendix D.

2.3.3 Other Fast Methods

There are two more fast methods have been used to accelerate the vortex method calculations. One is Anderson's method which has been introduced by Anderson [1992]. The author proposed a simple formulation of FMM based on Poisson's formula. Poisson's formula gives the solution of the boundary value problem of the Laplace equation. The details discussion and mathematical formulations are discussed there and reference there in. In Anderson's method potential value is used to express the multipole/local expansion, while the original FMM uses the coefficients of the expansion terms. The advantage of Anderson's method over the original FMM is it's simplicity.

The other one is pseudo-particle multipole method (P^2M^2) has been proposed by Makino [1999]. Makino's and Anderson's methods are quite similar. Both approximate the multipole expansion by discrete values on points on a sphere (Hardin and Sloane [1996]). The difference is that P^2M^2 assigns mass to points

while Anderson's method assigns potential. The continuous mass distribution is again approximated by finite number of pseudo-particles, and the potential exerted by these pseudo-particles are calculated in the same way as that exerted by physical particles.

Conceptually, physical particles are converted to pseudo-particles in the two ways. First, multipole expansion has been calculated from the physical particles. Then mass of pseudo-particles are assigned so that they have the same multipole expansion as physical particles. For details algorithm and mathematical formulations can be found in Makino [1999] and Kawai et al. [2001].

Since M2L is dominant part in FMM calculation, total calculation speed for the same expansion order would be faster for P^2M^2. Also P^2M^2 has an additional advantage that it can make use of special-purpose computers to achieve further accelerations.

Chapter 3

Acceleration Techniques for Vortex Methods

3.1 Introduction

One of the main difficulties of vortex methods to be accepted in the mainstream of computational fluid dynamics is the numerical complexity of calculating the velocity using the Biot-Savart law, which is in fact analogous to an "N-body problem" and hence requires $O(N^2)$ operations for N vortex elements as of Fig. 3.1.

The large number of elements are required for accurate vortex methods calculation at high Reynolds number flows which is very high computation cost. Therefore, significant acceleration techniques are necessary to reduce the computation cost of N-body interaction calculation for millions of particles having the cost of $O(N^2)$ with growing N.

There are two techniques to reduce the force calculation cost of an N-body simulation which have been discussed in chapters 1 and 2, respectively. In the hardware techinques, there are two techniques, one is parallel computers and the other is special-purpose computers. To accelerate the vortex methods calculation, parallel calculation has been widely used in previous studies (Liu et al. [2000]; Ploumhans et al. [2002]; Sbalzarini et al. [2006]). Eventhough accelerate the calculation significantly, there are some difficulties to use parallel computations for longer calculations. It has limitations with parallelization according to

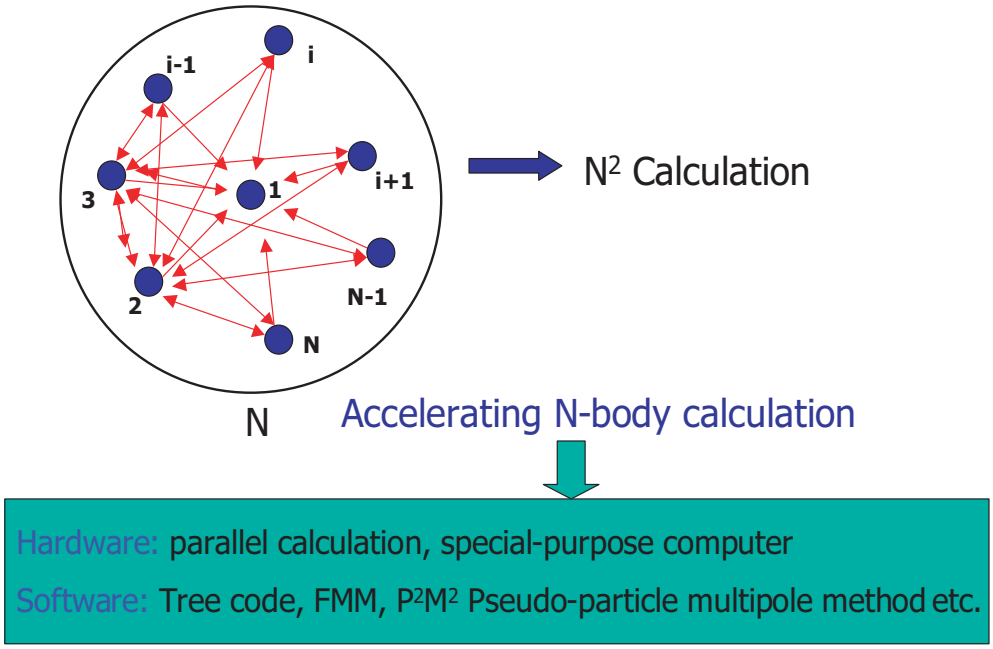


Figure 3.1: Bottleneck of vortex method calculation

hardware specifications. The memory bandwith is a big problem to calculate for large number of vortex elements which required special consideration. Power consumption and heat dissipation interupt the longer time calculations. These problems are becoming serious for advanced scientific computation. Shortcomings of parrallel computers, the special-purpose approach can solve parallelization limit thoroughly. It has relax power consumption according to hardware specification. The cost-performance is minimum 100 times better than that of parallel computation using ordinary cluster computers (Taiji et al. [2003]).

The special-purpose computer has been used in the present calculations to avoid the difficulties of parallel computations with higher speed. Hardware accelerators such as GRAPE-2A (Ito et al. [1993, 1994]), MD-GRAPE (Fukushige et al. [1996]), MD Engine (Toyoda et al. [1999]), MDGRAPE-2 (Susukita et al. [2003]), and MDGRAPE-3 (Taiji et al. [2003]; Narumi et al. [2006]) have been developed and successfully applied to MD simulations (Saito [1992]; Komeiji et al. [1997]; Susukita et al. [2003]). Yatsuyanagi et al. [2003a] has used MDGRAPE-2 to accelerate the velocity calculation of Biot-Savart integral equation for the simulation of 2D magnetohydrodynamics problems using the current vortex method.

In this chapter, some new acceleration techniques have been discussed which are used to accelerate present vortex method calculation at high Reynolds number flows.

3.2 The MDGRAPE-2

At first an MDGRAPE-2 board has been used to develop a fast vortex method. There are some steps to make MDGRAPE-2 as a compatible of vortex method calcaulation which will be discussed consequently. MDGRAPE-2 is a calculation accelerator board which dramatically increases the speed of molecular dynamics calculations by calculating the general force exerted between all pairs of particles in an N-body particle simulations (Narumi et al. [2001]; Susukita et al. [2003]). A brief discussion of MDGRAPE-2 architecture is introduced here, referred Narumi [1997] for details discussions.

3.2.1 Basic Structure

An MDGRAPE-2 board is a PCI long size card exclusively designed for MD simulations. This board is composed of four MDGRAPE-2 chips, interface logic, cell index counter, cell memory, particle index counter, and particle memory (Figure 3.13 in Narumi [1997]). A node computer communicates with MDGRAEP-2 chips and particle memory through interface logic. The index of a particle is determined by dual counters to support cell-index method. Cell index counter specifies the neighboring cell index c, and cell memory outputs the range of indices in the cell c. Particle index counter indicates the particle index j to particle memory. The position, charge, and particle type of a particle j are supplied to all of the MDGRAPE-2 chips and 8 Mbyte of SSRAM is used for the particle memory.

An MDGRAPE-2 chip is a data-flow-type numerical-processor LSI that is specially dedicated for molecular dynamics simulations. It is composed of four identical MDGRAPE-2 pipelines, one Input Unit, one Neighbor List Unit, and one Control Unit as shown in Figure 5.1 in Narumi [1997]. Four Pipelines calculate the forces on 24 different i-particles from one j-particle and accumulate them

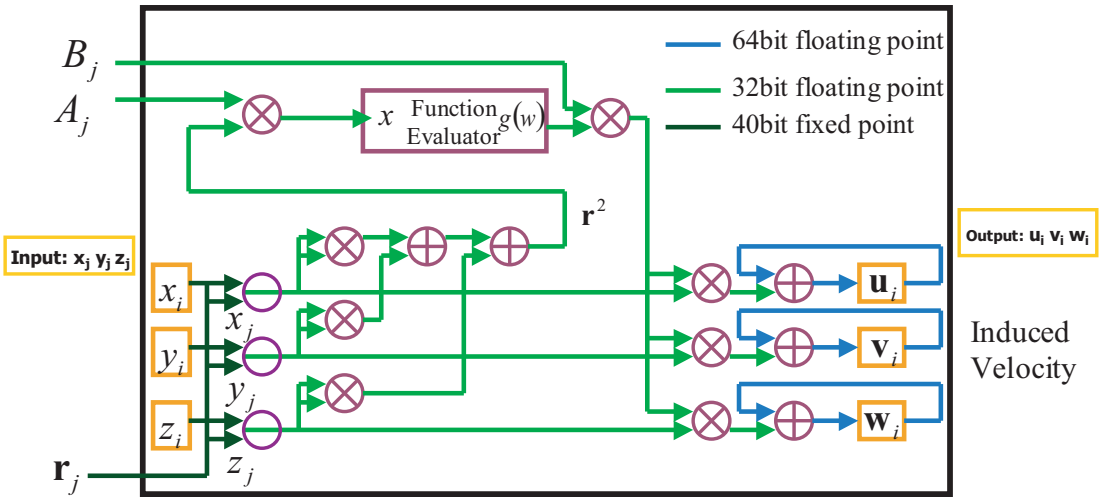


Figure 3.2: Block diagram of a pipeline of an MDGRAPE-2 chip

in every six clock cycles. Using Neighbor flags, supplied by Pipelines, Neighbor List Unit makes lists of neighbor particles of 24 i-particles. Control Unit controls the multiplexers and registers in the chip. Input Unit receives coordinates of j-particles (x_j, y_j, z_j), and scale factors (A_j, B_j), respectively. These data are stored in registers and kept in the next six clock cycles. Peak performance of an MDGRAPE-2 chip corresponds to about 16 GFlops at a clock frequency of 100 MHz.

An MDGRAPE-2 pipeline calculates the pairwise force $\vec{f}_{i,j}$ as:

$$\vec{f}_{i,j} = b_{ij} g\left(a_{ij} r_{ij}^2\right) \vec{r}_{ij} \tag{3.1}$$

where $g()$ is an arbitrary central force, and a_{ij} and b_{ij} are coefficients determined by particle types of particles i and j. Figure 3.2 shows the block diagram of the pipeline of an MDGRAPE-2 chip. The pipeline calculates $\vec{r}_{ij}$ and $a_{ij} r_{ij}^2$, and then evaluate $g()$ in the function evaluator(Figure 3.4). Function evaluator performs fourth-order interpolation segmented by $1,024$ region. The coefficients of the interpolation function are stored in the RAM in function evaluator. Therefore, it can be used any arbitrary central force by changing the contents of the RAM. After the function evaluator, the pipelines multiplies b_{ij} and $\vec{r}_{ij}$, and then accumulates them. The relative accuracy of a pairwise force is about 10^{-7}, since most of the arithmetice units in the pipeline use IEEE754 single floating point format.

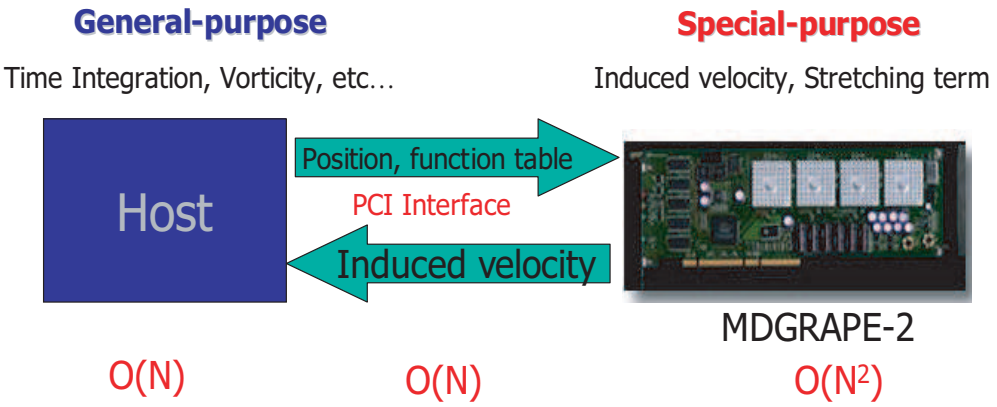


Figure 3.3: The basic structure of vortex methods calculation in MDGRAPE-2 system

The double floating point format is used for accumulating the force in order to prevent the underflow when large number of particles are used.

3.2.2 Calculation Procedures

Figure 3.3 represent the system that performs vortex method calculations consists of a host and MDGRAPE-2. The host is a PC or a workstation. MDGRAPE-2 consists of one or multiple MDGRAPE-2 boards. Each board is connected to the host as an extension board via a PCI bus (or PCI buses linked by PCI-PCI bridges). A special-purpose computer work as a back-end hardware and the host PC work as a front-end processor. The host computer calculates and update the positions of particles and send it to special-purpose computer. The special-purpose computer then calculates the induced velocity by using Biot-Savart law and stretching term which dominate the total calculation cost.

Since the load on MDGRAPE-2 is much heavier than those for host computer and communication interfaces between them, the total performance of the system is determined by the super-highspeed special-purpose computers rather than the host computer or communications, for a large number of particles. When the system performs VM simulations, MDGRAPE-2 calculates velocity from Biot-Savart law and Stretching term from vorticity equation. The host performs the other tasks such as giving the initial state of particles, time integration and the

control of MDGRAPE-2. The host sends necessary data for velocity or potential calculation to MDGRAPE-2 and receives velocity or potential from it via PCI bus (or the PCI buses).

A single board increases the computing speed of an ordinary PC to 64GFlops comparable to a supercomputer. The calculation of interactions between particles as represented by potential and force are carried out in MDGRAPE-2. In case of calculating the potential,

$$\Phi_i = \sum_{j=1}^{N} b_{ij} g(w) = \sum_{j=1}^{N} b_{ij} g\left(a_{ij}\left(|\mathbf{r}_{ij}|^2 + \epsilon_{ij}^2\right)\right) \tag{3.2}$$

and the force calculations

$$\mathbf{f}_i = \sum_{j=1}^{N} b_{ij} g(w) \mathbf{r}_{ij} = \sum_{j=1}^{N} b_{ij} g\left(a_{ij}\left(|\mathbf{r}_{ij}|^2 + \epsilon_{ij}^2\right)\right) \mathbf{r}_{ij} \tag{3.3}$$

are treated similarly, where $g(w)$ is an arbitrary function equivalent to an intermolecular force, and a_{ij}, b_{ij}, ϵ_{ij} are arbitrary coefficients which are settled down for every model. To apply these libraries to the calculation of a vortex method, Biot-Savart law in Eq. (2.3) is expressed as follows.

$$\mathbf{u}_i = \sum_{j=1}^{N} B_j g\left(A_j\left(|\mathbf{r}_{ij}|^2 + \epsilon_{ij}^2\right)\right) \mathbf{r}_{ij} \tag{3.4}$$

where A_j, B_j are arbitrary constants. The details mathematical formulations are introduced in chapter 4

The function $g()$ for an arbirary value $a|\mathbf{r}_{ij}|^2$ is calculated by interpolation, from values that are tabulated prior to the execution of the main program. If the interparticle distance is such that $a|\mathbf{r}_{ij}|^2$ falls out of this tabulated domain, the MDGRAPE-2 assumes $g()$ is zero. The number of tabulated points is constant. Thus, defining the table in a large domain would result in larger spacing between the tabulated points, and therefore a larger interpolation error. On the contrary, defining the table in a small domain would yield a higher probability that the interparticle spacing would fall outside the tabulated domain, which can also cause errors. The optimum range will be investigated for the vortex ring calculation by Sheel et al. [2007] for MDGRAPE-2 also in chapter 4.

3.3 Difficulties to use MDGRAPE-2

There are some difficulties to use MDGRAPE-2 for vortex method calculations to be solved while the architecture of the hardware is simple. First, the board is exclusively designed for molecular dynamics simulations but it is not designed for vector product calculations while Biot-Savart law contains vector product term. Second, it is important to produce an optimum function table in order to calculate the cut-off function considering the computational domain where the vortex elements are distributed. Third, due to hardware specifications, the subroutine runs partly with single precision, hence special care is necessary for floating-point operations. These difficulties have been solved carefully here as follows and in chapter 4.

3.3.1 Function Table

It is observed that an appropriate function table is essential to maintain the satisfactory accuracy for MDGRAPE-2 calculations. The hardware of MDGRAPE-2 is a board, which is mounted on a PCI-slot of a PC. The main program runs on the host PC, while the force calculation is done on the board via subroutine calls. Due to hardware specifications the subroutine runs partly with single precision, hence special care is necessary for floating-point operations. The important issue here is to rewrite the function table in MDGRAPE-2 that determines the formula of Eqs. (2.3) and (2.6) so that the range of the function table contains all elements inside the computational domain. The regorous investigation of computational domain is discussed in chapter 4. This is considered to be the primary importance of present research.

3.3.2 Function Evaluator

A Function Evaluator (FE) unit, a unit of pipeline of an MDGRAPE-2 chip(Fig. 3.2), evaluates $B_j g(w)$ cf. Eq. (3.4). It is composed of two 32-bit floating-point multipliers (32FMAJ, 32FMBJ), one 32-bit function evaluator (32FE) as shown in figure 3.4. The squared distance, $\mathbf{r}^2$ from distance vector unit, is multiplied with

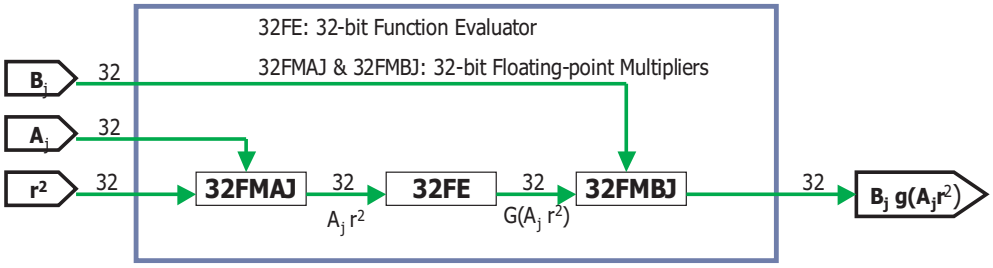


Figure 3.4: Function Evaluator unit of a pipeline of an MDGRAPE-2 chip(Narumi [1997])

the scale factor A_j by a 32-bit floating-point multiplier (32FMAJ). A Function Evaluator (32FE) evaluates $g(w)$ by 4-th order polynomial interpolation as,

$$g(w) = c_0 + w(c_1 + w(c_2 + w(c_3 + wc_4))) \tag{3.5}$$

In 32FE, c_0, c_1, c_2, c_3, and c_4 are represented by 2's complement integer format. The others are represented by unsigned integer format. The four coefficients are stored in a 1024-word 104-bit RAM on the board. The result of 32FE is multiplied with another scale factor B_j by a 32-bit floating-point multiplier (32FMBJ). The product $B_j g(A_j r^2)$ is sent to an accumulator unit. An accumulator unit accumulates forces in the force mode and potentials in the potential mode.

The basic concept on calculation of an arbitrary function by the FE unit is as follows:

A whole range $[x_{min},\ x_{max})$ is divided into 2^{10} segments. Assuming that an input x belongs to the $(k+1)$-th segment, i.e., $x \in [x_k, x_{k+1})$, the function $f(x)$ is approximated by a polynomial of degree 4;

$$f(x) \simeq \sum_{i=0}^{4} c_i^{(k)} (\Delta x)^i \tag{3.6}$$

where Δx is defined as the difference of x from the center of the segment x_c;

$$\Delta x = x - x_c \tag{3.7}$$

$$x_c = \frac{x_k + x_{k+1}}{2} \tag{3.8}$$

The coefficients $c_i^{(k)}$ are constants in each segment. For details see in Narumi [1997] ans Susukita et al. [2003].

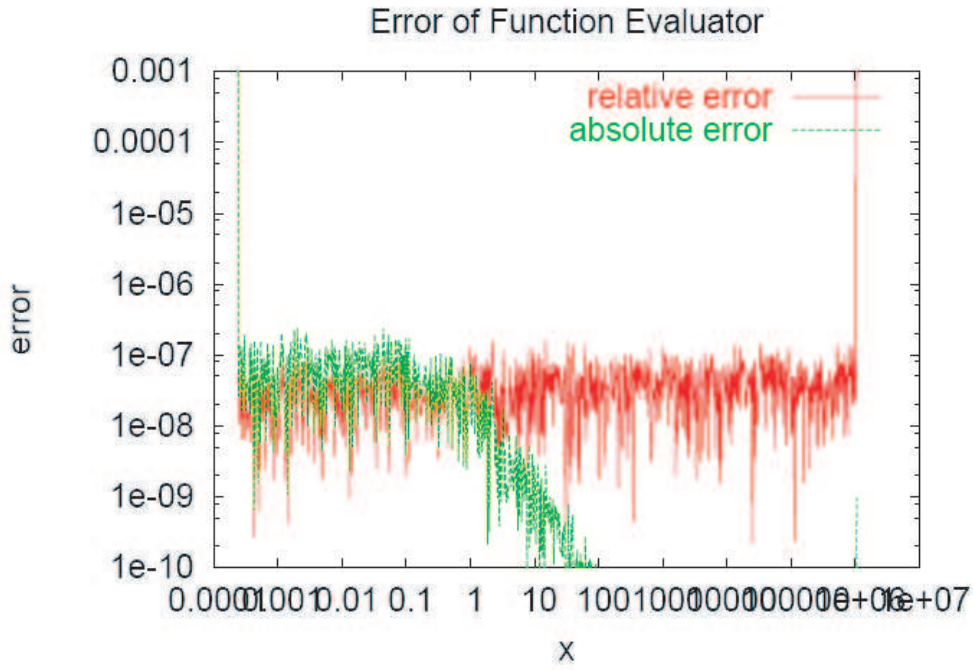


Figure 3.5: Relative and absolute error of MDGRAPE-2 chip

3.3.3 Error in Function Evaluator of MDGRAPE-2 chip

The net relative accuracy in the MDGRAPE-2 chip is set to 10^{-7}, since this accuracy is usually satisfactory in MD simulations. All the arithmatic units in an MDGRAPE-2 chip except FE use 64-bit and 32-bit floating-point format, which has an error smaller than 10^{-7}. In the calculation of the Biot-Savart law, the FE in a pipeline of MDGRAPE-2 chip calculates equation as follows.

$$g_1(w) = \frac{w + 5/2}{(w+1)^{5/2}}; \qquad w = \left(|\mathbf{r}_{ij}|/\sigma_j\right)^2 \tag{3.9}$$

Figure 3.5 shows the relative difference in $g_1(w) = (w + 5/2)/(w+1)^{5/2}$; where $w = \left(|\mathbf{r}_{ij}|/\sigma_j\right)^2$ between that calculated by double precision and that calculated by the software emulator of the FE. Relative accuracy is achieved 10^{-7}, which has been evaluated by 4-th order polynomial interpolation as of Eq. (3.5). Further details are introduced in chapter 4.

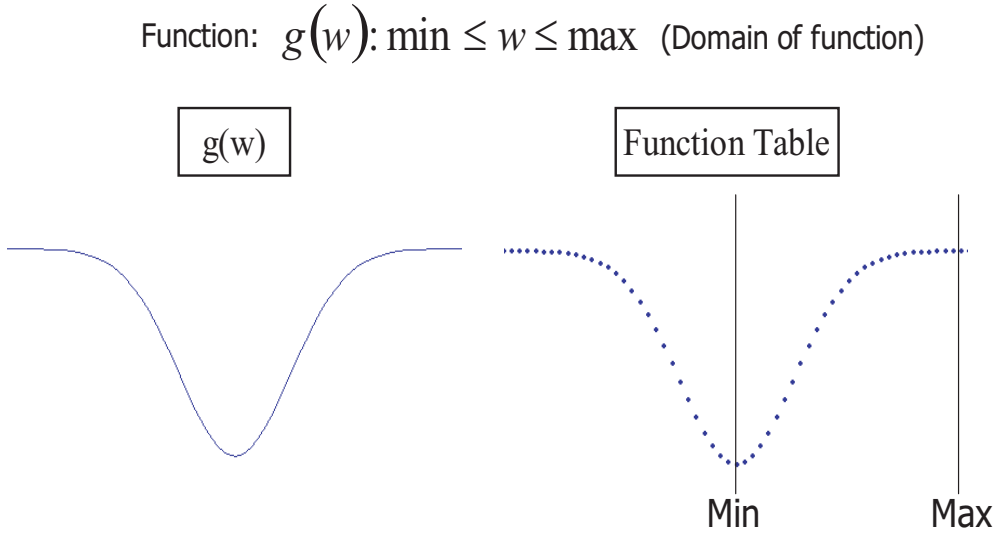


Figure 3.6: Function table in vortex method calculation.

3.3.4 Cutoff fucntion

The cut-off function is used in the vortex method calculation. It has the same shape between all particle interactions. It is important to produce an optimum function table in order to calculate the cut-off function considering the computational domain where the vortex elements are distributed (Chapter 4, section 4.3).

The function $g(w)$ (Fig. 3.6) is created prior to calculation and read during calculation. The domain of the function $g(w)$ is set to $w_{min} \leq w \leq w_{max}$ where $(w_{max}/w_{min}) \leq 2^{32}$ according to hardware specifications. To secure accurate calculations, it is important to know the domain correctly prior to calculation are discussed in chapter 4.

3.4 Performance and Implementations

3.4.1 Performance

The peak performance of the MDGRAPE-2 board is 64GFlops at 250MHz PC when the board calculates Coulomb forces for 30000 or more particles. This

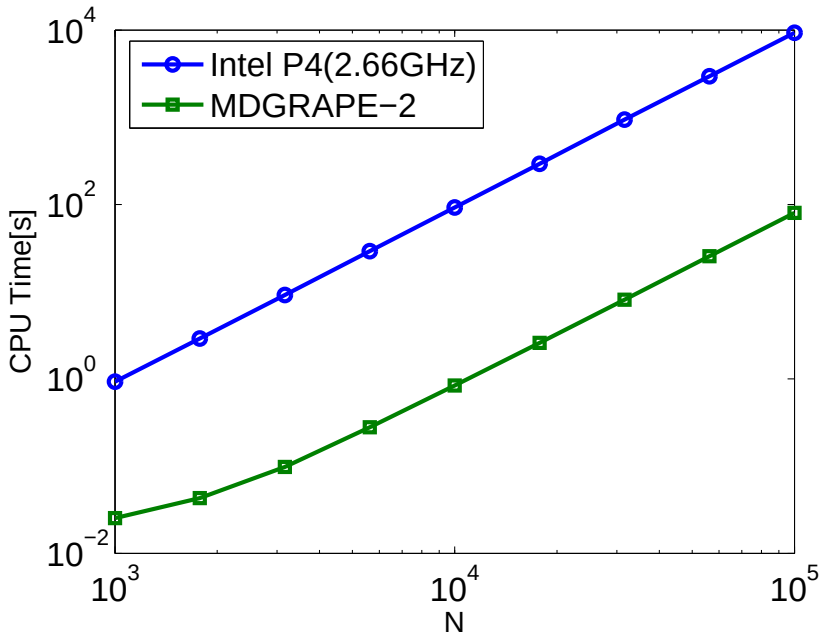


Figure 3.7: Calculation time against the vortex elements. ∘ –without the use of MDGRAPE-2 ; □ –with the use of MDGRAPE-2.

performance is not the same when vortex method has been calculated using this board. The CPU-time and effciency of VM calculations have been shown in Figures 3.7 and 3.8, respectively.

The computation time has been preformed and compared with ordinary PC (Intel Pentium 4 2.66GHz). The CPU-time has been calculated from Biot-Svart law (Eq. 2.3) and run for one time step while the number of particles are increased.

Figure 3.7 shows the calculation time against the number of vortex elements N with and without the use of MDGRAPE-2. The CPU-time has been achieved for the collision of two inclined vortex rings. The legends 'Intel P4(2.66GHz)' and 'MDGRAPE-2' represent the calculation time without and with the use of MDGRAPE-2, respectively. It is clearly observed that the calculation time is reduced by a factor of 100 for $N \sim 10^5$. This acceleration rate is below the expectated rate, but it can be improved by reducing the number of calls to the MDGRAPE-2 library for cross product calculations.

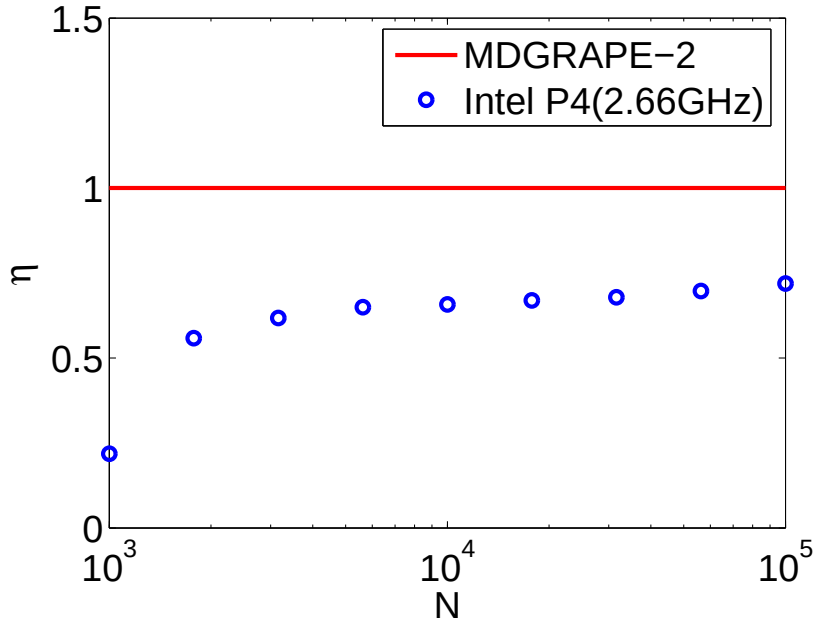


Figure 3.8: The results of efficiency measurement for the MDGRAPE-2 board.

The efficiency and breakdown of overhead communication for the host PC have been calculated as follows.

The total number of pairwise interactions (N_{step}) for one time step is counted from

$$N_{step} = nmd(xmd + ymd + zmd) \tag{3.10}$$

Where xmd, ymd, and zmd represent the number of pairwise interactions in x-direction, y-direction, z-direction.

The total number of pairwise interactions (N_{APPL}) for one second has been calculated from

$$N_{APPL} = \frac{N_{step}}{CPU\ \ Time(sec/step)} \tag{3.11}$$

Finally the efficiency can be defined by

$$\eta = \frac{N_{APPL}}{N_{GRAPE}} \tag{3.12}$$

Table 3.1: Hosts for performance measurement

Architecture	Host CPU	Cache	Memory	OS	Compiler
MDGRAPE-2	Intel P4 2.66GHz (1CPU 1 core)	512KB	1GB (2GB Swap memory)	Linux 8.0 Kernel 2.4.18-14	ifort
MDGRAPE-3	Xeon 5160 3.00 GHz (1CPU 2 core)	4096KB	32GB (0GB Swap memory)	Cent 4.3 (Final) Kernel 2.6.9-34_ELsmp	ifort

Where N_{GRAPE} represent the total number of pairwise interactions for one second (peak performance of MDGRAPE-2) for coulom force calculation which is $N_{GRAPE} = 5.5 \times 10^8$.

Result of the efficiency measurements is shown in Figure 3.8. Host use for the measurement is listed in Table 3.1. The N in x-axis represents the number of pariwise interactions particles for one second (N_{APPL}) and the y-axis represents the maximum efficiency of corresponding N. The solid line represents the peak performance of MDGRAPE-2 board for molecular dynamics calculations and circle represents the maximum efficiency (η) of vortex method calculations. From the results of this measurement, it is proved that the efficiency is improved with increase of number of particles, the board provides high performance if the number of particles is large. It is observed from the figure that the efficiency of the present calculation is 70% compared with the peak performance of MDGRAPE-2 for the largest N. This bellow efficiency may caused in due to power supply to MDGRAPE-2 during the calculation. This performance does not depend on the host PC instead of N and MDGRAPE-2 (Susukita et al. [2003]).

3.4.2 Implementations

Here I will show a sample program that use MDGRAPE-2 run on the host. To use MDGRAPE-2, the program calls the interface to it. To calculate the Biot-Savart law expressed as Eq. 2.3, the program calls as follows. The corresponding

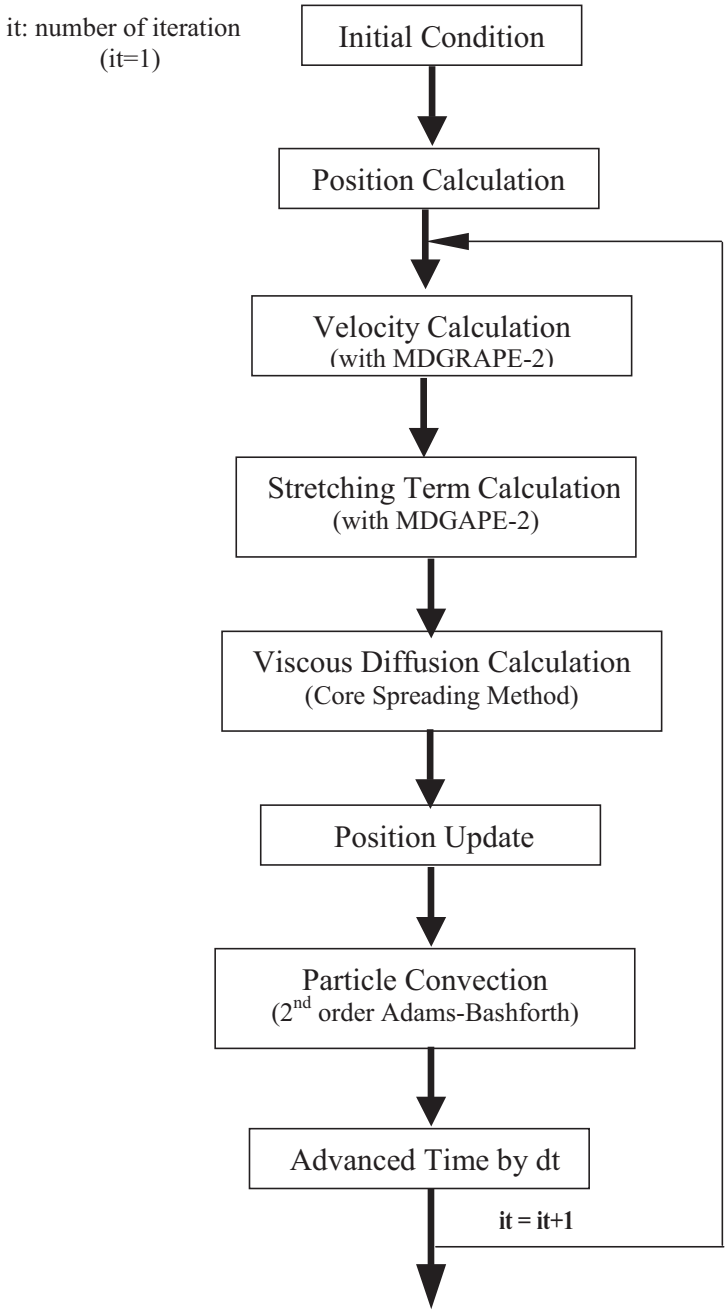


Figure 3.9: The flowchart for VM calculation using MDGRAPE-2.

flowchart is shown in Figure 3.9.

VM (Biot-Savart) program using the MDGRAPE-2 API's

```
C Initialization of particle positions and scale coefficients
do i = jsta,jend
    nmd = i-jsta+1
    amd(nmd) = 1/sj(i)** 2
    pos(1,nmd) = xj(i)
    pos(2,nmd) = yj(i)
    pos(3,nmd) = zj(i)
    bxmd(nmd) = gxj(i)/sj(i)** 3
    bymd(nmd) = gyj(i)/sj(i)** 3
    bzmd(nmd) = gzj(i)/sj(i)** 3
enddo
C Allocate and capture MDGRAPE-2 board for velocity calculation
n_unit = m2_allocate_unit('force.table',m2_force, * xminf,xmaxf,null_integer)
call m2_set_positions(n_unit,pos,nmd)
call m2_set_rscales(n_unit,amd,nmd)
C Positions stored in memory
do i = n0,n1
    nmdd = i-n0+1
    pos(1,nmdd) = xi(i)
    pos(2,nmdd) = yi(i)
    pos(3,nmdd) = zi(i)
end do
call m2_set_charges(n_unit,bxmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,xmd)
call m2_set_charges(n_unit,bymd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,ymd)
call m2_set_charges(n_unit,bzmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,zmd)
C Calculate total velocities and communicate with host machine
```

call m2_free_unit(n_unit)
C Other finalization with host PC

VM (Stretching) program using the MDGRAPE-2 API's

Sample programs for stretching term calculations using the MDGRAPE-2 API's are as follows. The flowchart is as of Figure 3.9.
C Initialization of particle positions and scale coefficients
C Allocate and capture MDGRAPE-2 board for vortex strength calculation in potential mode
n_unit = m2_allocate_unit('potential.table',m2_potential,
∗ xminf,xmaxf,null_integer)
call m2_set_positions(n_unit,pos,nmd)
call m2_set_rscales(n_unit,amd,nmd)

call m2_set_charges(n_unit,bxmd,nmd)
call m2_calculate_potentials(n_unit,pos,nmdd,bxmd)
call m2_set_charges(n_unit,bymd,nmd)
call m2_calculate_potentials(n_unit,pos,nmdd,bymd)
call m2_set_charges(n_unit,bzmd,nmd)
call m2_calculate_potentials(n_unit,pos,nmdd,bzmd)

C Calculate total strength and communicate with host machine
C Other calculation by host PC
call m2_free_unit(n_unit)

C Other initialization of particle positions and scale coefficients
C Allocate and capture MDGRAPE-2 board for vortex strength calculation in force mode
n_unit = m2_allocate_unit('force.table',m2_force, ∗ xminf,xmaxf,null_integer)
call m2_set_positions(n_unit,pos,nmd)
call m2_set_rscales(n_unit,amd,nmd)

```
call m2_set_charges(n_unit,bmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,gmd)
call m2_set_charges(n_unit,bxmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,xmd)
call m2_set_charges(n_unit,bymd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,ymd)
call m2_set_charges(n_unit,bzmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,zmd)

call m2_set_charges(n_unit,bmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,gmd)
call m2_set_charges(n_unit,bxmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,xmd)
call m2_set_charges(n_unit,bymd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,ymd)
call m2_set_charges(n_unit,bzmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,zmd)

call m2_set_charges(n_unit,bmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,gmd)
call m2_set_charges(n_unit,bxmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,xmd)
call m2_set_charges(n_unit,bymd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,ymd)
call m2_set_charges(n_unit,bzmd,nmd)
call m2_calculate_forces(n_unit,pos,nmdd,zmd)

C Calculate total strength and communicate with host machine
C Other calculation by host PC
call m2_free_unit(n_unit)

C Other finalization with host PC
```

3.5 The MDGRAPE-3

MDGRAPE-3 is a successor of MDGRAPE-2 and the third model of MD-GRAPE series. The nominal peak performance of the board for a Coulomb force calculation is 2.16 TFLOPS at 250 MHz for classical molecular dynamics simulations (Taiji et al. [2003]; Narumi et al. [2006]). The basic architecture and calculation systems are simlar as of its predecessor, MDGRAPE-2. The brief introduction of this processor is as follows.

3.5.1 Basic Architecture

The block diagram of an MDGRAPE-3 board can be seen in Narumi et al. [2006] (Fig. 1). It consists of twelve MDGRAPE-3 chips, and each chip is connected in serial to send/receive the data. Since the memory is embeded in the MDGRAPE-3 chip, the board will be extremely simple. The speed of the communication between the chips will be 1.3 Gbytes/sec, which corresponds to an 80-bit word transfer at 133 MHz. For these connections 1.5V-CMOS I/O cells will be used. The board has a control FPGA (or ASIC) with a special bus with 1 Gbytes/sec peak speed. The force calculations are parallelized as follows.

Parallelization in index i (i-parallelization) is adopted between the 40 parallel pipeline in each chip, and the forces *on* different particles are calculated. This is the most efficient solution since the pipelines in a chip share the memory unit due to the broadcast memory architecture. Typically, the parallelization in index j (j-parallelization) is used between 12 chips on the board, and the forces $from$ different particle groups are calculated. The j-parallelization is used for two reasons. First, the number of j-particles in a single-step calculation is multiplied by the number of connected chips. The chip has memory for only 32,768 particles; however, more particles often need to be considered. Second, the use of i-parallelization between the chips increases the number of i-particles to as high as 480. By using the cell-index method, the number of i-particles required for performing a parallel calculation may often be as small as 100; a large number of pipelines tends to increase their idle time. In the j-parallelization, each chip calculates a partial force. When the resulting force is returned, the MDGRAPE-3 chips accumulate the partial forces to decrease the amount of data

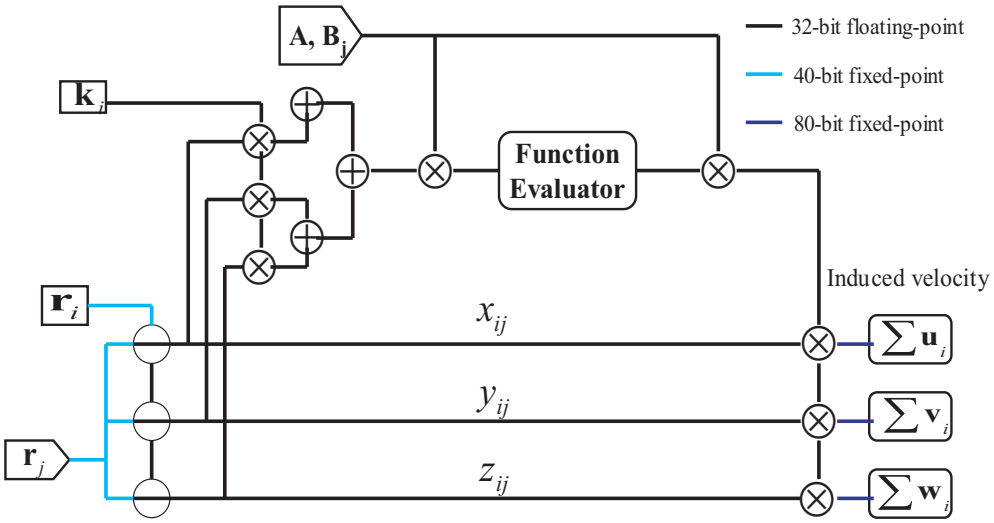


Figure 3.10: Block diagram of the force calculation pipeline in the MDGRAPE-3 chip (Taiji et al. [2003])

transferred to the host computer. Thus, the MDGRAPE-3 board is equivalent to an MDGRAPE-3 chip having 12-times higher processing speed and memory.

Figure 3.10 shows the block diagram of the force calculation pipeline of an MDGRAPE-3 chip. It will consist of three subtractor units, six adder units, eight multiplier units, and one function-evaluation unit. It can perform about 33 equivalent operation per cycle. In this case the function-evaluation unit calculates $x^{-3/2}$, which is equivalent to 16 floating-point operations. The count depends on the velocity to be calculated. Most of the arithmetic operations are done in 32-bit single-precision floating-point format, with the exception of the force accumulation. The induced velocity is accumulated in 80-bit fixed-point format and it can be converted to 64-bit double-precision floating-point format.

The function evaluator, which allows calculation of an arbitrary smooth function, is the most important part of the pipeline. This block is almost the same those in MDGRAPE-2 (section 3.2). It has a memory unit which contains a table for polynomial coefficients and exponents, and a hardwared pipeline for the fourth-order polynomial evaluation. It interpolates an arbitrary smooth function

$g(x)$ using segmented fourth-order polynomials by the Horner's method

$$g(x_0 + \Delta x) = (((c_4[x_0]\Delta x + c_3[x_0])\Delta x + c_2[x_0])\Delta x + c_1[x_0])\Delta x + c_0[x_0], \quad (3.13)$$

where x_0 is the center of the segmented interval, $\Delta x = x - x_0$ is the displacement from the center, and $c_k[x_0]$ are the coefficients of the polynomial.

The MDGRAPE-3 chip has 20 force calculation pipelines, a j-particle memory unit, a cell-index controller, a force summation unit, and a master controller. The master controller manages the timings and the inputs/outputs of the chip. The j-particle memory unit holds the coordinates of j-particles for 32,768 bodies and corresponds to the 'main memory' in general-purpose computers. Thus, the chip is designed using the memory-in-the-chip architecture and no extra memory is necessary on the system board. The amount of the memory is 6.6 Mbits and will be cnstructed by a static RAM. The same output of the memory is sent to all the pipelines simultaneously. The details of MDGRAPE-3 chip and its block diagram are in Taiji et al. [2003] (figure 4). Each pipeline calculates using the same data from j-particle unit and individual data stored in the local memory of the pipeline. Because the two-body force calculation is given by

$$\boldsymbol{F}_i = \sum_j f(\boldsymbol{r}_i, \boldsymbol{r}_j) \quad (3.14)$$

fro the parallel calculation of multipole $\boldsymbol{F}_i$, we can use the same $\boldsymbol{r}_i$. This parallelization scheme, 'the broadcast memory architecture', is one of the most important advantages of the GRAPE systems. It enables the coefficient parallelization at low bandwith realized by simple hardware.

3.5.2 Calculations System

One small MDGRAPE-3 board (consists of 2 chips) has the peak performance of 330 GFlops. In order to communicate with the host computer, a field-programmable gate array (FPGA, Xilinx XC2VP30) is installed on the board. It also controls the chips, thermal sensors, and so on. The board is connected to the host by a 10-Gbps serial communication link with a 4-lane 2.5-Gbps Xilinx RocketIO through

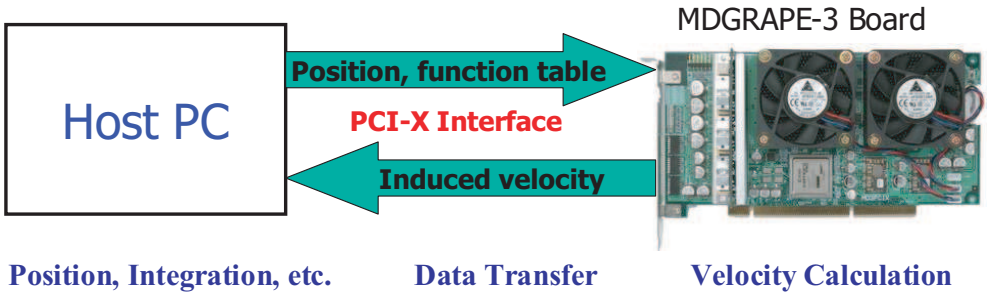


Figure 3.11: The basic structure of vortex methods calculation in MDGRAPE-3 system

an InfiniBand cable. The host computer has an interface card with an FPGA (Xilinx XC2VP7) attached to a PCI-X bus (Fig. 3.11). The present calculations, Xeon 5160 (3.0GHz) duel core processor has been used as a host PC. The calculation and data transfer systems are same as of MDGRAPE-2. It can speed up force calculation about 100-1000 times faster than that of general purpose computers of the same cost and the time complexity of force calculation is $O(N^2)$ for direct summation algorithm.

The MDGRAPE-3 chip functions as the core LSI of the system, and it performs the force calculations. To be used MDGRAPE-3, it is necessary to modify the program in order to call the application programing interface (API) of this system. The MDGRAPE-3 board can perform force calculations on many i-particles without any control of the host computer. Therefore, the host computer can perform other calculations, while the board performs the force calculations. Consequently, parallel calculations can be conveniently performed by the MDGRAPE-3 system since it can perform long calculations without host. One major problem in this sense is that the MDGRAPE-3 chips can only handle two types of calculations. The Coulomb potential

$$\Phi_i = \sum_{j=1}^{N} b_j g\left(a|\mathbf{r}_{ij}|^2\right), \tag{3.15}$$

and Coulomb force

$$f_i = \sum_{j=1}^{N} b_j g\left(a|\mathbf{r}_{ij}|^2\right) \mathbf{r}_{ij}. \tag{3.16}$$

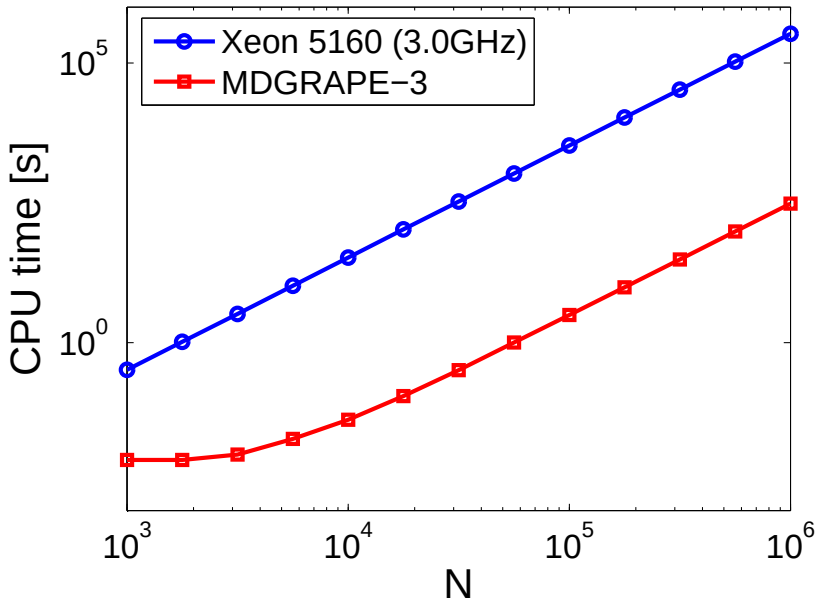


Figure 3.12: Calculation time against the vortex elements. ○–without the use of MDGRAPE-3 ; □–with the use of MDGRAPE-3.

$g()$ is an arbitrary function, which must be defined prior to the calculation same as of MDGRAPE-2. a and b_j are constants, which can be used for scaling.

3.6 Performance and Implementations

3.6.1 Performance

The peak performance of the MDGRAPE-3 board is 330GFlops when the board calculates Coulomb forces for 32000 or more particles. This performance is not the same when vortex method has been calculated using this board. The CPU-time and effciency of VM calculations have been shown in Figures 3.12 and 3.13, respectively.

The computation time has been preformed and compared with ordinary PC. The CPU-time has been calculated from Biot-Svart law (Eq. 2.3) and run for one time step while the number of particles are increased.

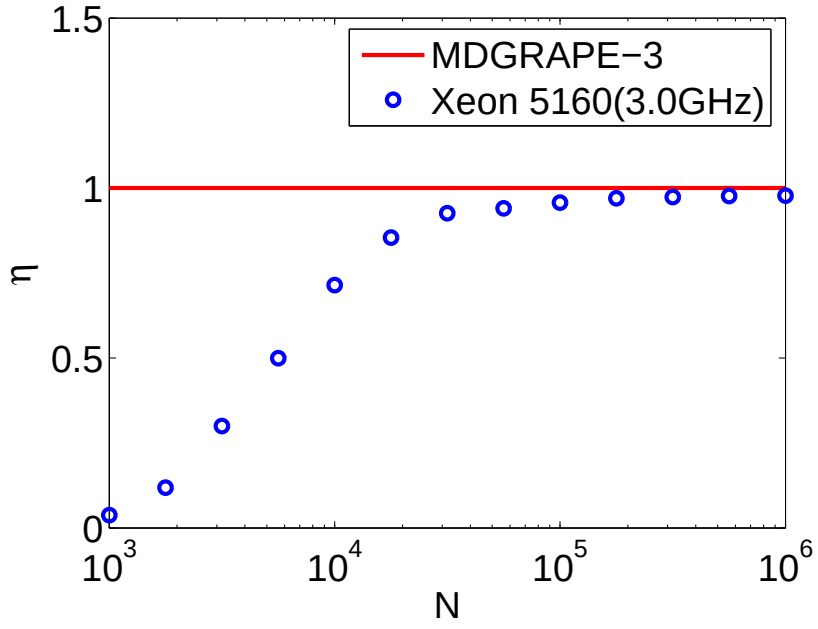


Figure 3.13: The results of efficiency measurement for the MDGRAPE-3 board.

Figure 3.12 shows the calculation time against the number of vortex elements with and without the use of MDGRAPE-3. The calculation time has been obtained for the calculation of the impingement of two identical inclined vortex rings. It is clearly seen that the computation time is reduced by a factor of 1000 for $N \sim 10^6$. This acceleration rate is above the speed of MD simulations and it can be further improved by reducing the number of calls to the MDGRAPE-3 library for cross product calculations.

The efficiency and breakdown of overhead communication for the host PC have been calculated similar as of Equations (3.10) to (3.12).

Result of the efficiency measurements is shown in Figure 3.8. Host use for the measurement is listed in Table 3.1. The N in x-axis represents the number of pairwise interactions particles for one second (Eq. 3.11) and the y-axis represents the maximum efficiency of corresponding N. The solid line represents the peak performance of MDGRAPE-3 board for molecular dynamics calculations and circle represents the maximum efficiency(η) of vortex method calculations. From the results of this measurement, it is proved that the efficiency is improved

with increase of number of particles and whatever the host is, the board provides high performance if the number of particles is large. It is clearly observed from the figure that the efficiency of the present calculation is 98% compared with the peak performance of MDGRAPE-3($N_{GRAPE} = 10^{10}$) for the largest N of VM calculations.

3.6.2 Implementations

Here again I will show a sample program that use MDGRAPE-3 run on the host. To use MDGRAPE-3, the program calls the interface to it. To calculate the Biot-Savart law expressed as Eq. 2.3, the program calls as follows. The corresponding flowchart is shown in Figure 3.14. The API's are similar as MDGRAPE-2 except the CALLing variable names.

VM (Biot-Savart) program using the MDGRAPE-3 API's

```
C Initialization of particle positions and scale coefficients
C Allocate and capture MDGRAPE-3 board for velocity calculation
n_unit = m3_allocate_unit('force.table',m3_force, * xminn,xmaxn,null_integer)
call m3_set_positions(n_unit,pos,nmd)
do i = ista,iend
    nmdd = i-ista+1
    pos(1,nmdd) = xi(i)/sj(1)
    pos(2,nmdd) = yi(i)/sj(1)
    pos(3,nmdd) = zi(i)/sj(1)
end do
call m3_set_charges(n_unit,bxmd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,xmd)
call m3_set_charges(n_unit,bymd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,ymd)
call m3_set_charges(n_unit,bzmd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,zmd)
C Calculate total velocities and communicate with host machine
call m3_free_unit(n_unit)
```

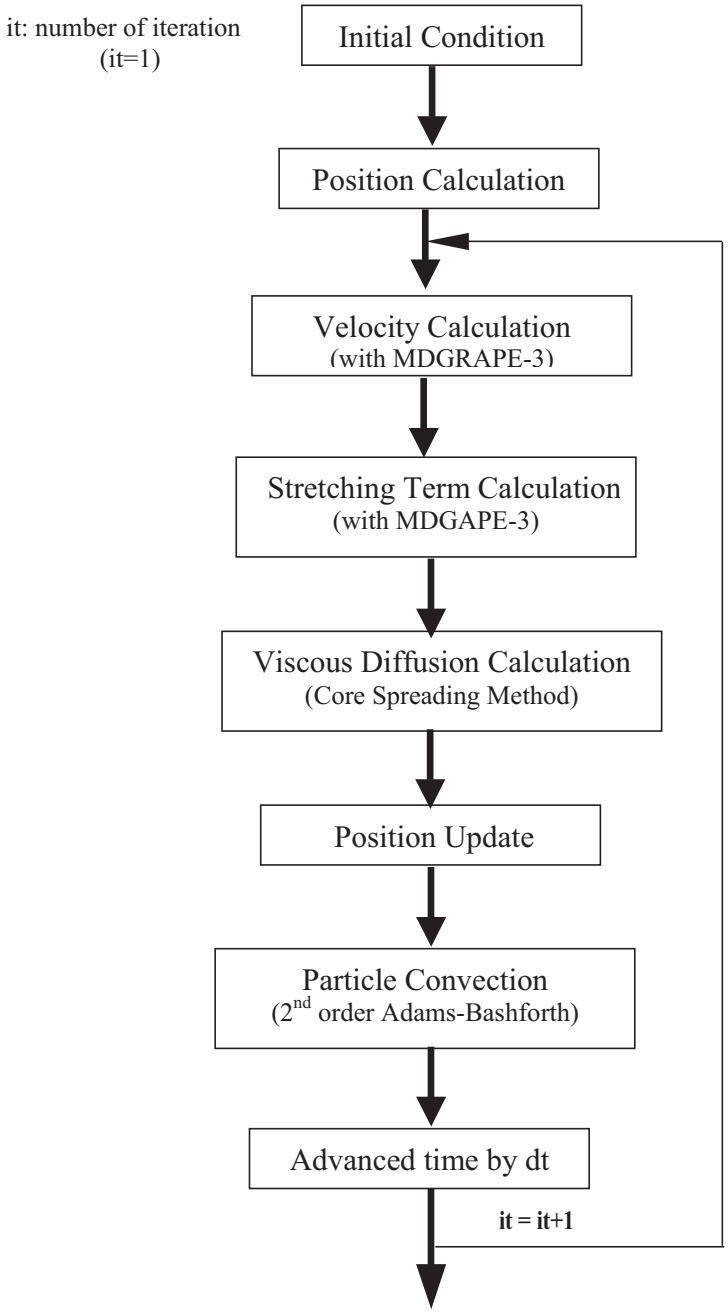


Figure 3.14: The flowchart for VM calculation using MDGRAPE-3.

C Other finalization with host PC
Sample programs for stretching term calculations using the MDGRAPE-3 API's are as follows. The flowchart is as of Figure 3.14.

VM (Stretching) program using the MDGRAPE-3 API's

C Initialization of particle positions and scale coefficients
C Allocate and capture MDGRAPE-3 board for vortex strength calculation in potential mode
n_unit = m3_allocate_unit('potential.table',m3_potential, * xminn,xmaxn,null_integer)
call m3_set_positions(n_unit,pos,nmd)
call m3_set_charges(n_unit,bxmd,nmd)
call m3_calculate_potentials(n_unit,pos,nmdd,bxmd)
call m3_set_charges(n_unit,bymd,nmd)
call m3_calculate_potentials(n_unit,pos,nmdd,bymd)
call m3_set_charges(n_unit,bzmd,nmd)
call m3_calculate_potentials(n_unit,pos,nmdd,bzmd)
C Calculate total strength and communicate with host machine
C Other calculation by host PC
call m3_free_unit(n_unit)
C Other initialization of particle positions and scale coefficients
C Allocate and capture MDGRAPE-3 board for vortex strength calculation in force mode
n_unit = m3_allocate_unit('force.table',m3_force, * xminn,xmaxn,null_integer)
call m3_set_positions(n_unit,pos,nmd)
call m3_set_charges(n_unit,bxmd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,xmd)
call m3_set_charges(n_unit,bymd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,ymd)
call m3_set_charges(n_unit,bzmd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,zmd)

call m3_set_charges(n_unit,bxmd,nmd)

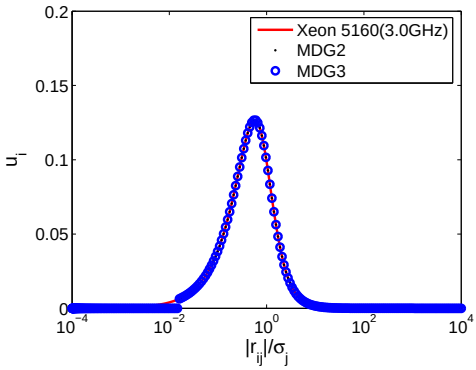
```
call m3_calculate_forces(n_unit,pos,nmdd,xmd)
call m3_set_charges(n_unit,bymd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,ymd)
call m3_set_charges(n_unit,bzmd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,zmd)

call m3_set_charges(n_unit,bxmd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,xmd)
call m3_set_charges(n_unit,bymd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,ymd)
call m3_set_charges(n_unit,bzmd,nmd)
call m3_calculate_forces(n_unit,pos,nmdd,zmd)
C Calculate total strength and communicate with host machine
C Other calculation by host PC
call m3_free_unit(n_unit)
C Other finalization with host PC
```

3.7 Comparative Study between MDGRAPE-2 and MDGRAPE-3

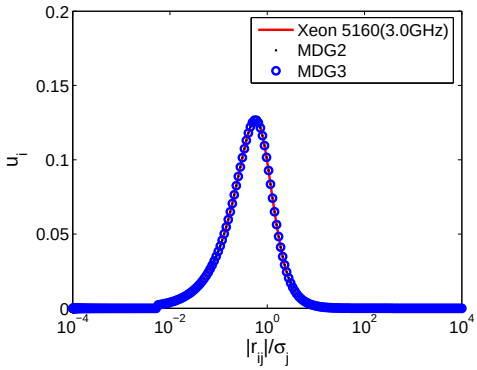
The difficulties to use MDGRAPE-3 are the same as of MDGRAPE-2 and which has been solved in details for MDGRAPE-2. The only difference between them is that MDGRAPE-3 can simultaneously calculate along with the host machine, but can only handle a small number of source particles at once (Narumi et al. [2006]). However, these differences do not have any effect on the same critical issues once solved and the findings of MDGRAPE-2 can be directly used for MDGRAPE-3.

3.7.1 Scaling Error

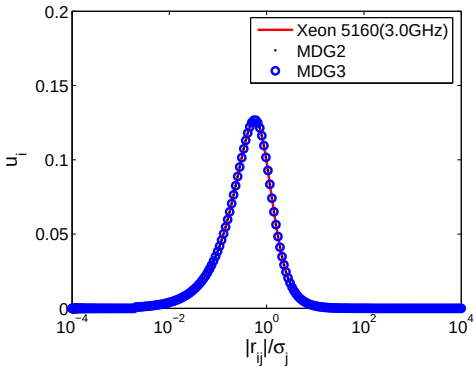
Here I will first confirm that MDGRAPE-3 outputs the same results as the previous calculations using MDGRAPE-2. Figure 3.15 shows typical velocity distribution on a logarithmic scale, calculated from Eq. (2.3), with and without the use of MDGRAPE for six different input ranges, where a single source particle



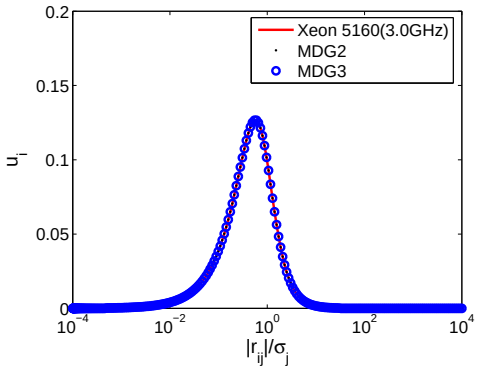
(a) $2^{-12} \leq w \leq 2^{20}$



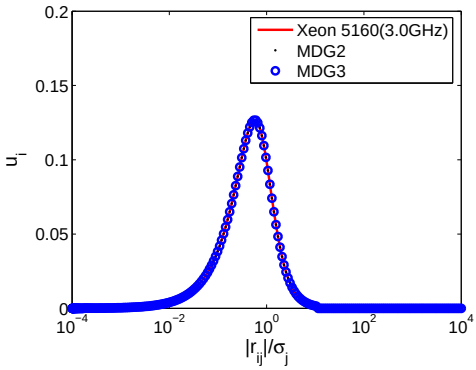
(b) $2^{-16} \leq w \leq 2^{16}$



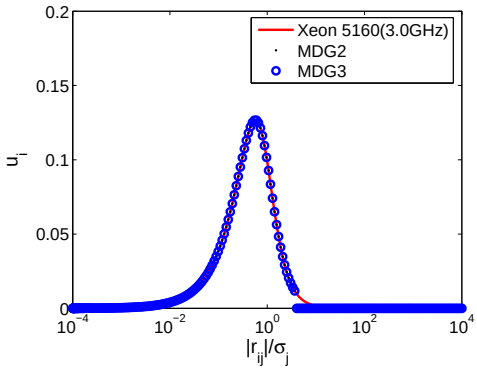
(c) $2^{-18} \leq w \leq 2^{14}$



(d) $2^{-22} \leq w \leq 2^{10}$



(e) $2^{-24} \leq w \leq 2^{8}$



(f) $2^{-28} \leq w \leq 2^{4}$

Figure 3.15: Different ranges of a function table.

is positioned at the origin and 1000 target particles are distributed from 10^{-4} to 10^{4}. Xeon 5160(3.0GHz), MDG2, and MDG3 stand for calculations without MDGRAPE, with MDGRAPE-2, and with MDGRAPE-3, respectively. The velocity becomes zero when $|\mathbf{r}_{ij}|/\sigma_j$ falls outside of the range of the table. Otherwise, the results of the Biot-Savart calculation on MDGRAPE-2 and MDGRAPE-3 match those of the results on the host computer in each case.

Figure 3.16 shows the comparative error for different input ranges between MDGRAPE-2 and MDGRAPE-3. It can be easily observed that the errors are different for different ranges. The errors in figures 3.16(a) and 3.16(b) are below 10^{-5}. The rest of errors are larger and are not satisfactory in the vortex method calculations. The optimum range is determined $1e^{-2} \sim 1e^{6}$ for entire calculations.

3.7.2 CPU-time and L^2 norm error

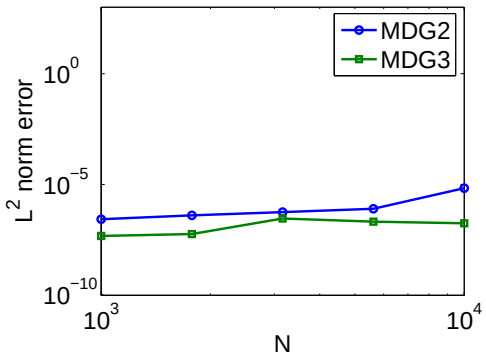
The calculation cost and accuracy are important issues for any numerical simulation. In this calculation these two factors have been investigated carefully. The calculation has been accelerated retained the accuracy at an acceptable label. The CPU-time has been compared with different acceleration techniques at one time step by changing the number of particles.

The L^2 norm error is defined as the difference in the induced velocity of the same particles between the host and MDGRAPE for the same time step as follows.

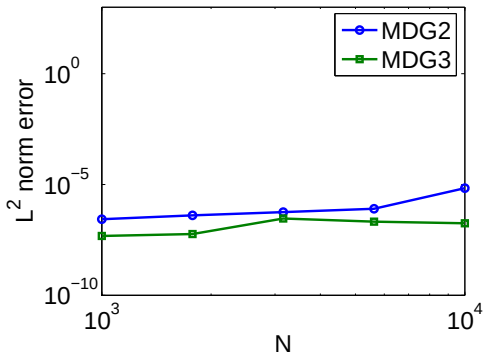
$$L^2(norm\ \ error) = \frac{\sum\left((u_{host}-u_{md})^2+(v_{host}-v_{md})^2+(w_{host}-w_{md})^2\right)}{\sum\left(u_{host}^2+v_{host}^2+w_{host}^2\right)} \quad (3.17)$$

where the suffices *md* and *host* represent with and without the use of MDGRAPE, respectively.

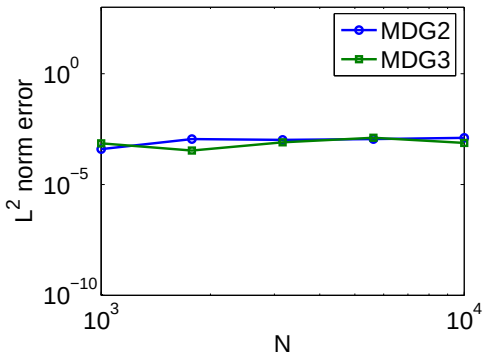
Figure 3.17 shows the cpu-time for one time step against the number of vortex elements with and without the use of MDGRAPE. The legends 'Xeon 5160(3.0GHz)', 'MDG2', and 'MDG3' correspond to the calculations without the use of MDGRAPE and with the use of MDGRAPE-2 and MDGRAPE-3. It is clearly seen that the calculation time has been reduced with the use of both schemes for $N \sim 10^6$ when compared with the host calculation time. From the figure it is shown that MDGRAPE-3 calculation has been further accelerated than MDGRAPE-2 when compared with host calculation. The MDGRAPE-3 is 1000



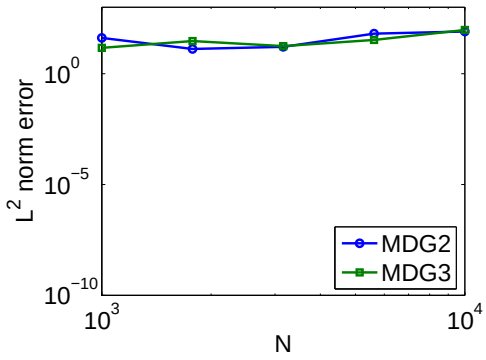
(a) $1e^{-2} \sim 1e^{6}$



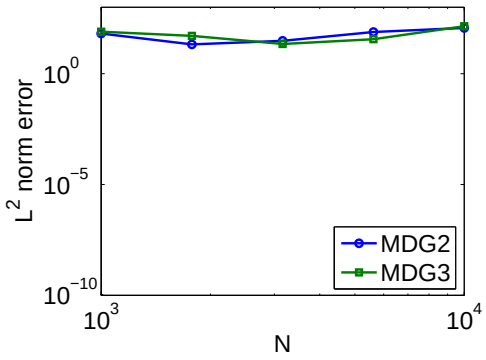
(b) $1e^{-3} \sim 1e^{5}$



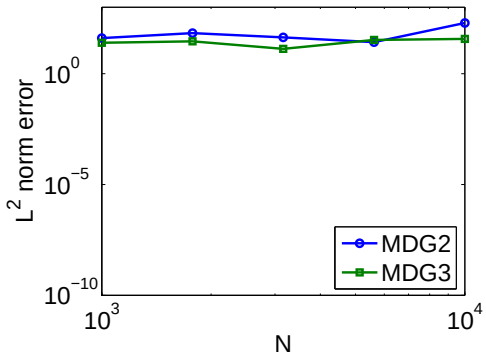
(c) $1e^{-4} \sim 1e^{4}$



(d) $1e^{-5} \sim 1e^{3}$



(e) $1e^{-6} \sim 1e^{2}$



(f) $1e^{-7} \sim 1e^{1}$

Figure 3.16: Comparative scaling error between MDGRAPE-2 and MDGRAPE-3.

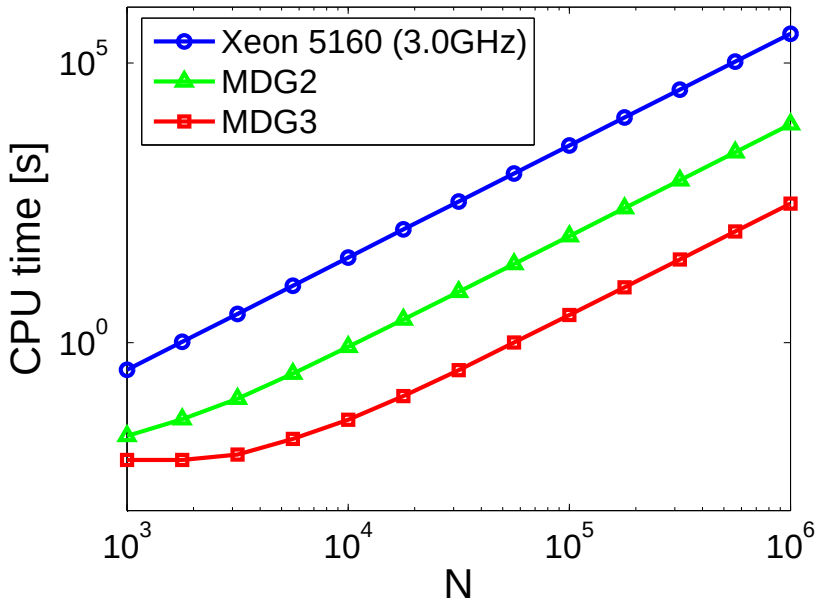


Figure 3.17: Acceleration using MDGRAPE-2 and MDGRAPE-3

times faster when compared with the host calculation and 25 time faster compared with the MDGRAPE-2. This means that MDGRAPE-3 imply much faster calculation than MDGRAPE-2 which leads me to use the new special-purpose computer.

Numerical accuracy is an important issue for any numerical simulation and engineering applications as well. Therefore, here I will check the accuracy of MDGRAPE-3 calculations compared with that of MDGRAPE-2. It is already observed in Fig. 3.17 that MDGRAPE-3 is faster compared with MDGRAPE-2. I must check the accuracy of present calculation before going to implement the actual calculation on it. Figure 3.18 represent the L^2 norm error as of Eq.(3.17) for Biot-Savart calculation compared between the old and new board. It is shown that both error are in the same order of magnitude and below 10^{-5}. On the one hand MDGRAPE-3 gives less error for small number of elements but it keeps the same order of magnitude compared to large numbers. On the other hand MDGRAPE-2 gives large error for large number of elements but it keeps the same order of magnitude with small number of elements. This negligible discrepency

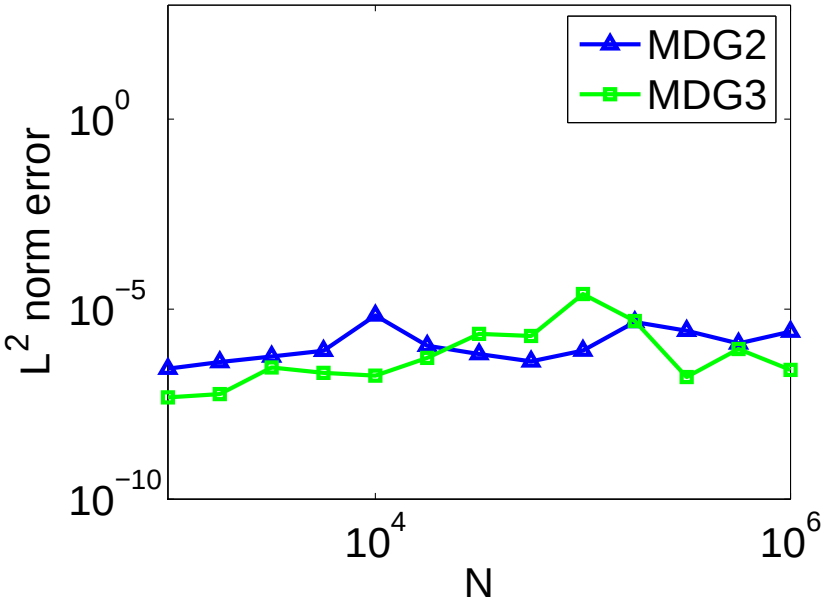


Figure 3.18: Accuracy of MDGRAPE-2 and MDGRAPE-3

may caused by slightly different hardware specifications and the floating point operations between them.

3.8 FMM on MDGRAPE

In this section the calculation algorithm of the simultaneous use of the FMM and MDGRAPE has been discussed. The most time consuming parts of FMM calculations are the multipole to local(M2L) translations and the direct summation. A breif mathematical formulations has been in chapter 2. For details mathematical formulations of FMM and its implementation on vortex methods calculation can be found in Cheng et al. [1999], and Yokota et al. [2007].

Since the MDGRAPE chips can only handle two types of calculations. The Coulomb potential (Eq. 3.15) and the Coulomb force (Eq. 3.16). The direct form of the Biot-Savart equation (2.3) and the stretching term (2.6) can be calculated by using a combination of (3.15) and (3.16), but the mutipole and local expansions and their translations are impossible to calculate. Therefore, the MDGRAPE

can only be used for the final step of the FMM where it calculates the direct interaction of particles.

The inefficiency of the above method resides in the fact that only one of the two hot spots of the FMM is calculated on MDGRAPE. It is possible to calculate both hot spots on the FMM if one can convert the multipole to local translation into a direct calculation. This requires the use of two independent methods, the Poisson integral method by Anderson [1992] and the pseudo-particle method by Makino [1999]. Instead of calculating the multipole and local expansions at the center of the boxes, these methods calculate the physical properties of interest at quadrature points placed on a spherical shell surrounding the boxes. In contrast to the original FMM, which uses 5 different equations for the expansions and translations, these methods use only 2. One for the multipole translation

$$Q_i = \sum_{j=1}^{N} q_i \sum_{n=0}^{p} \frac{2n+1}{K} \left(\frac{\rho_j}{r_s}\right)^n P_n\left(\cos\gamma_{ij}\right) \tag{3.18}$$

and another for local translations

$$Q_i = \sum_{j=1}^{N} q_i \sum_{n=0}^{p} \frac{2n+1}{K} \left(\frac{r_s}{\rho_j}\right)^{n+1} P_n\left(\cos\gamma_{ij}\right). \tag{3.19}$$

The notations still follow that of Cheng et al. [1999], but additional variables have been introduced. Q and q are the physical property of interest, which are the potential for Anderson's method, and circulation for Makino's method. Q represents the physical property after the translation and q represents the one before. K is the number of quadrature points on the sphere surrounding the box, so the index i runs from 1 to K, and r_s is the radius of this sphere. γ_{ij} is the angle between the position vector of source and target particles. Given that $\mathbf{x}_i = (r_i, \theta_i, \phi_i)$ and $\mathbf{x}_j = (\rho_j, \alpha_j, \beta_j)$, γ_{ij} can be written as

$$\cos\gamma_{ij} = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{r_i \rho_j} \tag{3.20}$$

Next, I will give a brief explanation of how these two methods are actually used. Suppose that the Biot-Savart equation has been calculated here. The flow of calculation is shown in Fig. 3.19. In this procedure the potential equation

$$\mathbf{\Phi}_i = \sum_{j=1}^{N} \frac{\boldsymbol{\gamma}_j}{4\pi r_{ij}} \tag{3.21}$$

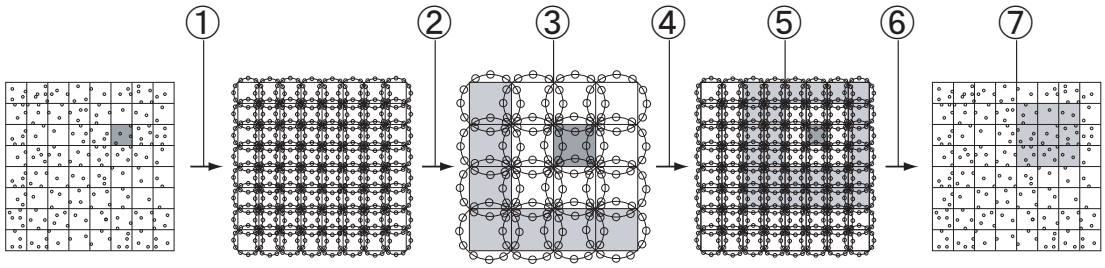


Figure 3.19: Flow of FMM calculation without multipoles

is also calculated.

Step.① Eq. (3.21) is calculated for the quadrature points on a large sphere, having a radius twice as that of the circumscribing sphere. Then a system of equations has been solved to calculate the circulation of the quadrature points on the circumscribing sphere.

Step.② Makino's method (Eq. 3.18) is used to translate the circulation onto the larger spheres.

Step.③ Eq. (3.21) is calculated for the quadrature points on the non-neighboring spheres. In the corresponding figure only the adjacent boxes (3×3) are defined as neighbors, which is different from the actual case. Since the interaction is calculated between the quadrature points on the circumscribing sphere instead of the multipole moments at the center of the box, the quadrature points become too close for a far field approximation. In the present method the definitioin of neighbors is expand to a larger region (5×5) to retain accuracy. Consequntly, the coarsest level in this calculation should be level 3 instead of 2.

Step.④ Anderson's method, Eq. (3.19) is used to translate the potential onto the smaller spheres.

Step.⑤ Eq. (3.21) is calculated for the quadrature points on the remaining non-neighboring spheres. The neighbor region is 5×5, so the number of source boxes can reach 875 per target. This increase is quite large and will slow down the method considerably.

Step.⑥ Solve a system of equations to determine the circulation of the quadrature points on a large sphere, having a radius twice as that of the circumscribing sphere. Then, calculate Eq. (2.3) to obtain the velocity of all particles in the corresponding box.

Step.⑦ Calculate the remaining induced velocity using Eq. (2.3) for all particles in the light grey box in the last figure.

Now, the two most time consuming steps ⑤ and ⑦ can both be calculated on the MDGRAPE. In this case the complexity of the calculation should remain $O(N)$.

3.9 Conclusions

In this chapter, several acceleration techniques have been proposed and developed a fast vortex method. Some critical issues have been solved to be used these techniques for vortex method calculations. It is found that the generation of a function table is essential for accurate MDGRAPE calculations. The accuracy and speed have been compared for two differenet special-purpose computers. It is observed that MDGRAPE-3 has been reduced compuation cost is 25 times compared with MDGRAPE-2 for the same calculation while the numerical accuracy retained same. The performance of MDGRAPE-2 and MDGRAPE-3 boards has been investigated for vortex method calculations and compared with the peak performance of both boards for Coulomb force calculation. The simultaneous use of the FMM and MDGRAPE have been proposed for further acceleration of the present vortex method calculations. The details analyses and its application to vortex rings collisions will be discussed in the following chapters 4 and 5.

Chapter 4

Fast Vortex Method Calculation using a Special-purpose Computer

4.1 Introduction

In this chapter, a mathematical formulation has been developed for the 3D vortex method calculation using a special-purpose computer MDGRAPE-2 that was originally designed for molecular dynamics simulations. A rigorous assessment of this hardware has been made for a few representative problems and compared the results with and without it. It is found that the generation of appropriate function tables, which are used to call libraries, embedded in MDGRAPE-2 is of primary importance in order to retain accuracy. The error arising from the approximation is evaluated by calculating a pair of vortex rings impinging to themselves.

The three critical issues regarding the implementation of the MDGRAPE-2 on vortex method have been solved. Due to the simple architecture of MDGRAPE-2, the efficient calculation of the Biot-Savart and Stretching equations have been performed. It is required to generate an optimum function table when the embedded libraries are called from the main routine. The cross product calculation which is not considered in the original command set must be handled in a proper manner, which is treated in some previous works, e.g., Yatsuyanagi et al.

[2003a,b], Elmegreen et al. [2002, 2004] and Sheel et al. [2007]. Due to hardware specifications force calculates partly with single precision, hence the minimization of the round-off error has been determined. Consequently, special attentions are necessary for the floating-point operations. These points are discussed one after another in the subsequent sections.

4.2 Mathematical Formulations

MDGRAPE-2 is a special-purpose hardware for the calculation of force or potential between point-mass or point-charge particles that was originally designed for molecular dynamics simulation. The calculation of interactions between particles as represented by potential and force are carried out in MDGRAPE-2. In case of calculating the potential,

$$\Phi_i = \sum_{j=1}^{N} b_{ij} g(w) = \sum_{j=1}^{N} b_{ij} g\left(a_{ij}\left(|\mathbf{r}_{ij}|^2 + \epsilon_{ij}^2\right)\right) \tag{4.1}$$

and the force calculations

$$\mathbf{f}_i = \sum_{j=1}^{N} b_{ij} g(w) \mathbf{r}_{ij} = \sum_{j=1}^{N} b_{ij} g\left(a_{ij}\left(|\mathbf{r}_{ij}|^2 + \epsilon_{ij}^2\right)\right) \mathbf{r}_{ij} \tag{4.2}$$

are treated similarly, where $g(w)$ is an arbitrary function equivalent to an intermolecular force, and a_{ij}, b_{ij}, ϵ_{ij} are arbitrary coefficients which are settled down for every model. To apply these libraries to the calculation of a vortex method, Biot-Savart law in Eq. (2.2) is expressed as follows.

$$\mathbf{u}_i = \sum_{j=1}^{N} B_j g\left(A_j\left(|\mathbf{r}_{ij}|^2 + \epsilon_{ij}^2\right)\right) \mathbf{r}_{ij} \tag{4.3}$$

where A_j, B_j are arbitrary constants. To implement Eqs. (2.3) and (2.4), it is required to take special treatment to the cross-product calculation in MDGRAPE-2 which is similar to Yatsuyanagi et al. [2003a,b] and Elmegreen et al. [2002, 2004] as follows.

The calculation of the cross product $\mathbf{r}_{ij} \times \gamma_j$ is considered.

$$\mathbf{r}_{ij} = (x_{ij},\ y_{ij},\ z_{ij})\,; \quad \gamma_j = \left(\gamma_j^x,\ \gamma_j^y,\ \gamma_j^z\right) \tag{4.4}$$

The total moment is

$$\sum_j \mathbf{r}_{ij} \times \gamma_j = \sum_j \left(y_{ij}\gamma_j^z - z_{ij}\gamma_j^y,\ \ z_{ij}\gamma_j^x - x_{ij}\gamma_j^z,\ \ x_{ij}\gamma_j^y - y_{ij}\gamma_j^x \right) \tag{4.5}$$

The cross product is considered with regard to the sum of the following three tensor products.

$$\sum_j \mathbf{r}_{ij} \otimes \gamma_j^x = \sum_j \left(x_{ij}\gamma_j^x,\ \ y_{ij}\gamma_j^x,\ \ z_{ij}\gamma_j^x \right) \tag{4.6}$$

$$\sum_j \mathbf{r}_{ij} \otimes \gamma_j^y = \sum_j \left(x_{ij}\gamma_j^y,\ \ y_{ij}\gamma_j^y,\ \ z_{ij}\gamma_j^y \right) \tag{4.7}$$

$$\sum_j \mathbf{r}_{ij} \otimes \gamma_j^z = \sum_j \left(x_{ij}\gamma_j^z,\ \ y_{ij}\gamma_j^z,\ \ z_{ij}\gamma_j^z \right) \tag{4.8}$$

It should be noted that only non-diagonal components of Eqs. (4.6) to (4.8) are required for the calculation of a moment of Eq. (4.5).

From Eq. (2.3), the Biot-Savart law reduces according to Eq. (4.3) as

$$\mathbf{u}_i = -\frac{1}{4\pi} \sum_j \frac{1}{\sigma_j^3} g_1(w) \left(\mathbf{r}_{ij} \times \gamma_j \right) \tag{4.9}$$

where $g_1(w)$ is a function for velocity calculation defined as

$$g_1(w) = \frac{w + 5/2}{(w+1)^{5/2}}; \qquad w = \left(|\mathbf{r}_{ij}|/\sigma_j \right)^2 \tag{4.10}$$

The stretching term appearing in Eq. (2.6) can be divided into two parts. The first and second terms of the right hand side of Eq. (2.6) have been defined as **stx** and **tx**, respectively. The mathematical details are as follows.

First Part:

$$\begin{aligned} \mathbf{stx} &= -\frac{1}{4\pi} \sum_j \frac{|\mathbf{r}_{ij}|^2 + (5/2)\sigma_j^2}{\left(|\mathbf{r}_{ij}|^2 + \sigma_j^2 \right)^{5/2}} \gamma_i \times \gamma_j \\ &= -\frac{1}{4\pi} \sum_j g_1(w) \left(\gamma_i \times \gamma_j \right) \frac{1}{\sigma_j^3} \\ &= -\frac{1}{4\pi} \sum_j g_1(w) \left(\gamma_i^y\gamma_j^z - \gamma_i^z\gamma_j^y,\ \ \gamma_i^z\gamma_j^x - \gamma_i^x\gamma_j^z,\ \ \gamma_i^x\gamma_j^y - \gamma_i^y\gamma_j^x \right) \frac{1}{\sigma_j^3} \end{aligned} \tag{4.11}$$

where $g_1(w)$ is defined as above Eq. (4.10) which is summarized in Table 4.1.

Table 4.1: Function and coefficients

$g(w)\,[w=(r_{ij}/\sigma_j)^2]$	A_j	B_j	ϵ_{ij}
$g_1(w)=\frac{w+5/2}{(w+1)^{5/2}}$	$\frac{1}{\sigma_j^2}$	$\frac{\gamma_j}{\sigma_j^3}$	0
$g_2(w)=\frac{w+7/2}{(w+1)^{7/2}}$	$\frac{1}{\sigma_j^2}$	$\frac{\gamma_j}{\sigma_j^5}$	0

Second Part:

$$\begin{aligned}\mathbf{tx} &= \frac{3}{4\pi}\sum_j \frac{|\mathbf{r}_{ij}|^2+(7/2)\sigma_j^2}{\left(|\mathbf{r}_{ij}|^2+\sigma_j^2\right)^{7/2}}(\gamma_i\cdot\mathbf{r}_{ij})\,(\mathbf{r}_{ij}\times\gamma_j)\\ &= \frac{3}{4\pi}\sum_j g_2(w)\,(\gamma_i\cdot\mathbf{r}_{ij})\,(\mathbf{r}_{ij}\times\gamma_j)\,\frac{1}{\sigma_j^5}\\ &= \frac{3}{4\pi}\sum_j g_2(w)\,(\gamma_i\cdot\mathbf{r}_{ij})\left(y_{ij}\gamma_j^z-z_{ij}\gamma_j^y,\;\; z_{ij}\gamma_j^x-x_{ij}\gamma_j^z,\;\; x_{ij}\gamma_j^y-y_{ij}\gamma_j^x\right)\frac{1}{\sigma_j^5}\end{aligned} \tag{4.12}$$

where $g_2(w)$ is a function for the stretching term calculation summarized in Table 4.1 and defined as

$$g_2(w)=\frac{w+7/2}{(w+1)^{7/2}} \tag{4.13}$$

From Eq. (4.12) we can write for γ_j^x as follows.

$$\begin{aligned}\mathbf{I}_i &= \sum_j g_2(w)\,(\gamma_i\cdot\mathbf{r}_{ij})\left(\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij}\\ &= (\gamma_i\cdot\mathbf{r}_i)\sum_j g_2(w)\left(\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij}-\Big\{\gamma_i^x\sum_j g_2(w)\left(x_j\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij}\\ &\quad+\gamma_i^y\sum_j g_2(w)\left(y_j\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij}+\gamma_i^z\sum_j g_2(w)\left(z_j\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij}\Big\}\\ &= (\gamma_i\cdot\mathbf{r}_i)\,\mathbf{S}-(\gamma_i^x\mathbf{T1}+\gamma_i^y\mathbf{T2}+\gamma_i^z\mathbf{T3})\end{aligned} \tag{4.14}$$

with

$$\begin{aligned}
\mathbf{S} &= \sum_j g_2(w)\,\frac{\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij} \\
\mathbf{T1} &= \sum_j g_2(w)\,\frac{x_j\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij} \\
\mathbf{T2} &= \sum_j g_2(w)\,\frac{y_j\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij} \\
\mathbf{T3} &= \sum_j g_2(w)\,\frac{z_j\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij}
\end{aligned}$$

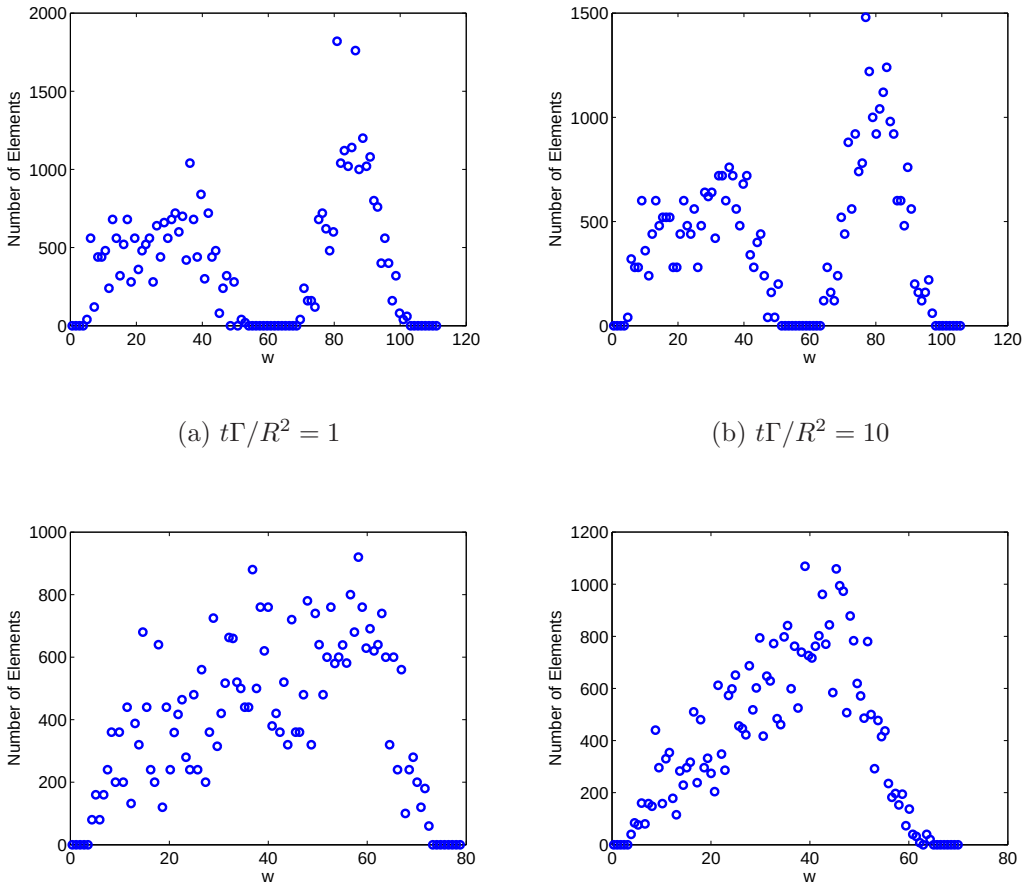
Similar formulations can be readily obtained for the other two components, γ_j^y and γ_j^z.

To solve Eq. (4.9), it is necessary to call the library in Eq. (4.2), embedded in MDGRAPE-2, three times to compute the cross-product in a three-dimensional problem. Similarly, to solve Eq. (4.11), it is required to call the library in Eq. (4.1) three times. Finally, for the Eq. (4.12), it is required to call library in Eq. (4.2) for $\left(\gamma_j^x, \gamma_j^y, \gamma_j^z\right) \times 4$, i.e. twelve times according to Eq. (4.14). The values shown in Table 1 were substituted for function $g(w)$ and constants A_j, B_j, ϵ_{ij} are defined for each calculation. Further details fo mathematical expressions can be found in Appendix B.

The generation of function tables has been discussed in chapter 3. An optimum range of a function table will be determined for a pair of vortex rings calculations by considering the computational domain of vortex elements distributed there in. The typical distribution of vortex elements has been observed to determine the computational domain before optimized a function table as follows.

4.3 Typical Distribution of Vortex Elements

In order to examine the validity and the applicability of the present method, the three different configurations of two identical colliding vortex rings have been invesitgated; the details of the computational algorithms and results are discussed in section 4.6. The typical distribution of vortex elements for each configuration



(a) $t\Gamma/R^2 = 1$

(b) $t\Gamma/R^2 = 10$

(c) $t\Gamma/R^2 = 50$

(d) $t\Gamma/R^2 = 100$

Figure 4.1: Typical distributions of vortex elements (head-on)

has been observed carefully. Once finished, the determination of an optimal range of a function table will be determined hereafter.

Head-on Collisions

Figure 4.1 shows the histogram of the distance between vortex elements scattering over the computational domain in head-on collisions. The results are extracted from the data at non-dimensional times $t\Gamma/R^2 = 1$, 10, 50 and 100. In the figures, the abscissa w carries the same meaning as that defined in the function

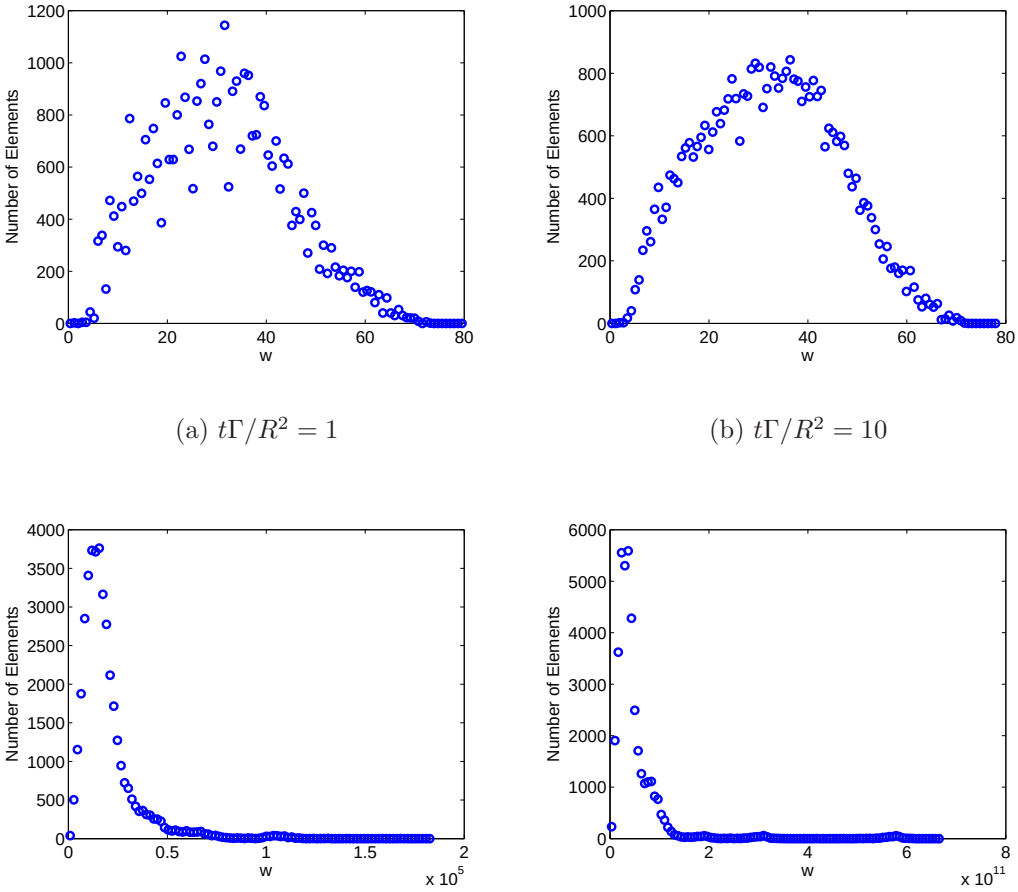
table, cf. also in Eq. (4.10). It is seen from Fig. 4.1(a) that the minimum and maximum distance in computational domain at time $t\Gamma/R^2 = 1$ are 3.90 and 103.2, respectively. These values vary as a function of time which are 2.94 and 60.37, respectively at $t\Gamma/R^2 = 100$. The density of vortex elements is changed as a function of time and observed here carefully. It is clearly seen that the total number of distributed particles have been changed. In Fig. 4.1(a), it was 2000 and in Fig. 4.1(d) it was 1200 and is different at different time stages. Because the initial distribution of particles in a vortex rings and the particle distributions after collisions between two rings are not the same. Consequently computational domain has been changed.

Offset Collisions

Figure 4.2 shows the histogram of the distance between vortex elements scattering over the computational domain in offset collisions. The results are extracted from the data at non-dimensional times $t\Gamma/R^2 = 1$, 10, 50 and 100. In the figures, the abscissa w carries the same meaning as of head-on collisions. It is seen from Fig. 4.2(a) that the minimum and maximum distance in computational domain at time $t\Gamma/R^2 = 1$ are 3.5 and 66.45, respectively. These values vary as a function of time. Consequently, it is observed from Fig. 4.2(d) that they are 3.1×10^9 and 1.35×10^{11} at $t\Gamma/R^2 = 100$. It is also clearly seen that the total number of distributed particles have been changed. In Fig. 4.2(a), it was 1200 and in Fig. 4.2(d) it was 6000 which has been varied at different time stages. This indicates that the density of vortex elements is not same for different times and also for different configurations of vortex rings. It can be shown that the particle distrubutions of head-on and offset collisions are not same is expected.

Inclined Collisions

Figure 4.3 shows the histogram of the distance between vortex elements scattering over the computational domain in inclined collisions. The results are extracted from the data at non-dimensional times $t\Gamma/R^2 = 1$, 10, 50 and 100. In the figures, the abscissa w carries the same meaning as of previous two configurations. It is seen from Fig. 4.3(a) that the minimum and maximum distance in computational



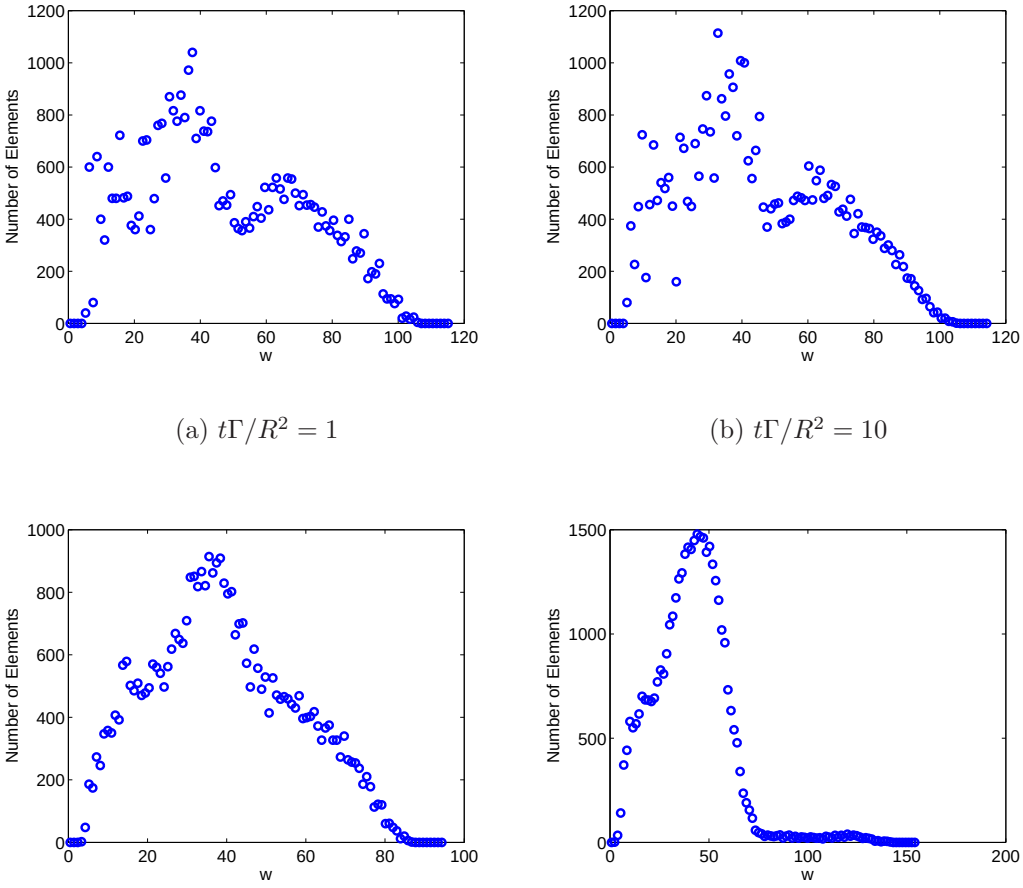
(a) $t\Gamma/R^2 = 1$

(b) $t\Gamma/R^2 = 10$

(c) $t\Gamma/R^2 = 50$

(d) $t\Gamma/R^2 = 100$

Figure 4.2: Typical distributions of vortex elements (offset)



(a) $t\Gamma/R^2 = 1$

(b) $t\Gamma/R^2 = 10$

(c) $t\Gamma/R^2 = 50$

(d) $t\Gamma/R^2 = 100$

Figure 4.3: Typical distributions of vortex elements (inclined)

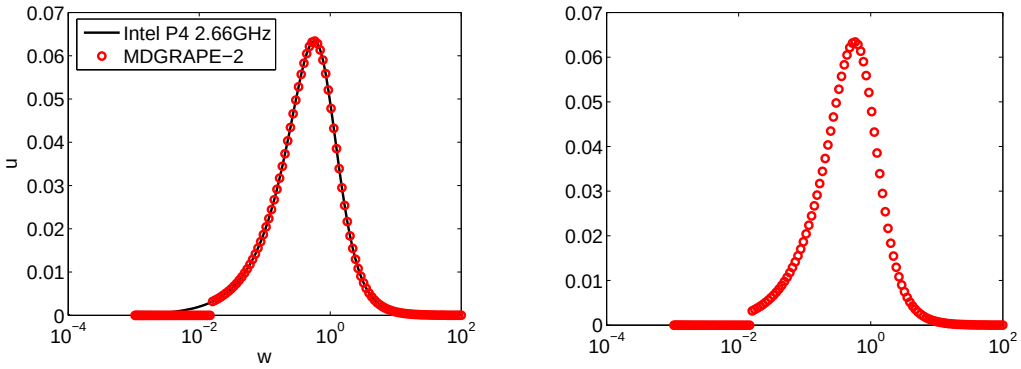
domain at time $t\Gamma/R^2 = 1$ are 3.76 and 99.37, respectively. These values vary as a function of time. Consequently, it is followed from Fig. 4.3(d) that they are 2.156 and 127.2 at $t\Gamma/R^2 = 100$. It is also clearly seen that the total number of distributed particles have been changed. In Fig. 4.3(a), it was 1200 and in Fig. 4.3(d) it was 1500 and is different at different time stages. Same observations have been found here as of head-on and offset collisions.

In summary, an optimal range of a function table will be determined for the collision of two identical vortex rings considered the computational domain discussed above. The range of a function table must contains all vortex elements of colliding vortex rings for longer calculations.

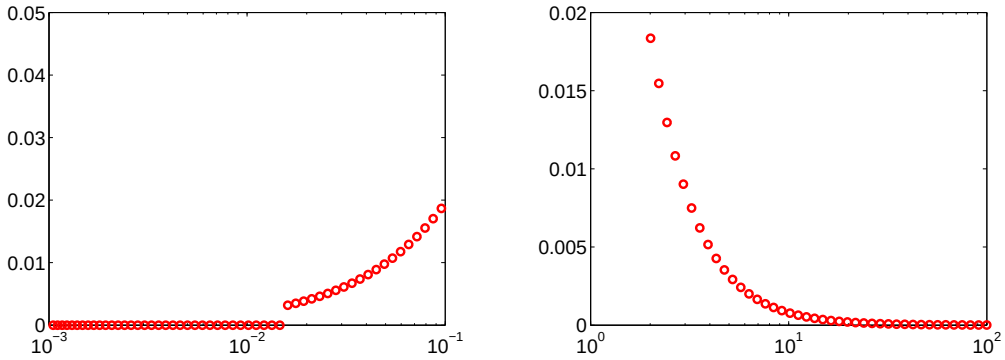
4.4 Optimum Range of a Function Table

The table range must be adjusted for individual problem being calculated. It is necessary to generate a new function table for a new problem. The preparation of the function table operation takes only a few minutes, which does not affect the performance of the entire calculations. To simulate high Reynolds number flows using vortex method, it is required to incorporate large number of vortex elements for small-scale structures. There is no connection between the number of elements and the range of function table. The range is the key factor in maintaining the accuracy and the single precision calculation of MDGRAPE-2 board, which does not have any influence on the calculation of high Reynolds number flows.

Based on the above observations, it appears that the table ranges have been defined carefully prior to use MDGRAPE-2. Various function tables are generated with different finite input ranges in the host calculation. The output ranges of those function tables have been checked for the accuracy of MDGRAPE-2 and found that all ranges are not satisfied with the computational domain, which was determined in previous section (4.3). Figure 4.4 shows the finite range of a function table of one of the input domains is $2^{-12} \leq w \leq 2^{20}$ where $w = |r_{ij}|/\sigma_j$. Figure 4.4(a) shows a typical velocity distribution on a logarithmic scale, calculated from Eq. (2.3), with and without the use of MDGRAPE-2, where a single source particle is positioned at the origin and 1000 target particles are distributed from 10^{-4} to 10^2. Intel P4(2.66GHz) and MDGRAPE-2 stand for calculations



(a) $2^{-12} \leq w \leq 2^{20}$ (b)



(c) 2^{-12} (d) 2^{20}

Figure 4.4: Range of a function table.

without the use of MDGRAPE-2, and with the use of MDGRAPE-2, respectively. The velocity becomes zero when $|\mathbf{r}_{ij}|/\sigma_j$ falls outside of the range of the table. Otherwise, the results of the Biot-Savart calculation on Intel P4(2.66GHz) and MDGRAPE-2 match each other. It can be observed from figs. 4.4(c) and 4.4(d) that the minimum and maximum output ranges are 2^{-12} and 2^{20}, respectively. These ranges are different for different input domains which will be discussed next.

Figure 4.5 shows a typical velocity distribution on a logarithmic scale, calculated from Eq. (2.3), with and without the use of MDGRAPE-2, for six different input ranges. Consequently, six different output ranges have been observed. Figs. 4.5(a) to 4.5(f) represent the ranges and scaling errors of six different types of input ranges of function tables where the x-axis stands for the range of the function $g(w)$ and the y-axis for induced velocity $\mathbf{u}$. The output ranges of these figures are mentioned in each. It is observed that the finite range of $2^{-12} \leq w \leq 2^{20}$ has been satisfied by the computational domain of the present entire calculations to obtain significant accuracy.

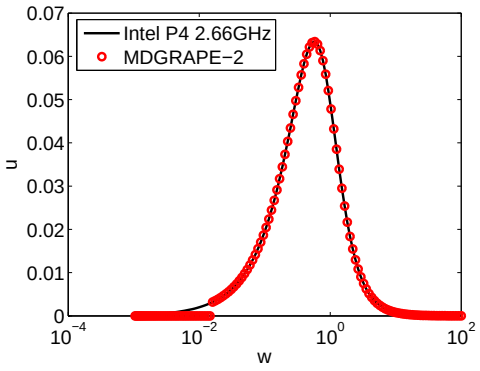
4.5 Convection Error

The issue here is to evaluate the overall error caused by the use of MDGRAPE-2. The final result of the calculation has been compared in terms of the position of the particles for the present test cases. The convection error is defined as the difference in the position of the same particles between the host and MDGRAPE-2 for the same time steps. Here convection error is defined as the distance δ is as follows.

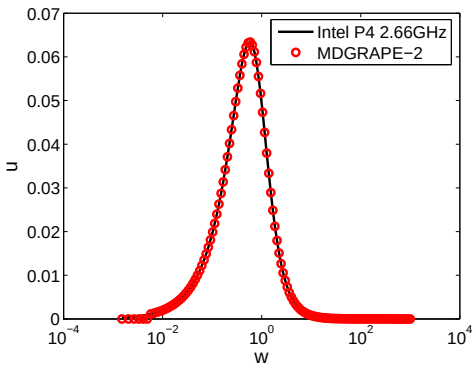
$$\delta\,(\text{difference}) = \sqrt{(x_{host} - x_{md})^2 + (y_{host} - y_{md})^2 + (z_{host} - z_{md})^2} \qquad (4.15)$$

where the suffices *md* and *host* represent with and without the use of MDGRAPE-2, respectively.

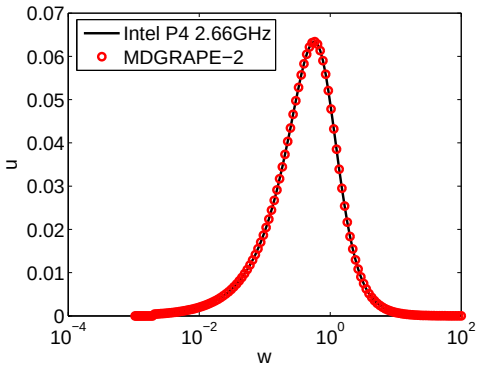
The three different configurations of a pair of identical vortex rings have been choosen. The main objective to validate the special-purpose computer and its accuracy for vortex method calculations. The calculation alogorithm of vortex



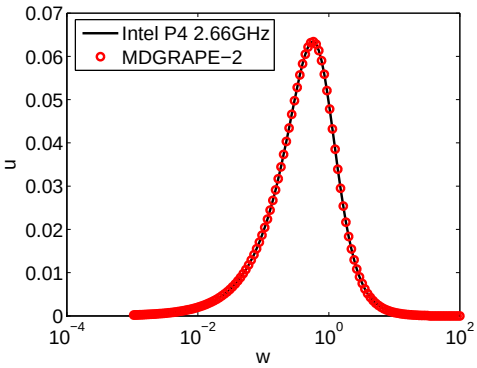
(a) $2^{-12} \leq w \leq 2^{20}$



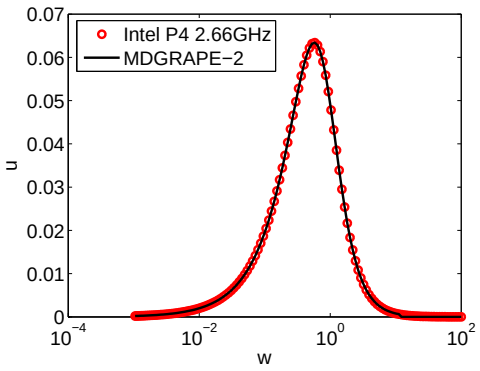
(b) $2^{-16} \leq w \leq 2^{16}$



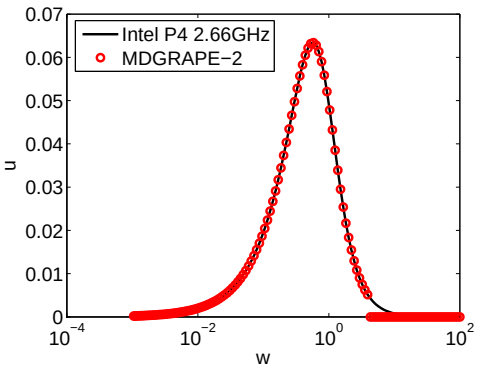
(c) $2^{-18} \leq w \leq 2^{14}$



(d) $2^{-22} \leq w \leq 2^{10}$



(e) $2^{-24} \leq w \leq 2^{8}$



(f) $2^{-28} \leq w \leq 2^{4}$

Figure 4.5: Scaling error for function table in six different ranges. ∘ with MDGRAPE-2; — without MDGRAPE-2

rings collisions and its physical properties have been simulated using MDGRAPE-2 after investigated the convection errors.

Head-on Collisions

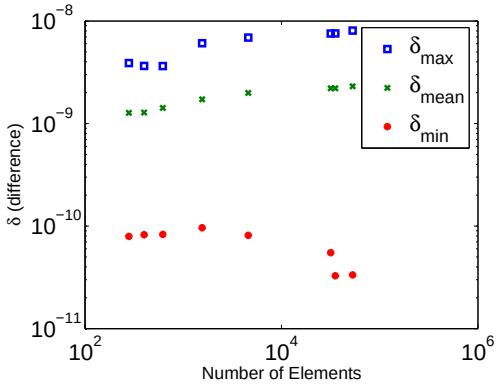
Figure 4.6 represents the convection error of head-on collisions of two identical vortex rings calculated by Eq. (4.15) for different number of iterations. In all figures, δ_{max}, δ_{mean} and δ_{min} stand for the maximum, average and minimum values of δ evaluated from all of the particles in the domain. It is observed that δ slightly increases for a larger number of elements. This means that MDGRAPE-2 induces certain errors for calculations with a large number of elements but the *error* stays within a finite range. This error is caused by the influence of self-induced velocities from the initial position of the particles.

The errors vary for different iterations. In fig. 4.6(a) at initial time iteration, all errors are below 10^{-8}. These values increase for time iterations 5 and 10. In fig. 4.6(b), it was below 10^{-5} at time iteration 5 and it was below 10^{-4} in fig. 4.6(c).

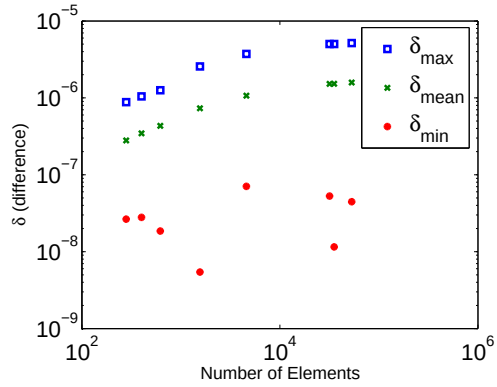
Offset Collisions

Figure 4.7 represents the convection error of offset collisions of two identical vortex rings calculated by Eq. (4.15) for different numbers and iterations. In all figures, δ_{max}, δ_{mean} and δ_{min} stand for the maximum, average and minimum values of δ evaluated from all of the particles in the domain. It is observed that δ slightly decreases for a larger number of elements. This means that MDGRAPE-2 reduces certain errors for these calculations with a large number of elements but the *error* remains constant within a finite range. This improvement is caused by the influence of self-induced velocities from the initial position of the particles.

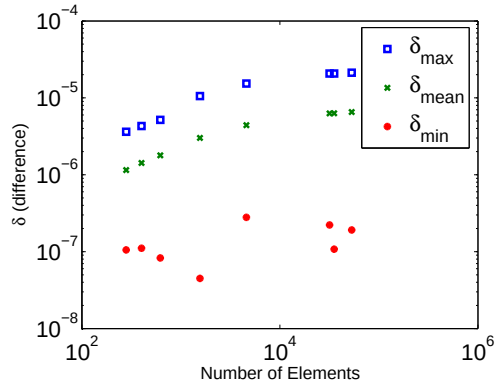
The errors vary for different iterations. In fig. 4.7(a) at initial time iteration, all errors are below 10^{-6}. These values increase for time iterations 5 and 10. In figs. 4.7(b) and 4.7(c), it was below 10^{-4} in both time iterations 5 and 10.



(a) *Time step* = 1

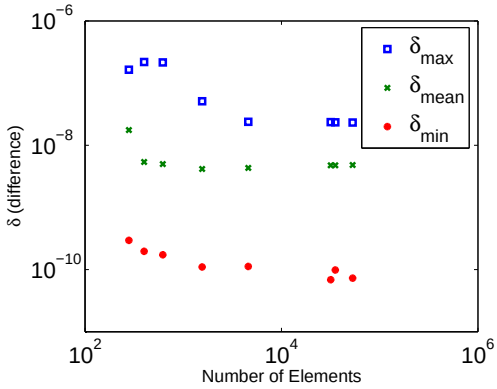


(b) *Time step* = 5

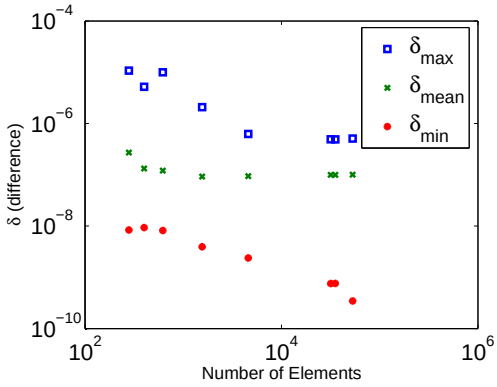


(c) *Time step* = 10

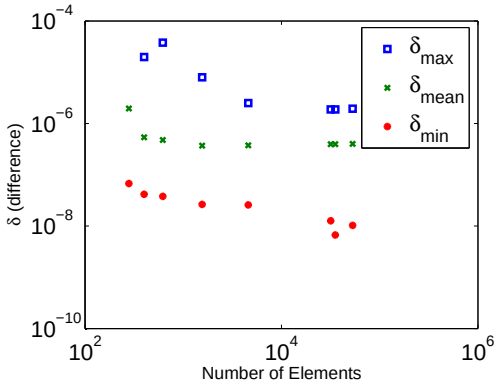
Figure 4.6: Convection error for MDGRAPE-2 for different numbers of elements(head-on). Here, □ –maximum value of all δ, × –average value of all δ, ∗ –minimum value of all δ



(a) *Time step* = 1

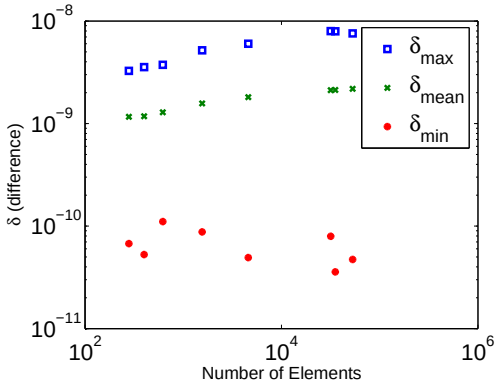


(b) *Time step* = 5

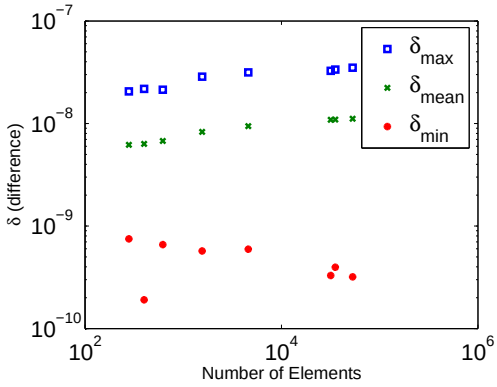


(c) *Time step* = 10

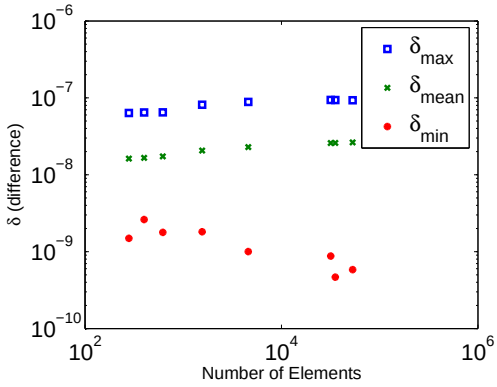
Figure 4.7: Convection error (offset).



(a) *Time step* = 1



(b) *Time step* = 5



(c) *Time step* = 10

Figure 4.8: Convection error(inclined).

Table 4.2: Initial Conditions

Position	R	r	σ	Γ_0	N	Re_Γ	S
Head-on	0.5	0.05	0.05R	1	$2 \times 314 \times 91$	2500	1
Offset	0.5	0.03	0.05R	1	$2 \times 104 \times 7$	250	R/2
Inclined	1	0.05	0.065	1	$2 \times 502 \times 61$	400	2.7

Inclined Collisions

Figure 4.8 represents the convection error of inclined collisions of two identical vortex rings calculated by Eq. (4.15) for different number of iterations and number of particles. The legends stand for same meaning according to previous two cases. The overall behavior are similar as of head-on collisions except the quantitative behavior. This error is caused by the influence of self-induced velocities from the initial position of the particles.

The errors vary for different iterations. In fig. 4.8(a) at initial time iteration, all errors are below 10^{-8}. These values increase for time iterations 5 and 10. In fig. 4.8(b), it was below 10^{-7} at time iteration 5 and it was below 10^{-5} in fig. 4.8(c).

In summary, the overall errors induced from MDGRAPE-2 calculations remain as of expectated level within the finite ranges. These results indicate that the use of MDGRAPE-2 does not induce any extra error which resist the vortex method calculations while it accelerates the calculations significantly.

4.6 Application

4.6.1 Computational Algorithm

In order to check the validity and the application of the prpoposed acceleration method, three different configurations of colliding vortex rings have been simulated as test case. One of the three configurations will be used for other methods to investigate in details. The initial conditions are details as follows and summarized in table 4.2.

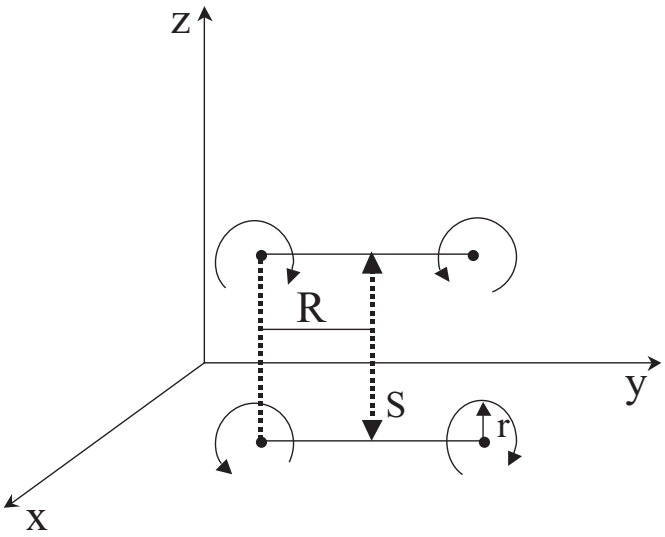


Figure 4.9: Initial condition for the computation of the collision of two vortex rings. Here R—radius of ring, $\mathbf{r}$—radius of cross-section, S—distance between two rings

The head-on collision has been considered as the first test case to compare the result with and without MDGRAPE-2. In the entire simulations it is assumed the initial radius of vortex ring is $R = 0.5$, cross section radius $r = 0.05$ (Fig. 4.9), Reynolds number $Re_\Gamma = 2500$, core radius $\sigma = 0.05R$, circulation of ring $\Gamma = 1$ and number of elements $N = 57148$ where the number of cross-section in the circumference direction is 314, while the number of elements in each cross-section is 91. All elements are evenly distributed.

Second, the offset collisions is applied according to Chatelain et al. [2003]. In this case, the initial radius of the vortex rings is $R = 0.5$, the cross-section radius is $r = 0.03$ (Fig. 4.10), where core radius $\sigma = 0.05R$, and Reynolds number $Re_\Gamma = 250$ have been used, the number of particles is N=1456 where the number of cross-section in the circumference direction is 104, while the number of elements in each cross-section is 7. All elements are evenly distributed. The initial rings are placed at a distance of $R/2$ apart in the z direction, offset by R along the y axis and they move in opposite directions along the z-axis.

Finally, the inclined collisions is considered according to Winckelmans and Leonard [1993]. Here I assumed that the initial radius of the vortex rings is $R = 1$

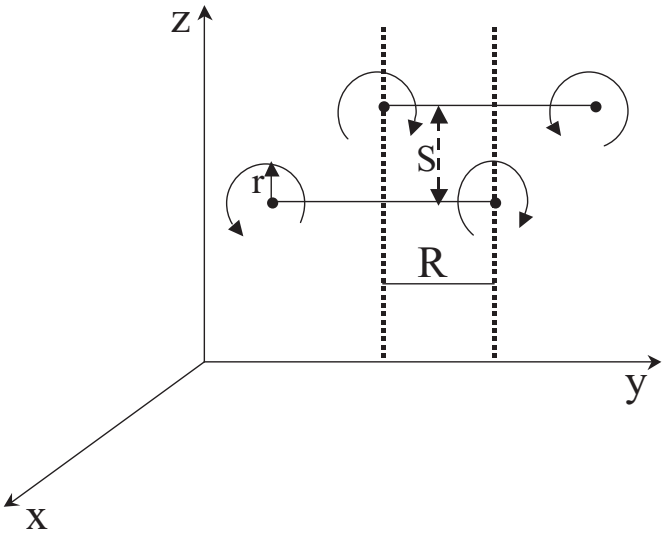


Figure 4.10: Initial condition for the computation of the collision of two vortex rings. Here R—radius of ring, $\mathbf{r}$—radius of cross-section, S—distance between two rings

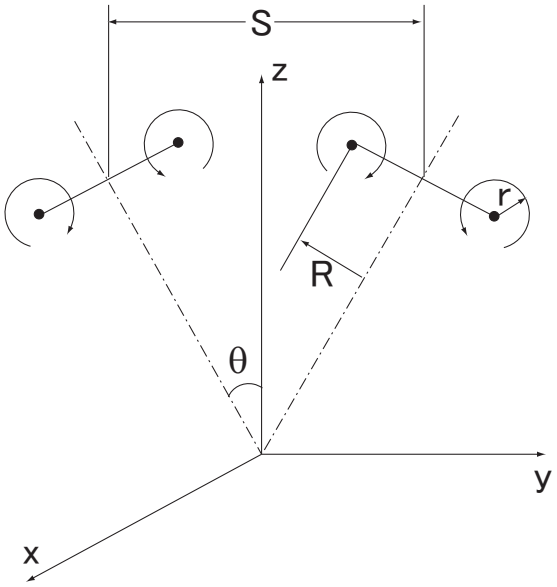


Figure 4.11: Initial condition for the computation of the collision of two vortex rings. Here R—radius of ring, $\mathbf{r}$—radius of cross-section, S—distance between two rings, θ—inclined angle

while the cross-section radius $r = 0.05$, see Fig. 4.11. The Reynolds number based on the ring circulation is $Re_\Gamma = 400$, and the core radius $\sigma = 0.065$. The rings are inclined at an angle $\theta = 15°$ relative to the z-axis. The total number of elements used for the preliminary calculation was $N = 6 \times 10^4$, with the number of cross sections in the circumference direction being 502, while 61 elements were distributed in each cross-section. All elements were evenly distributed.

In all calculations, the viscous diffusion was calculated using the core-spreading method developed by Leonard [1980]. For convection of the particles, the second order accurate Adams-Bashforth method was used in the calculation of time advances (Moin [2001]).

The kinetic energy K or E and enstrophy Ω or ζ are evaluated from the particle positions and strengths according to Winckelmans and Leonard [1993], are defined as follows.

$$E(or K) = \frac{1}{16\pi} \sum_{i,j} \left[\frac{2\left(\gamma_i \cdot \gamma_j\right)}{\left(\mathbf{r}_{ij}^2 + \sigma_j^2\right)^{1/2}} + \frac{\left(\mathbf{r}_{ij} \cdot \gamma_i\right)\left(\mathbf{r}_{ij} \cdot \gamma_j\right) - \mathbf{r}_{ij}^2\left(\gamma_i \cdot \gamma_j\right)}{\left(\mathbf{r}_{ij}^2 + \sigma_j^2\right)^{3/2}} \right] \tag{4.16}$$

and

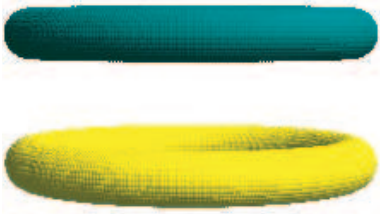
$$\begin{aligned} \Omega(or \zeta) \quad = \quad & \frac{1}{4\pi} \sum_{i,j} [\frac{5\sigma_j^4 - \mathbf{r}_{ij}^2\left(\mathbf{r}_{ij}^2 + 3.5\sigma_j^2\right)}{\left(\mathbf{r}_{ij}^2 + \sigma_j^2\right)^{7/2}} \left(\gamma_i \cdot \gamma_j\right) \\ & + 3 \frac{\left(\mathbf{r}_{ij}^2\left(\mathbf{r}_{ij}^2 + 4.5\sigma_j^2\right) + 3.5\sigma_j^4 i\right)\sigma_j^2}{\left(\mathbf{r}_{ij}^2 + \sigma_j^2\right)^{9/2}} \left(\mathbf{r}_{ij} \cdot \gamma_i\right)\left(\mathbf{r}_{ij} \cdot \gamma_j\right)] \end{aligned} \tag{4.17}$$

4.6.2 Numerical Results

In the following the numerical results have been discussed in details. The MDGRAPE-2 has been used to develop a fast vortex method. Three different configurations of colliding vortex rings have been tested and simulated. The numerical results are discussed separately as follows.

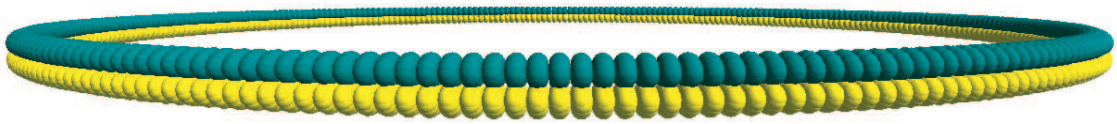
Head-on Collisions

Figure 4.12 shows snap-shots of head-on collision of the vortex rings at various time stages . The initial setup in colliding ring simulations consists of two coaxial vortex rings initially placed at upward and downward of the plane $z = 0$. In Fig.



(a) $t\Gamma/R^2 = 1$

(b) $t\Gamma/R^2 = 3$



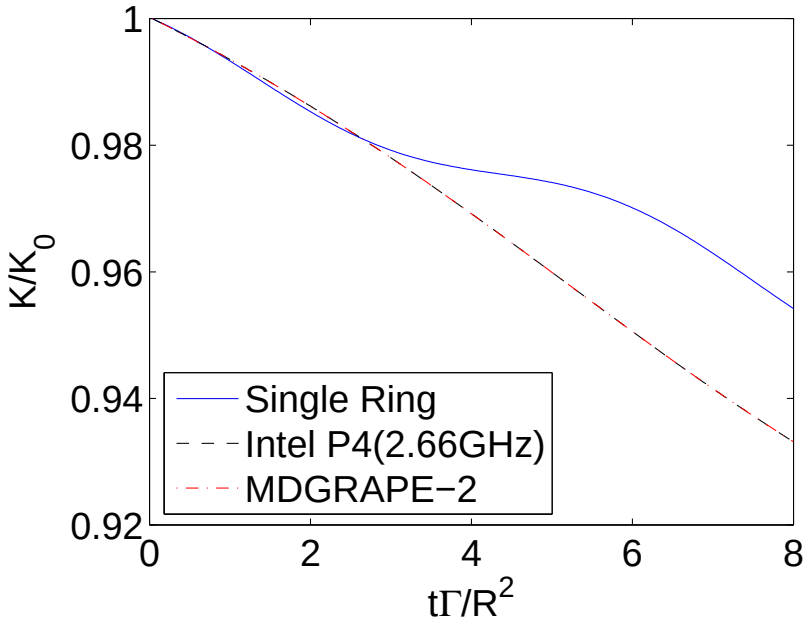
(c) $t\Gamma/R^2 = 8$

Figure 4.12: Snapshots of vortex elements for different time (head-on).

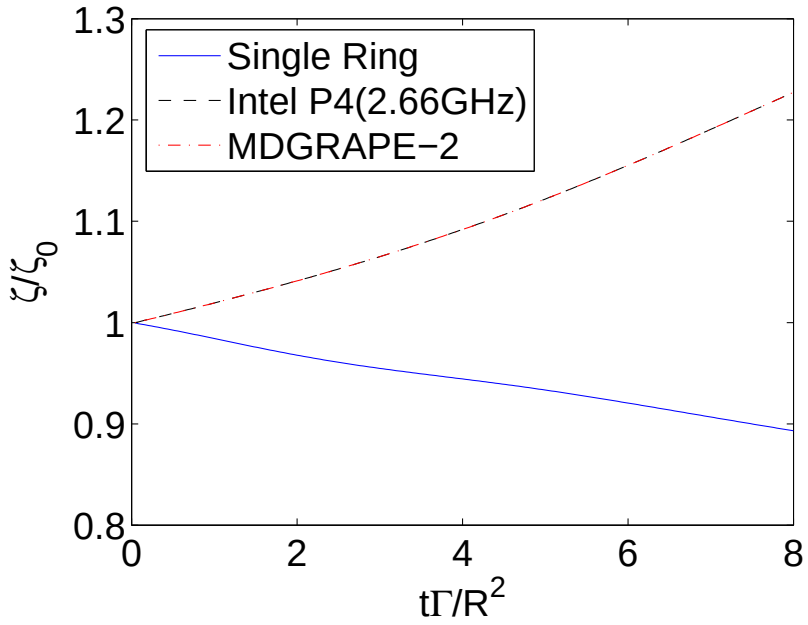
4.12(a), the rings are initially placed at a distance of 0.4 apart in the z-direction. As the rings approach to each other, they are stretched in the radial direction by the velocity induced by the other ring. In Fig. 4.12(b), the impingement and the increase in ring diameter and the subsequent merger of the rings are clearly depicted. By the end of the simulation, in Fig. 4.12(c), early signs of reconnection can be observed. After the impingement from the initial position, the rings expand outwards rapidly, which leads to the formation of small-scale structures yet to further time steps in our calculation.

The evolution of the kinetic energy and enstrophy for the various cases is plotted in Fig. 4.13. These quantities are evaluated from the particle positions and strengths, as defined in Winckelmans and Leonard [1993]. In the figure 4.13(a), K represents the kinetic energy against the different time span, Host and MDGRAPE-2 stand for the calculation without and with the use of MDGRAPE-2. It can be easily observed that the results of host and MDGRAPE-2 calculations coincide with each other. In the present calculations the flow is incompressible and unbounded, so there are no (physical) kinetic energy sources. The kinetic energy can be dissipated by both viscosity and mathematical scheme. Enstrophy is dissipated by viscosity, but can also be generated by stretching of vortex lines. Figure 4.13(a) also shows that for the single ring the kinetic energy experiences a steady monotonic decrease, amounting to a 4.5% drop by the end of the calculation. For the colliding ring simulations, the kinetic energy decreases in the initial stages at the same rate as for the single ring, but then the rate of decline increases.

In the figure 4.13(b), ζ represents the enstrophy against the different time span. In this case also the results of host and MDGRAPE-2 calculations coincide with each other. Enstrophy is dissipated by viscosity, but can also be generated by stretching of vortex lines. Figure 4.13(b) also shows that for the single ring the decay of the kinetic energy is accompanied by a similar decay in the enstrophy, with a 13% drop at the end of the calculation. On the other hand, the enstrophy increases in the colliding ring calculations, indicating significant vortex stretching. The similar explanations can be found in Mansfield et al. [1999] work. The present calculated results agreed with Mansfield work up to certain time (here 8) even though all calculation conditions are not exactly the same.



(a) Kinetic Energy



(b) Enstrophy

Figure 4.13: Kinetic energy and enstrophy of head-on collisions.

Offset Collisions

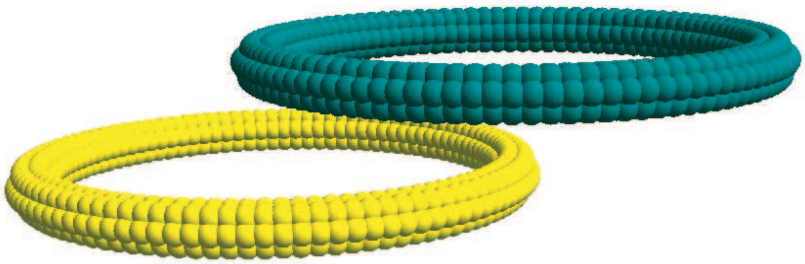
Figure 4.14 shows the snapshots of the position of vortex elements of offset collisions in different times. Initially rings are apart from each other in opposit direction along the z-axis (fig. 4.14(a)). At time progress, the both rings are approach each other and stretched. It can be seen that the stretched has been started in Fig. 4.14(b). The strong colliding behavior has been observed in Fig. 4.14(c) and represents the reconnection process according to Chatelain et al. [2003]. To obtain the similar results compared with Chatelain et al. [2003] work, furher modifications are needed.

Figure 4.15 represents the kinetic energy and enstrophy, respectively according to head-on collisions. The qualitative result has good agreement with referred work upto $t\Gamma/R^2 = 1.6$ but does not match the quantitative results as because the definitions and numerical schemes are different from our calculations. In this case also the results of MDGRAPE-2 differed from Host compared with that of head-on collision. This is also due to the difference in particle distribution between the two cases. It is highly considered to improve the quantitative results by increasing the number of elements in further research.

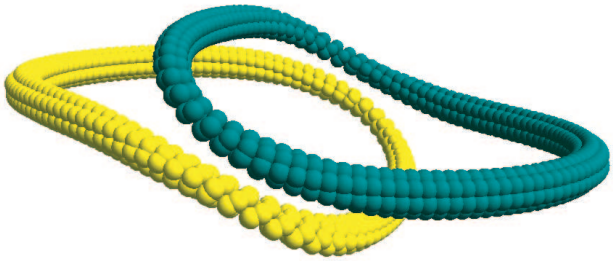
Inclined Collisions

Figure 4.16 shows the snapshots of vortex elements of two colliding inclined vortex rings at various time stages. The initial setup in colliding ring simulations consists of two identical vortex rings initially inclined at an angle $\theta = 15°$. In Fig. 4.16(a), the rings are initially placed at a non-dimensional distance of $s = 2.7$ in the z-direction. Each vortex ring approaches by self-induced velocity from this initial stage. At $t^* = 3$ in Fig. 4.16(b), where $t^* = t\Gamma/R^2$ with Γ being the initial circulation, the first impact occurs and the two vortex rings are stretched and deformed. As time progresses, considerable differences appear in each stage. At $t^* = 8$, the arced-shape structure is formed and the downward stretch is strong, cf. Fig. 4.16(c).

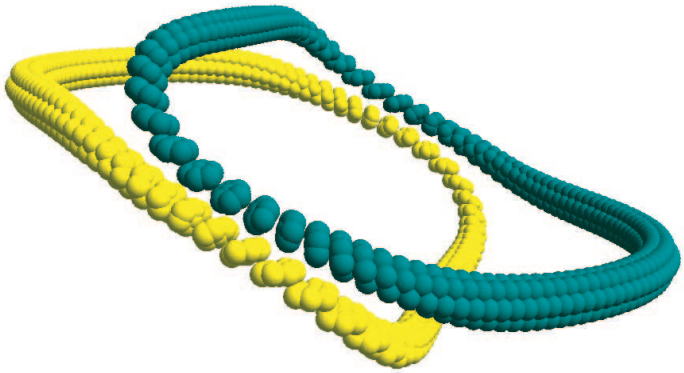
The evolution of the kinetic energy of host and MDGRAPE-2 compared with Winckelmans work are shown in Fig. 4.17(a). In the present calculations the flow is incompressible and unbounded, so there are physically no kinetic energy



(a) $t\Gamma/R^2 = 0$

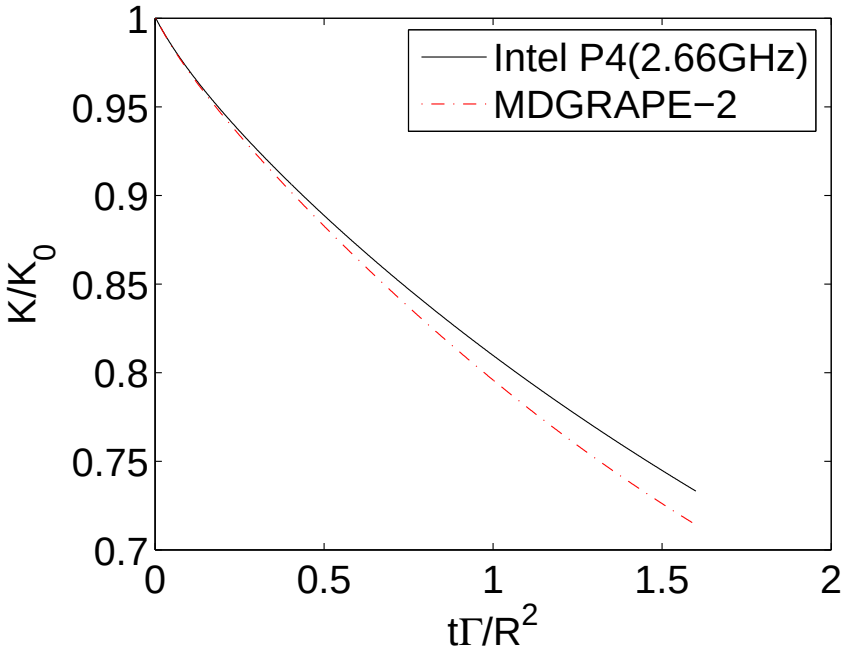


(b) $t\Gamma/R^2 = 0.8$

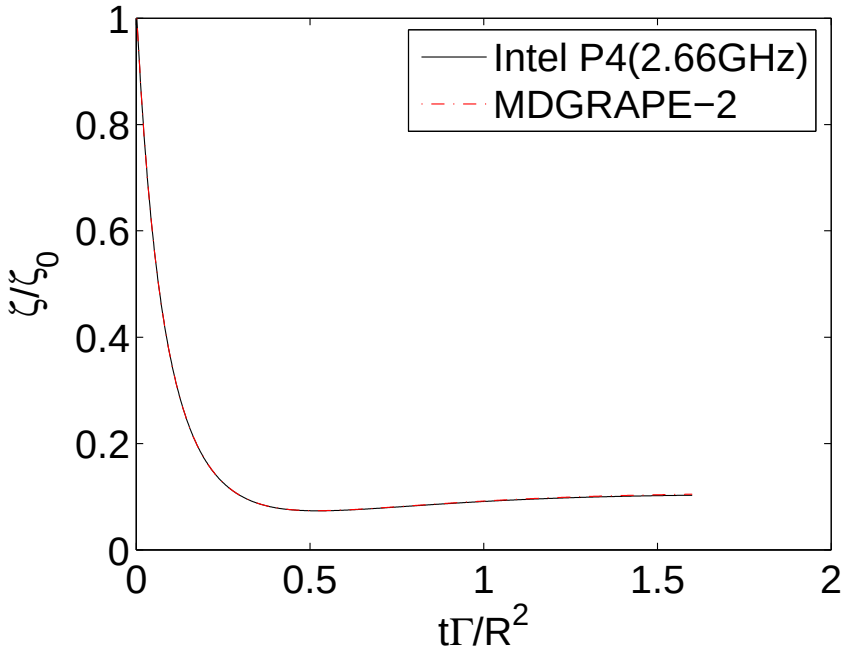


(c) $t\Gamma/R^2 = 1.6$

Figure 4.14: Snapshots of vortex elements for different time (offset).

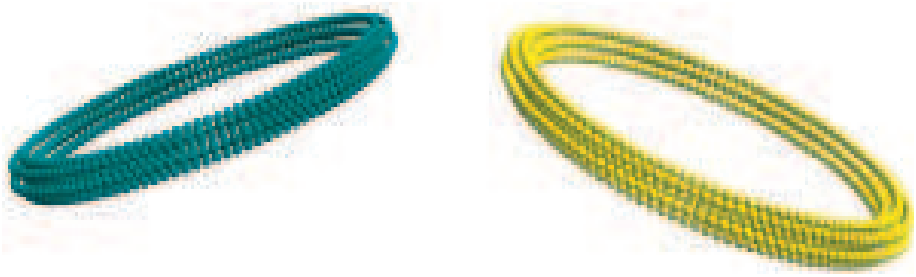


(a) Kinetic Energy

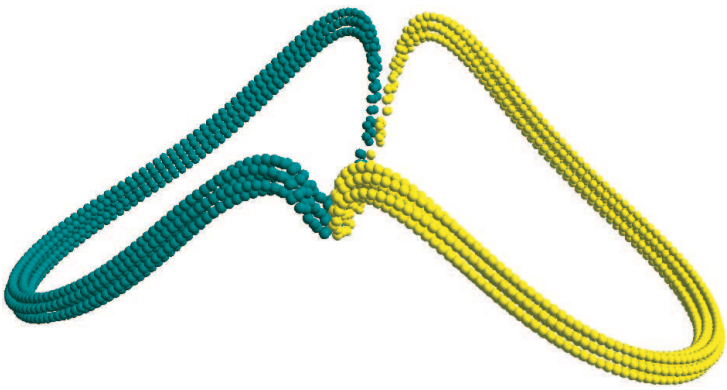


(b) Enstrophy

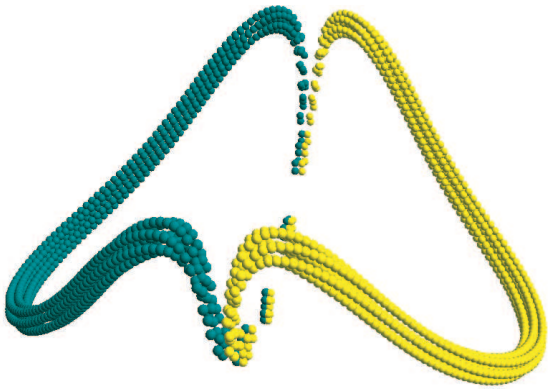
Figure 4.15: Kinetic energy and enstrophy of offset collisions.



(a) $t\Gamma/R^2 = 1$

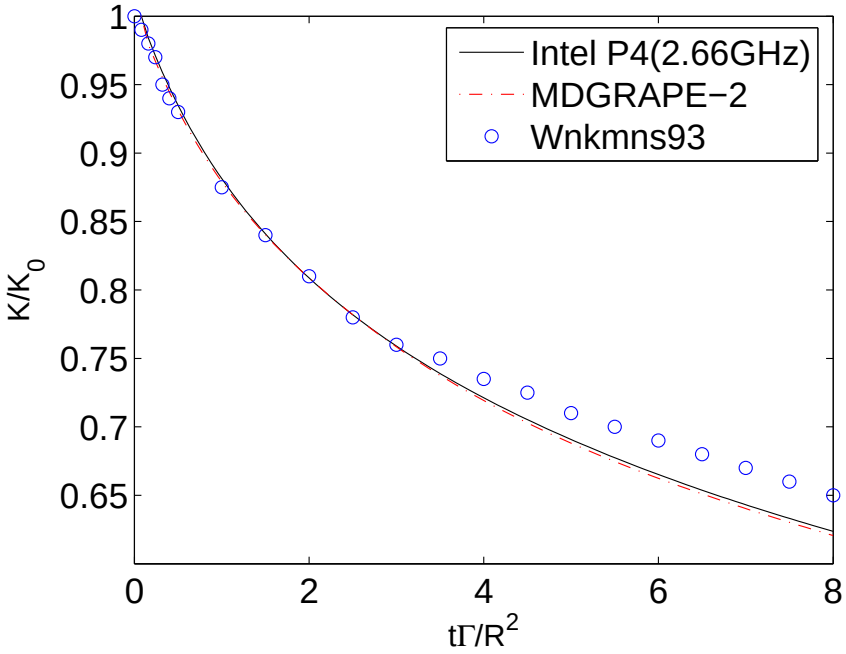


(b) $t\Gamma/R^2 = 3$

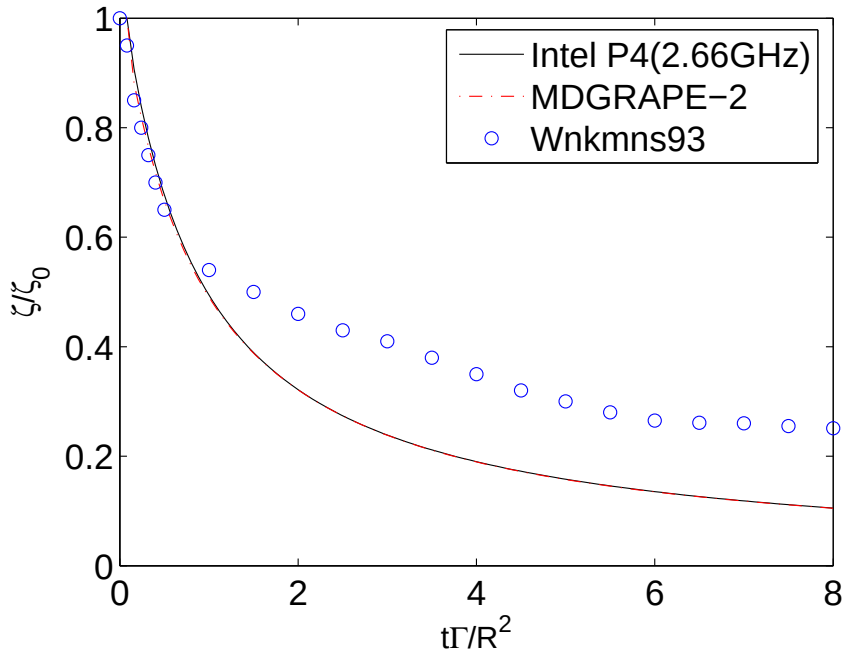


(c) $t\Gamma/R^2 = 8$

Figure 4.16: Snapshots of vortex elements for different time (inclined).



(a) Kinetic Energy



(b) Enstrophy

Figure 4.17: Time series of kinetic energy and enstrophy compared with Winckelmans work.

sources. The kinetic energy can be dissipated by both viscosity and numerical errors. From the comparison between the results obtained with the various time steps, it has been observed that there is no significant difference between host calculation and MDGRAPE-2. It is shown in Fig. 4.17(a) for the time step $\Delta t = 0.08$, it is easily observed that the agreement with the existing data of Winckelmans and Leonard [1993] is also satisfactory.

On the other hand, the slight difference in the decay of enstrophy, Fig. 4.17(b), is observed between the present computation and that by Winckelmans and Leonard [1993], though this is due to the difference in the treatment of viscous diffusion schemes and has nothing to do with the accuracy of MDGRAPE-2. Totsuka and Obi [2005, 2007] have also observed a similar tendency in the computation of two-dimensional homogeneous isotropic turbulence where the decay of enstrophy is subject to the choice of diffision approximation. Nevertheless, the main target of this article is to discuss the issues related to the use of MDGRAPE-2 in combination with vortex method calculation and accuracy of different viscous diffusion schemes does not have any influence on the assessment and the accuracy of MDGRAPE-2.

4.7 Conclusions

A special-purpose computer MDGRAPE-2 for N-body simulations was applied to the calculation of the vortex method. A mathematical formulation has been developed for 3D vortex method using this special-purpose computer. The calculation cost has been reduced significantly for present calculations. The definition of the function table range plays an essential role to achieve satisfactory accuracy in MDGRAPE-2. The acceptable accuracy has been achieved by investigating for three different configurations of impinging vortex rings. The results have good agreement when compared with the host calculations and referenced work. Further acceleration can be achieved with use of MDGRAPE-3 and the simultaneous use of FMM with MDGRAPE-2 and MDGRAPE-3 are discussed in chapter 5.

Chapter 5

The Study of Colliding Vortex Rings using a Special-purpose Computer and FMM

5.1 Introduction

The main focus of this chapter is to implement the fast algorithms on special-purpose processors. The first implementation of fast algorithms on the GRAPE (Sugimoto et al. [1990]) architecture was presented by Makino [1991], and showed a 30-50 times increase in computational speed compared to the treecode without GRAPE. The implementation of the fast multipole method (FMM) on MDGRAPE-2 was presented by Chau et al. [2002a,b] and similar results were obtained. The application of P^2M^2 tree code on MDGRAPE-2 was presented by Kawai et al. [2004] and accelerates the calculation by a factor of 20-200 compared with conventional PCs. The above mentioned implementations have been applied to astrophysical problems and MD simulations.

In this study, the possibility of further acceleration will be investigated with the simultaneous use of the FMM by Cheng et al. [1999] using special-purpose hardware; MDGRAPE-3 (Narumi et al. [2006]). Some issues regarding the optimum level of the FMM, and the use of pseudo-particle methods (Makino [1999]) are addressed. The collision of two inclined vortex rings at high Reynolds numbers has been simulated in this calculation to check the validity and applicability

of new method. The dynamics of the colliding vortex rings have been studied and the computation time has been reduced by a factor of 2000 compared to a direct calculation on Xeon 5160 (3.0GHz).

At first the various forms of FMM and their performance on MDGRAPE-3 have been discussed. Then this method is applied to the vortex method calculation of colliding vortex rings. The effect of spatial and temporal resolution at high Reynolds numbers is investigated by comparing the energy spectrum and decay rate of the kinetic energy and enstrophy. The reconnection of the vortex rings was clearly observed, and the discretization error became nearly negligible for the calculation using 10^7 elements.

5.2 FMM on MDGRAPE-3

The section focuses on the simultaneous use of the FMM and MDGRAPE-3. The details mathematical formulations and calculation algorithm are discussed in chapters 2 and 3, respectively. The optimization techniques and its performance on the calculation of two colliding inclined vortex rings will be discussed below.

In the present calculations, the FMM by Cheng et al. [1999] has been used. The most time consuming parts of the FMM are the multipole to local(M2L) translation and the direct calculation. The balance between these two steps is dependent on the level of box divisions. Dividing the particles into excessively small boxes will result in an enormous amount of multipole to local translations, whereas not dividing them enough would result in a large amount of direct calculation of neighboring particles. These two steps must be balanced by changing the level of box divisions according to the number of particles being calculated.

The mutipole and local expansions and their translations are impossible to calculate on the GRAPE architecture. Therefore, in a straightforward implementation of the FMM, MDGRAPE-3 can only be used for the final step of the FMM where the direct interaction of the particles is calculated.

The inefficiency of the above method rests in the fact that only one of the two hot-spots of the FMM is calculated on MDGRAPE-3. The hot-spots of FMM are discussed in appendix D. It is possible to calculate both hot-spots of the FMM on MDGRAPE-3 if the multipole to local translation is converted

into an N-body interaction. This requires the use of two independent methods - the Poisson integral method by Anderson [1992], and the pseudo-particle method by Makino [1999]. Instead of calculating the multipole and local expansions at the center of the boxes, these methods calculate the physical properties of interest at quadrature points(Hardin and Sloane [1996]) placed on a spherical shell surrounding the boxes. Details of the implementation of these algorithms on MDGRAPE-2 are given in Chau et al. [2002a,b] and on MDGRAPE-3 are in chapter 3.

In Anderson's method, the multipole expansion of the potential due to a clump of particles is effectively expressed in terms of the values of potentials on a sphere surrounding the particles. The potential outside the sphere is given by the surface integral on that sphere, which is then approximated by the sum over sampling points. This method, though elegant, appears to be rather indirect.

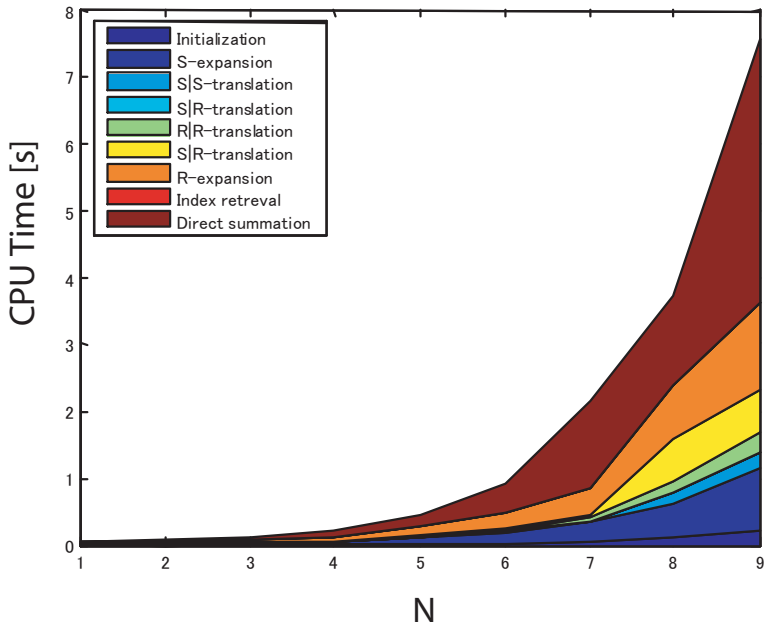
An alternative approach is to use multiple particles to represent the multipole expansion. The basic idea here is to place small number of pseudoparticles which reproduce the multipole expansion of the original physical particles (Makino [1999]).

The difference is that the value of potential is used in Anderson's method and the mass distribution itself is used in Makino's method. The advantage of Makino's method is that it can be applied in special-purpose computer compared with Anderson's method.

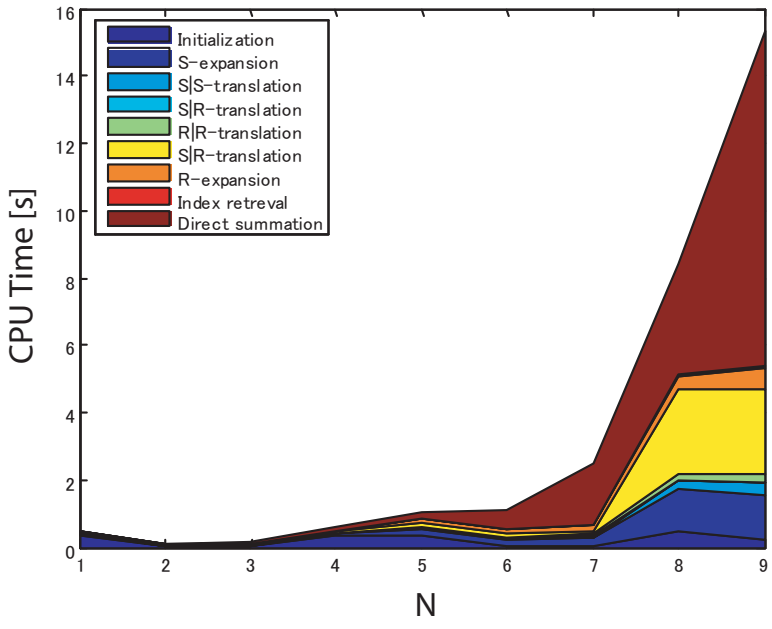
Some critical issues to be used FMM such as elapsed time, momentum effect of its accuracy and optimization of the calculation will be discussed in consequent sections before applying to vortex rings calculation.

5.2.1 Elapsed Time

In this section, the elapsed time has been investigated for every step of FMM and pseudo-particle multipole method(PP-FMM) calculation with and without the use of MDGRAPE-3. This analysis will leads to estimate and optimize the calculation cost and use of both techniques to implement in actual vortex method calculations.



(a) FMM-MDG3



(b) PP-FMM-MDG3

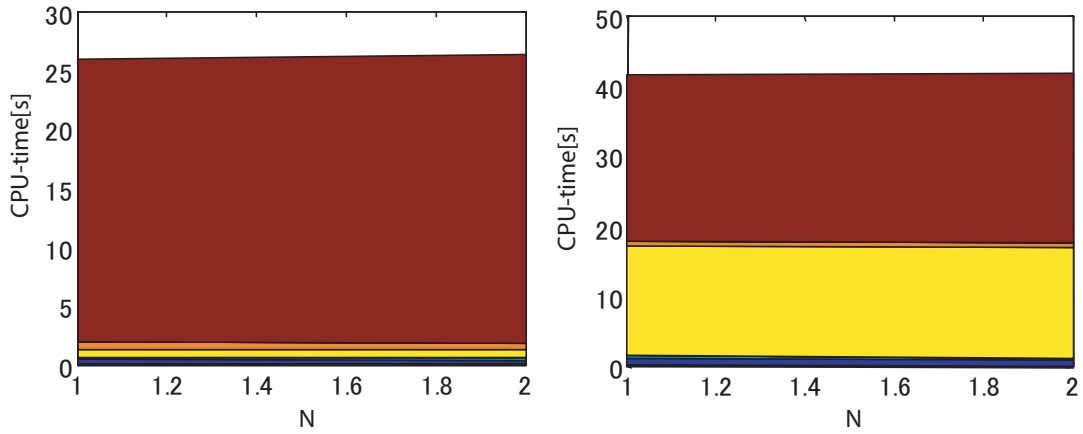
Figure 5.1: Elapsed time on MDGRAPE-3(Biot-Savart)

Figure 5.1 shows the elapsed time to implement the FMM and the pseudoparticle multipole method(PP-FMM) on MDGRAPE-3 for Biot-Savart calculation which calculates induced velocity from vorticity transport equation, Eq. (2.1). Elapsed time of different calculations represent in different colours. It can be easily seen that the brown colour consumed half of the total time in both cases which stands for direct calculation. This means it is required to reduce direct calculation time to get the maximum efficiency of these methods. In Fig. 5.1(b), multipole to local translation(M2L) consumed second largest time which contrary to pseudo-particle method. This may caused for the limitations of MDGRAPE-3 hardware specifications. Total time of PP-FMM calculation is larger than that of FMM calculation for the same calculation condition. This may caused for the neighbor region of pseudo-particles.

Figure 5.2 shows the elapsed time with and without the use of MDGRAPE-3. In Figure 5.2(a), it can be observed that direct calculation consumed most of the calculation time without using MDGRAPE-3. The pseudo-particle FMM balanced between the direct and M2L calculation (Fig. 5.2(b)) but the total elapsed time is larger than FMM only. Both of the direct and M2L calculation time have been reduced further and balanced when pseudo-paticle method has been implemented on MDGRAPE-3 shown in Figure 5.2(c). The total time also reduced in this case. It is clearly observed that the between direct and M2L calculation is performed with the use of PP-FMM with MDGRAPE-3. The simultaneous use of FMM and MSGRAPAE-3 reduced the total calculation time significantly. Consequently, the vortex method calculation can be further accelerated with the use of this acceleration technique.

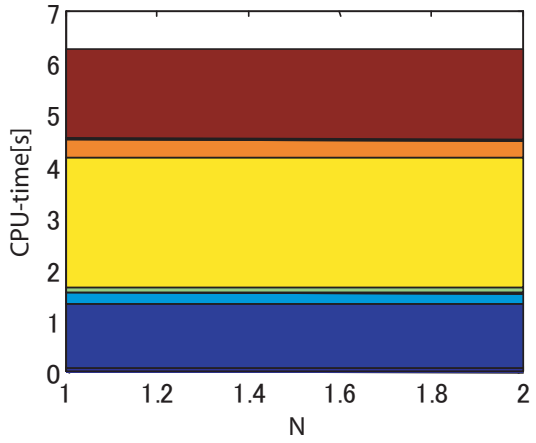
5.2.2 Momentum Effect on FMM Accuracy

Multipole moment is the coefficients of a series expansion of a potential due to continuous or discrete sources (e.g., particle distribution). A multipole moment usually involves powers (or inverse powers) of the distance to the origin, as well as some angular dependence. The order of multipole moment p (Eqs. 2.13 to 2.16) strongly affects the balance between the accuracy and speed of FMM calculation.



(a) FMM

(b) PP-FMM



(c) PP-FMM-MDG3

Figure 5.2: Elapse time with and without the use of pseudo-particle method. Here, brown→ Direct calculation, yellow→ M2L calculation, blue→ Initialization

In this section the accuracy of FMM calculation has been checked to determine the order of multipole moment for entire calculations.

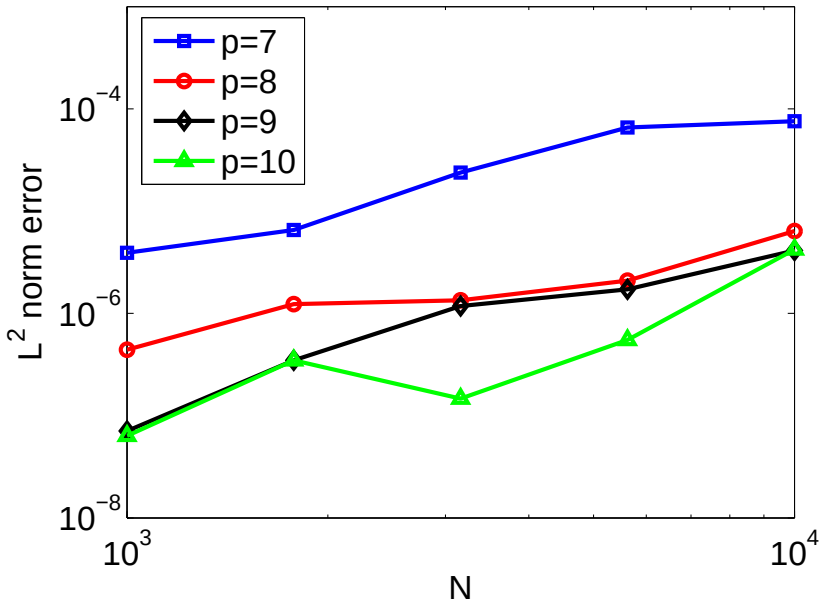
Biot-Savart and stretching term calculations have been performed separately by changing the order of moments from $p = 7$ to 10 as shown in Fig. 5.3. Figure 5.3(a) represent L^2 norm error for Biot-Savart calculation for different moments. For $p = 7$, initially the error is above 10^{-6} and it was increased for larger $N = 10^4$ which is below 10^{-4}. The error has been decreased when order of moment increases accordingly. It was below 10^{-7} initially for $p = 9$ and 10, also it is below 10^{-6} for $p = 10$.

Figure 5.3(b) represent L^2 norm error for stretching term calculation for different p. The similar behaviour as of Biot-Savart calculation has been observed. The overall error is increased nearly 10^{-6} compared to Biot-Savart one. This may influenced for other internal errors of stretching term but it will not have any affect when applied to actual calculations. Here the maximum number of particles $N = 10^4$ has been used. The overall results indicate that the error will be further increased for larger N. It has been confirmed that the error of FMM calculation is below 10^{-5} for the present entire calculations. Therefore, the order of moment is $p = 10$ is used in the present vortex method calculation by considering the above mentioned results.

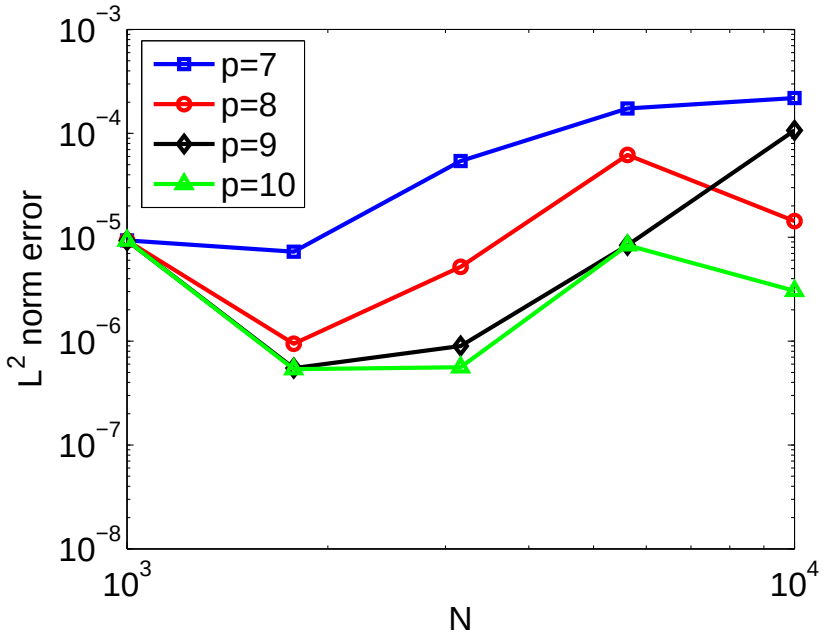
5.2.3 Optimization of FMM

An optimization problem is to find the variables that minimize or maximize the objective function while satisfying the constraints. In the present calculation, a level of box division has been determined to achieve minimum cost for large number elements calculation. This level is called the optimum level. The optimum level changes according to the total number of elements N being calculated.

Optimization of FMM calculations is an important issue to use it either with or without MDGRAPE-3. The order of multipole moments (here 'p' in Eqs. 2.13 to 2.16) strongly affects to balance the accuracy and speed of FMM calculation. On the one hand, higher order multipole moments gives the high accuracy but it requires very high computation cost. On the other hand, lower order multipole



(a) Biot-Savart



(b) Stretching

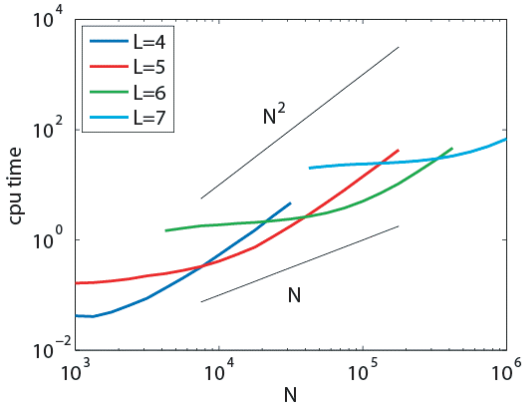
Figure 5.3: Accuracy of FMM at different moments

moments causes large amount of error(Figure 5.3) while it requires less computation cost. The most time consuming parts of FMM are the M2L and direct calculation. It is important to balance these two steps by determining optimum level of box divisions according to number of elements are being calculated. I will briefly discuss the optimization techniques of present FMM calculations as follows.

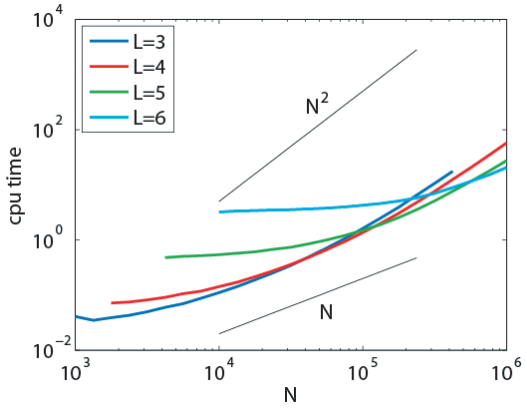
The optimum box level will be investigated for the FMM without MDGRAPE-3, FMM with MDGRAPE-3, and the pseudo-particle method on MDGRAPE-3. The cpu-time of the Biot-Savart calculation is plotted against the number of elements for different box levels in Fig. 5.4. L is the level of the oct-tree box division, where the original domain is divided into $2^L \times 2^L \times 2^L$ boxes. N is the number of elements.

As mentioned earlier, the workload of the multipole to local translation and direct calculation must be optimized. The calculation cost of the multipole to local translation is a function of L, but not N. Therefore, the FMM with higher L has a minimum calculation cost, which is not affected by N. This is clearly observed in all plots in Fig. 5.4. On the other hand, the direct calculation for neighboring particles is still a $O(N^2)$ operation. Therefore, keeping L constant and increasing N eventually results in a $O(N^2)$ curve, because the direct calculation will consume most of the computational time. This is also clearly observed in most plots in Fig. 5.4. The balance of the workload is achieved by simply selecting the box level L that gives the minimum CPU-time. The wavy nature of FMM plot in Fig. 5.7 is the result of connecting the minimum lines in Fig. 5.4(a).

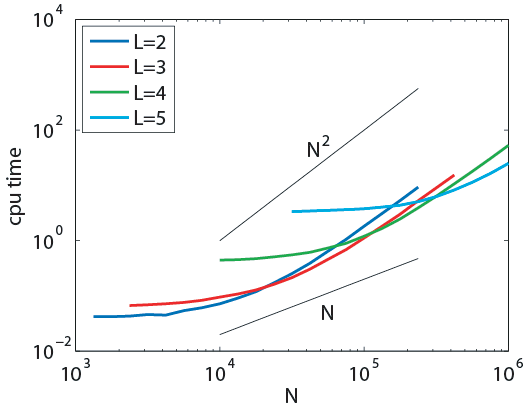
When the FMM is used with MDGRAPE-3, the balance changes significantly due to the acceleration of the direct calculation, as shown in Fig. 5.4(b). The points at which the lines crossover are not as clear as in Fig. 5.4(a). However, it will be shown in Fig. 5.7 that there is still a significant gain in speed when used the FMM on MDGRAPE-3. Furthermore, contrary to my expectations, the pseudo-particle method on MDGRAPE-3 did not require the use of larger L. The expectation was that, since the pseudo-particle method can calculate both the mutipole to local translation and direct calculation on MDGRAPE-3, the balance of the two should remain somewhat close to the origianl FMM without MDGRAPE-3. However, the results in Fig. 5.4(c) show that the optimum level



(a) FMM



(b) FMM-MDG3



(c) PPM-MDG3

Figure 5.4: Change in optimum box level for different methods

is even lower than that of the FMM on MDGRAPE-3. This is caused by the increase in the computational cost of the multipole to local translation when I use pseudo-particle methods. Since these methods require 216 pseudo-particles to represent a multipole moment of order 10, the calculation cost of the multipole to local translation is 216^2 per box, while the rotation-based FMM requires only 385 calculations per box.

In summary, the optimum level of box division of the FMM on the MDGRAPE-3 is approximately two levels lower than that of the FMM without the MDGRAPE-3 because only the direct summation is accelerated.

5.2.4 Test for CPU-time and Error

In this section, the maximum efficiency will be discussed of the proposed scheme considered the highest accuracy to be achieved. A comparative study among the different algorithms on different platforms has been observed to select the best approach for the actual calculation.

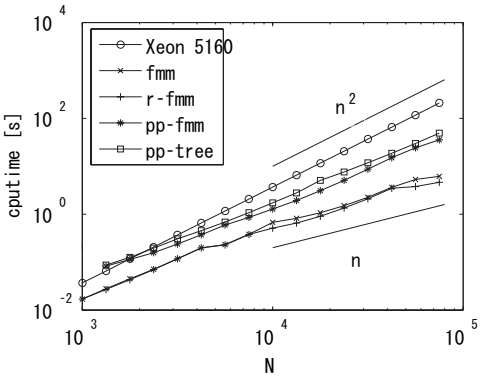
The calculation cost and accuracy are an important issue for any numerical simulation. In this calculation these two factors have been investigated carefully. The calculation has been accelerated retained the accuracy at an acceptable level. The cputime has been compared with different acceleration techniques at one time step by changing the number of particles.

The L^2 norm error is defined as the difference in the induced velocity of the same particles between the host and MDGRAPE-3 for the same time step as follows.

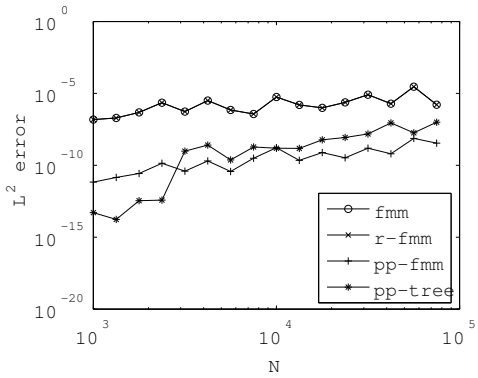
$$L^2(norm\ \ error) = \frac{\sum \left((u_{host} - u_{md3})^2 + (v_{host} - v_{md3})^2 + (w_{host} - w_{md3})^2\right)}{\sum \left(u_{host}^2 + v_{host}^2 + w_{host}^2\right)} \tag{5.1}$$

where the suffices $md3$ and $host$ represent with and without the use of MDGRAPE-3, respectively.

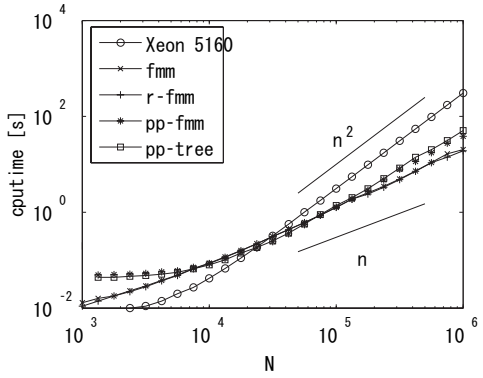
The Biot-Savart and stretching term calculation are performed separately and evaluate the cpu-time for different element numbers. The same calculations are done with and without the MDGRAPE-3. The cpu-time of the Biot-Savart calculation for one time step is plotted against the number of elements



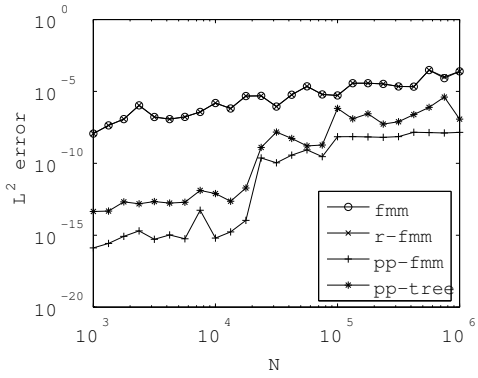
(a) Cpu-time (host)



(b) FMM error (host)



(c) Cpu-time (MDG3)



(d) FMM error (MDG3)

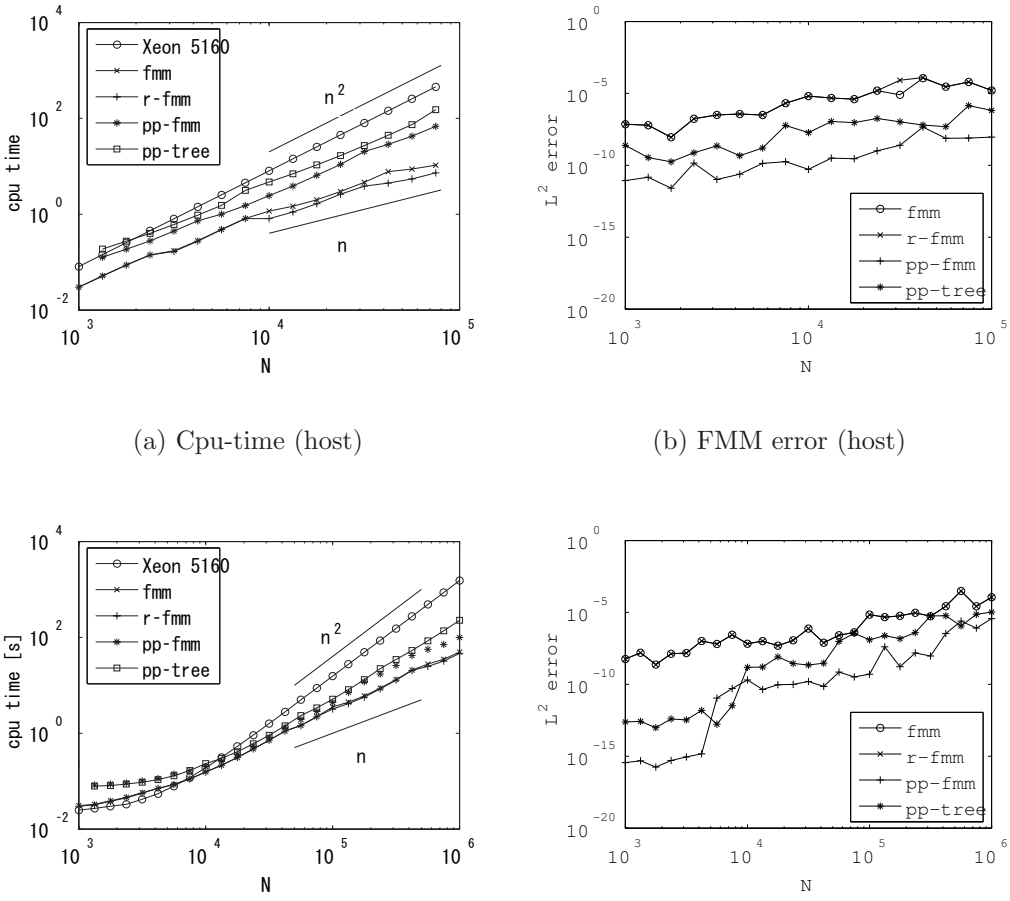
Figure 5.5: Cputime and error of Biot-Savart calculation

in Fig. 5.5. The L^2 norm error between the direct calculation and FMM calculations are also shown. The two figures on the top are for the calculations without the MDGRAPE-3, and the ones below are with the MDGRAPE-3. The number of elements is changed from $10^3 - 10^5$ for the calculations without the MDGRAPE-3, and $10^3 - 10^6$ for the calculations with MDGRAPE-3. The legends 'Xeon 5160', 'fmm', 'r-fmm', 'pp-fmm', and 'pp-tree' correspond to, the calculation without FMM, standard FMM, rotation based FMM, pseudo-particle FMM, pseudo-particle tree code.

The direct calculation on 'Xeon 5160(3.0GHz)' has a cost of $O(N^2)$. The FMM and rotation based FMM are close to O(N), but the pseudo-particle methods both have a steeper slope. This is caused by inefficiency in having to expand the neighbor region, which is the price I must pay for calculating the FMM entirely on the MDGRAPE-3. If these methods have been used without the MDGRAPE-3 they are simply slow methods, as shown in Fig. 5.5(a). However, when these methods are used with the MDGRAPE-3 they are faster than the standard and rotation based FMM when $10^4 < N < 10^5$, as shown in Fig. 5.5(c). For larger N, the pseudo-particle methods are slower due to their steeper slope.

In the present calculations $p = 10$ has been used and the L^2 norm error is below 10^{-5} for most calculations. The pseudo-particle methods have an extremely small error for small N, because these calculations are single level calculations and do not contain multipole to multipole or local to local translation errors. It can be seen that using a 5×5 neighbor region makes the multipole to local translation error quite small. It is not shown here but using a 3×3 neighbor region produces an excessive amount of error. Therefore, the large neighbor region must be used despite the fact that the resulting accuracy seems too high compared to standard FMMs. One resonable way to trade the accuracy surplus of the pseudo-particle methods would be to reduce p.

The cpu-time and L^2 norm error for the stretching term calculation are shown in Fig. 5.6. The general behavior is identical to the Biot-Savart case, but since the stretching term equation is more complex than the Biot-Savart, the calculation takes longer. When compared to the Biot-Savart calculation, the breakeven points between the different slopes are shifted to a smaller N. Thus, the effect of using a method with better scaling is more significant for this case.



(a) Cpu-time (host)

(b) FMM error (host)

(c) Cpu-time (MDG3)

(d) FMM error (MDG3)

Figure 5.6: Cputime and error of stretching term calculation

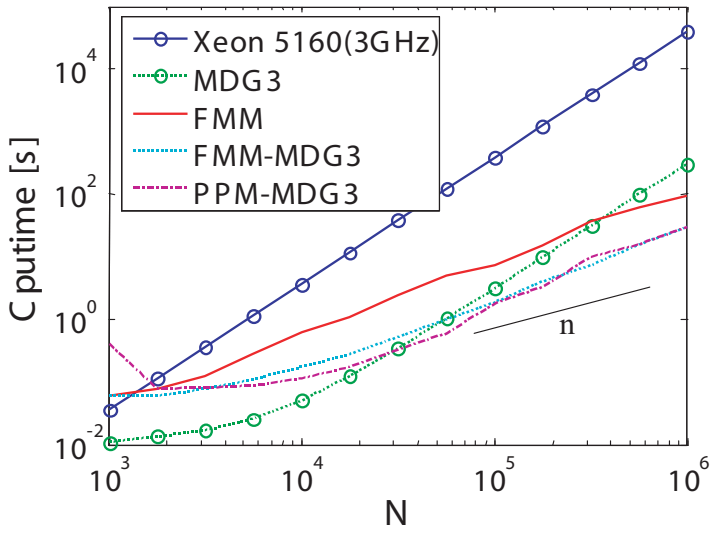


Figure 5.7: CPU-time of different methods

The CPU-time for all methods (when optimized) are plotted in Fig. 5.7. Xeon 5160(3GHz), MDG3, FMM, FMM-MDG3, and PPM-MDG3 represent the calculation without FMM or MDGRAPE-3, with MDGRAPE-3, with FMM, with FMM and MDGRAPE-3, and with the pseudo-particle method and MDGRAPE-3. The direct calculation without MDGRAPE-3 has a high asymptotic constant and an order of $O(N^2)$. All calculations were performed on a dual core Xeon 5160 (3.0GHz) processor. The direct calculation on MDGRAPE-3 has a lower asymptotic constant but still has a scaling of $O(N^2)$. On the contrary, the FMM without MDGRAPE-3 has a high asymptotic constant, but its complexity is $O(N)$. The combination of the FMM and MDGRAPE-3 results in a calculation with a low asymptotic constant and $O(N)$ complexity. The pseudo-particle method on MDGRAPE-3 has a speed comparable to the FMM on MDGRAPE-3. However, it appears from Fig. 5.7 that the pseudo-particle method on MDGRAPE-3 does not quite scale as $O(N)$. At $N = 10^6$ the FMM on MDGRAPE-3 is approximately 4 times faster than the FMM.

The $|L^2|$ norm error from the direct calculation without MDGRAPE-3 is shown for all other methods in Fig. 5.8. The MDGRAPE-3 contains errors of its own, which stem from the partially single precision calculation, and use of

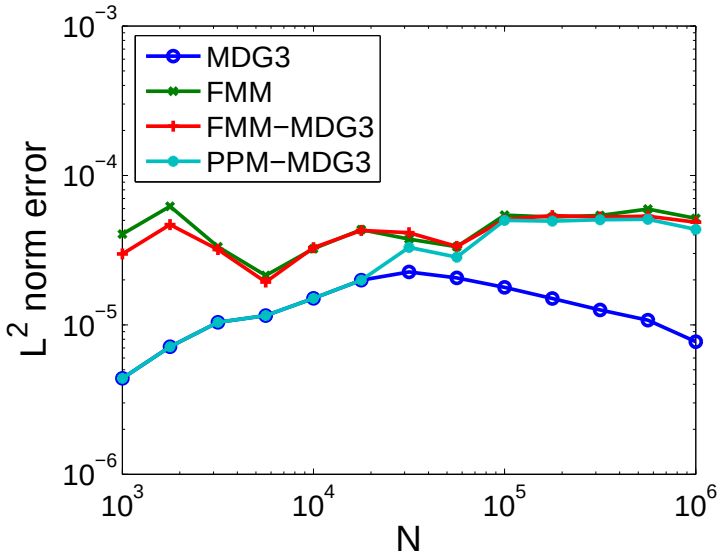
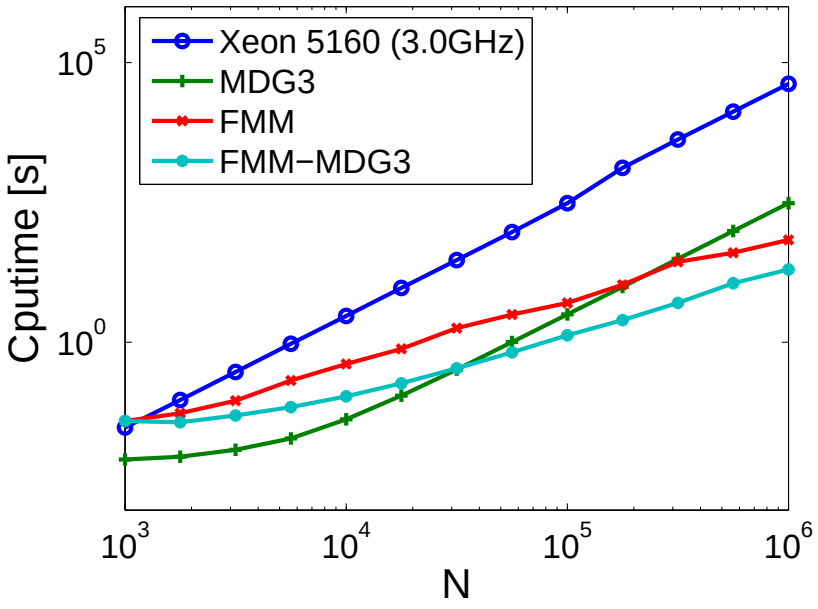


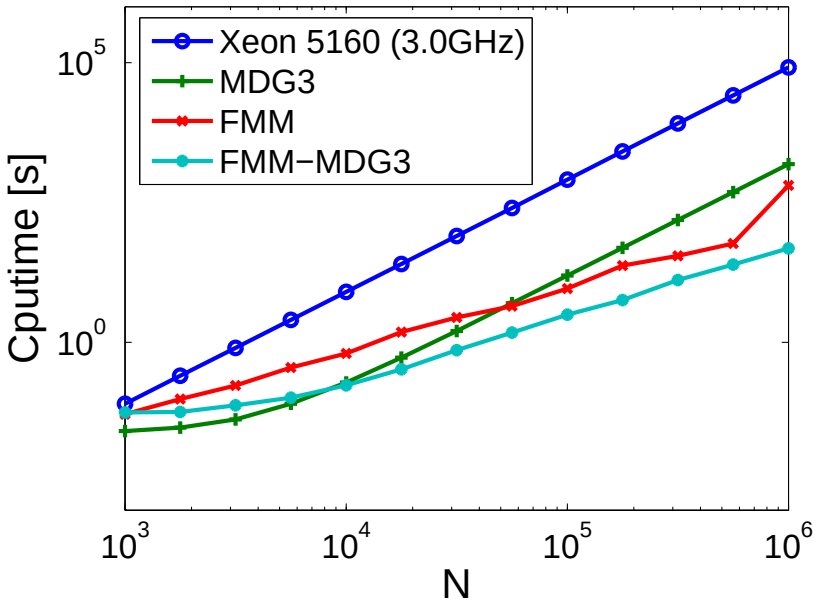
Figure 5.8: $|L^2|$ of different methods

interpolation for the calculation of the cut-off function. This error is constantly lower than the FMM errors for all N. The error of the pseudo-particle method is lower than that of MDGRAPE-3 for $N < 4 \times 10^4$, thus the PPM-MDG3 matches MDG3 until this value.

It is necessary to choose which type of FMM will be used to calculate the collissions of two vortex rings with $N > 10^6$ elements. Considering the results above, the rotation based FMM has been selected by using MDGRAPE-3 for the direct calculation. To clarify the acceleration and accuracy achieved by the combination of FMM and MDGRAPE-3, the results are reploted for the Biot-Savart and stretching term calculations, without any acceleration, with FMM, with MDGRAPE-3, and both FMM and MDGRAPE-3 in Figs. 5.9 and 5.10, respectively. Note that the 'FMM' in these plots correspond to the 'r-fmm' in earlier plots(Figs. 5.5 and 5.6), but for $10^3 < N < 10^6$. It is clearly seen that the direct calculation on 'Xeon 5160' and MDGRAPE-3 both have a scaling of $O(N^2)$, the use of the FMM brings them both down to $O(N)$ in both cases.(Figs. 5.9(a) and 5.9(b)). The MDGRAPE-3 has an small error compared to FMM and in combination with both in Fig. 5.10(a). The error of MDGRAPE-3 controlled by its system and using FMM this error goes slightly increase as in the same or-

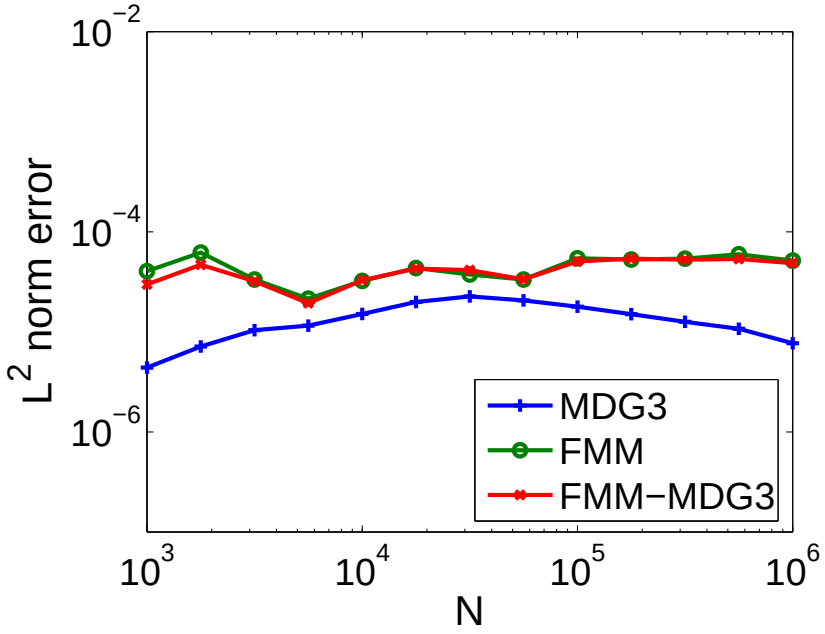


(a) Biot-Savart

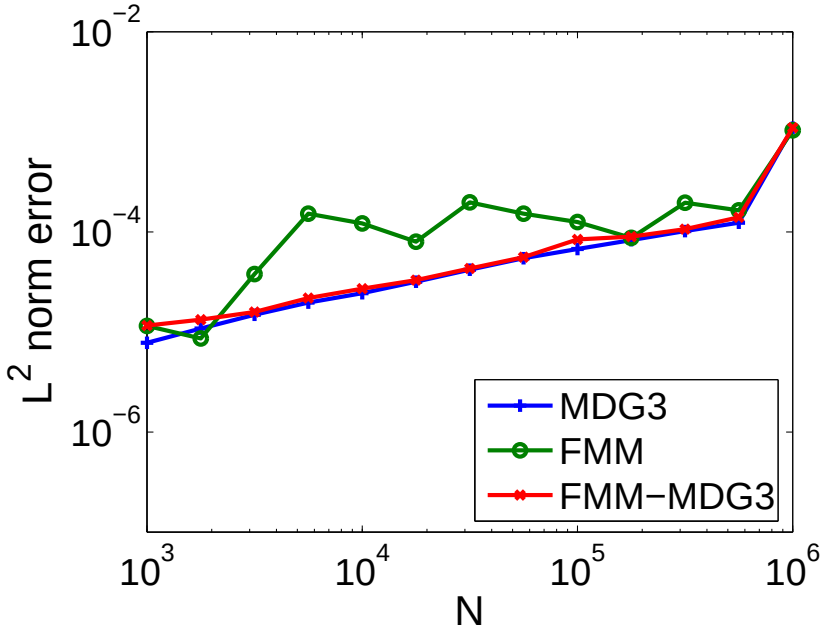


(b) Stretching term

Figure 5.9: Cpu-time of FMM, MDGRAPE-3, and both



(a) Biot-Savart



(b) Stretching term

Figure 5.10: Accuracy of FMM, MDGRAPE-3, and both

Table 5.1: Accelaration ratio at $N = 10^6$

Biot-Savart		stretching	
Xeon 5160 (3.0GHz)		Xeon 5160 (3.0GHz)	
↓ ×462	↓ ×119	↓ ×613	↓ ×52
FMM	MDG3	FMM	MDG3
↓ ×4.1	↓ ×16	↓ ×2.8	↓ ×33
FMM+MDG3		FMM+MDG3	

der of magnitude when compared with FMM. The calculation time takes longer and the L^2 norm error becomes larger for large N in the case of stretching term calculations (Figs. 5.9(b) and 5.10(b)) when compared with Biot-Savart calculation. This error may caused by the vortex strength and the discretization error of stretching term.

The quantitative acceleration ratio for $N = 10^6$ is given in Table. 5.1. For the Biot-Savart calculation, the FMM alone accelerates the calculation 462 times, and simultaneous use of the MDGRAPE-3 further accelerates it 4.1 times. From a different perspective, the MDGRAPE-3 can accelerate the calculation 119 times, but the simultaneous use of the FMM allows a 16 fold increase from that. Similarly, the stretching term calculation is 613 times faster when used the FMM, and another 2.8 times faster when we combine it with the MDGRAPE-3. The MDGRAPE-3 accelerates the stretching term calculation 52 times, and another 33 times if we use the FMM on it.

5.3 Vortex Ring Calculation

5.3.1 Calculation Conditions

The initial radius of the vortex rings was $R = 1$ while the cross-section radius was $r = 0.05$, see Fig. 4.11. The rings were inclined at an angle $\theta = 15^\circ$ relative to the z-axis. The Reynolds number based on the ring circulation was $Re_\Gamma = \Gamma/\nu = 400$.

Two types of initial conditions were used for the present investigation. The

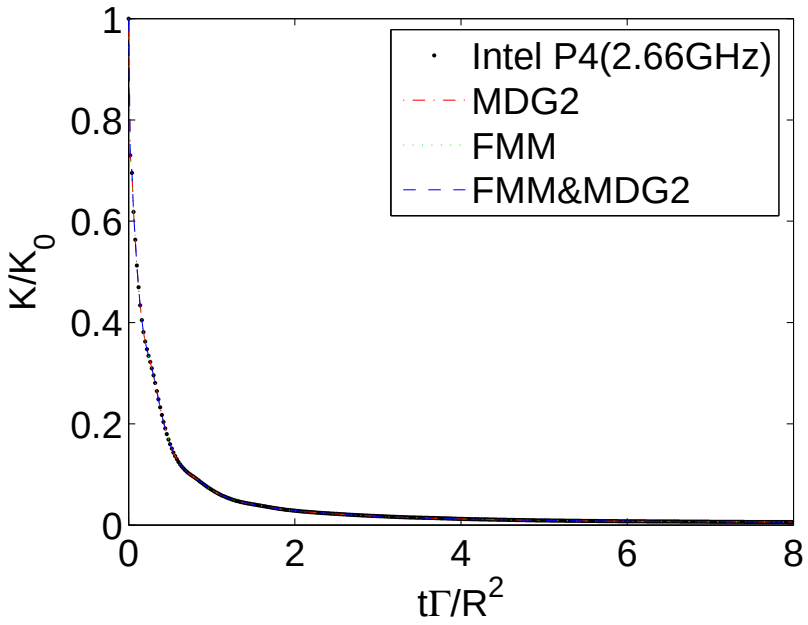
first was identical to the previous calculations (Sheel et al. [2007]) which was used to validate the present method by reproducing previous results. For this case, the initial core radius of the vortex elements was $\sigma_0 = 0.065$, and the total number of elements was $N \approx 6 \times 10^4$, with the number of cross sections in the circumference direction being 502, while 61 elements were distributed in each cross-section of the two rings(Table 4.2). The absolute value of the vortex strength was constant for all elements.

In the second condition, the initial condition was modified so that the vortex method could stably calculate until the reconnection occured. This second initial condition had a Gaussian distribution of vorticity in the cross section, as observed in experiments (Shariff et al. [1994]). The vortex elements were distributed up to $3\sigma_g$, where σ_g is the standard deviation of the Gaussian distribution. This allows the diffusion to take place at the regions surrounding the vortex ring. Furthermore, the initial core radius of the vortex elements σ_0 was set to twice the inter-particle spacing, which guarantees the overlap of elements for long time calculations.

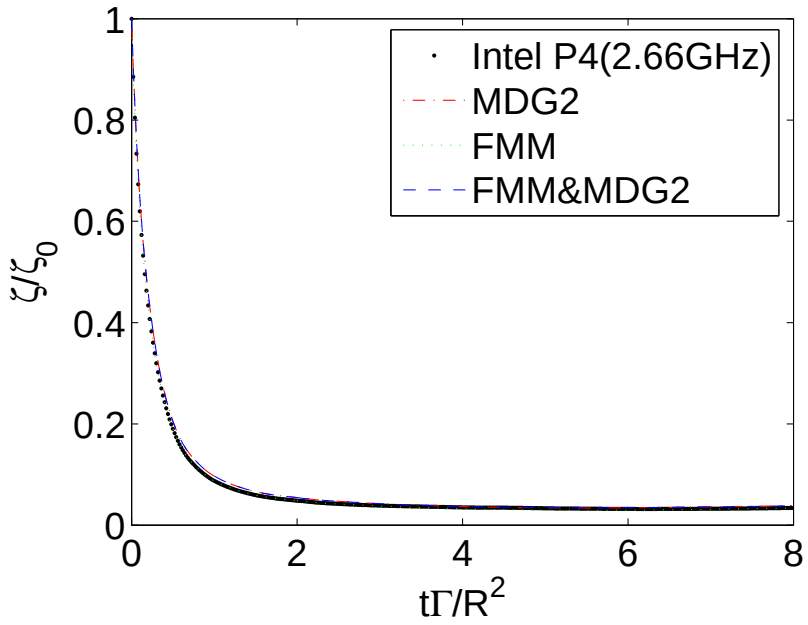
5.3.2 Comparison with previous calculations

Here I will discuss the validity and comparative results as an implementation of FMM on the both of MDGRAPE-2 and MDGRAPE-3. The global kinetic energy and enstrophy have been investigated for the case of MDGRAPE-2 while other details results are checked for MDGRAPE-3 case. The comparative study between MDGRAPE-2 and MDGRAPE-3 has been discussed in chapter 3. The kinetic energy and enstrophy have been calculated according to the Eqs. (4.16) and (4.17), respectively.

The evolution of the kinetic energy compared with different schemes are shown in Fig. 5.11(a). In the figure, Intel P4(2.66GHz), MDG2, FMM and FMM-MDG2 stand for without the use of any acceleration techniques, with the use MDGRAPE-2, with the use of FMM and with the simultaneous use of the FMM and MDGRAPE-2. From the comparison among the results obtained with the various time steps, it has been observed that there is no significant difference between host calculation among with others.



(a) Kinetic Energy



(b) Enstrophy

Figure 5.11: Time history of kinetic energy & enstrophy

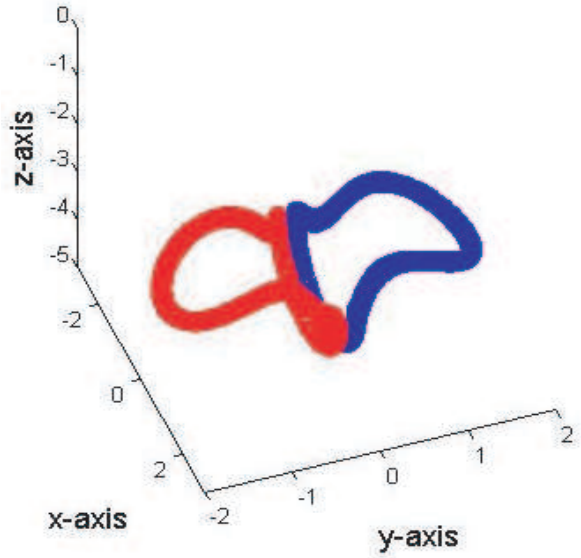
The evolution of the enstrophy compared with different schemes are shown in Fig. 5.11(b). The legends are same as of Fig. 5.11(a). From the comparison between the results obtained with the various time steps, it has been observed that there is no significant difference between host calculation among with others in this case also.

The above results indicate that the new methods are calculated accurately in the present calculation compared with the host and previous calculations as well.

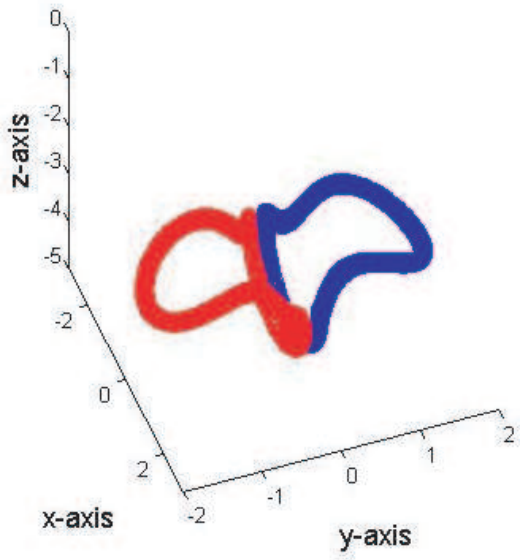
Now I will confirm the validity for the simultaneous use of the FMM and MDGRAPE-3 by comparing the previous results. In this case, the collision of vortex rings is calculated using the first condition mentioned in previous section. In this calculation, the viscous diffusion was calculated using the core-spreading method discussed in chapter 2. For convection of the particles, the second order accurate Adams-Bashforth method was used in the calculation of time advances (Moin [2001]). The kinetic energy K and enstrophy ζ are evaluated from the particle positions and strengths according to Winckelmans and Leonard [1993], and are defined in chapter 4(Eqs. 4.16 and 4.17). The energy spectra are calculated from the velocity distribution along the z-axis at selected times. The fast Fourier transform has been used to calculate one dimensional energy spectra from this velocity.

Figure 5.12 shows the position of vortex elements of two colliding vortex rings in different schemes for the same number of particles at the same time $t\Gamma/R^2 = 10$. It is clearly observed that the flow patterns are quite similar for different schemes in Figs. 5.12 (a)-(c). Considered the above results, it is difficult to analyze the difference pattern without observing the physical characteristics of ring impingement and will be discussed later.

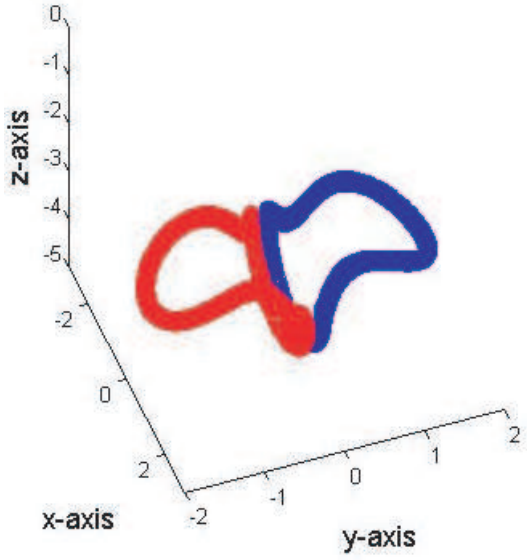
The evolution of the kinetic energy and enstrophy are compared with that of Winckelmans and Leonard [1993] in Fig. 5.13. The time is normalized by the circulation and radius of the vortex ring. The energy and enstrophy are normalized by their initial values. The Xeon 5160, MDG3, FMM-MDG3, and Wnkmns93, represent the direct calculation without MDGRAPE-3, direct calculation on MDGRAPE-3, the calculation of FMM on MDGRAPE-3, and the resutls of Winckelmans and Leonard [1993].



(a) Xeon 5160

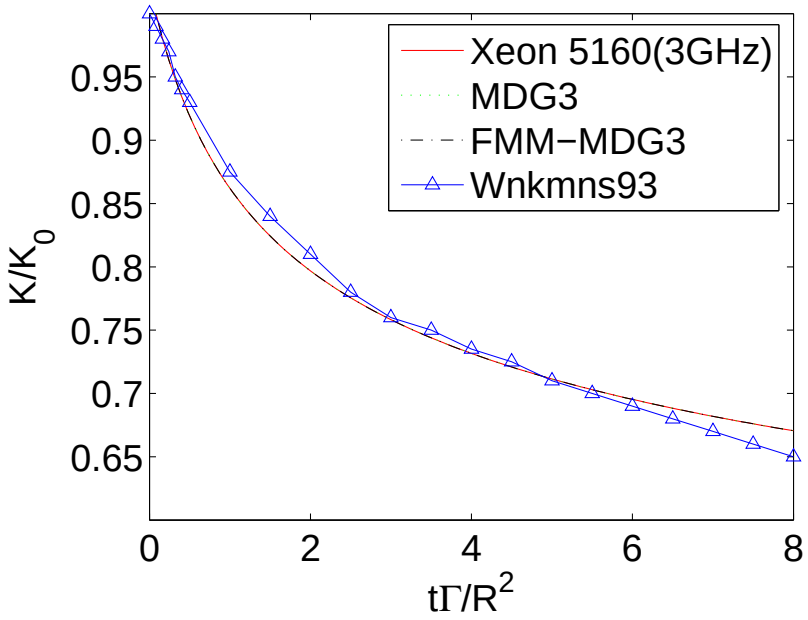


(b) MDG3

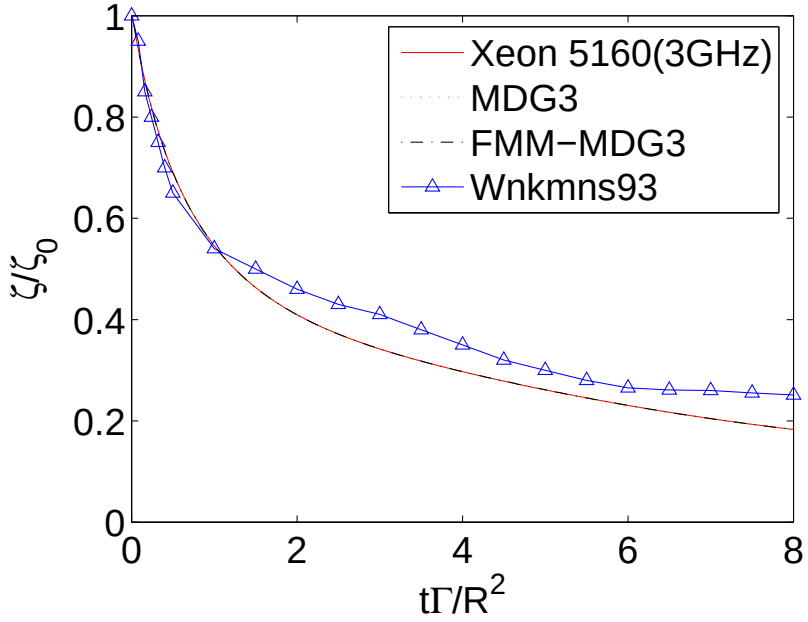


(c) FMM+MDG3

Figure 5.12: Visualization of vortex elements ($t\Gamma/R^2 = 10$)



(a) Kinetic Energy



(b) Enstrophy

Figure 5.13: Time history of kinetic energy & enstrophy

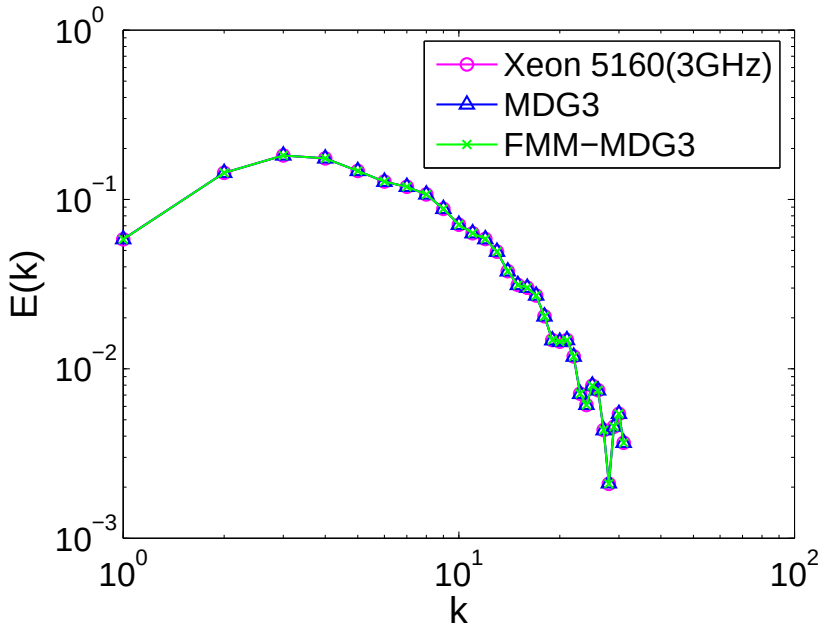


Figure 5.14: Energy Spectra

The results with and without the FMM and MDGRAPE-3 do not show any notable difference. The difference between the results of Winckelmans and Leonard [1993], which is thought to be a consequence of using different viscous diffusion schemes, is also marginal.

The one dimensional kinetic energy spectra is ploted in figure 5.14. In the figure x-axis represents the wave number space and y-axis stands for the one dimensional energy spectra calculated from the velocity distribution along the z-axis. It is clearly observed that the results have excellent agreement with the host calculation among with other methods.

5.3.3 Calculation with Improved Initial Conditions

The collision of vortex rings is calculated using the second condition. The number of particles is changed for $10^5 \leq N \leq 10^7$, while the Reynolds number is kept constant($Re_\Gamma = 400$). The corresponding number of elements per cross section and number of cross sections are shown in Table 5.2. These numbers are

Table 5.2: Breakdown of the Number of Elements

Case	A	B	C
Number of Rings	2	2	2
N per Cross Section	190	418	910
Cross Sections	271	1261	5677
Total	102980	1054196	10332140

determined by choosing a inter-particle distance that yields the total number of elements closest to $N \approx 10^5$, $N \approx 10^6$, and $N \approx 10^7$.

The initial vorticity distribution has been modified so that it has a Gaussian distribution in the cross section (Shariff et al. [1994]).

$$\boldsymbol{\omega} = \frac{\Gamma}{2\pi\sigma^2} \exp\left(\frac{-r^2}{2\sigma^2}\right) \tag{5.2}$$

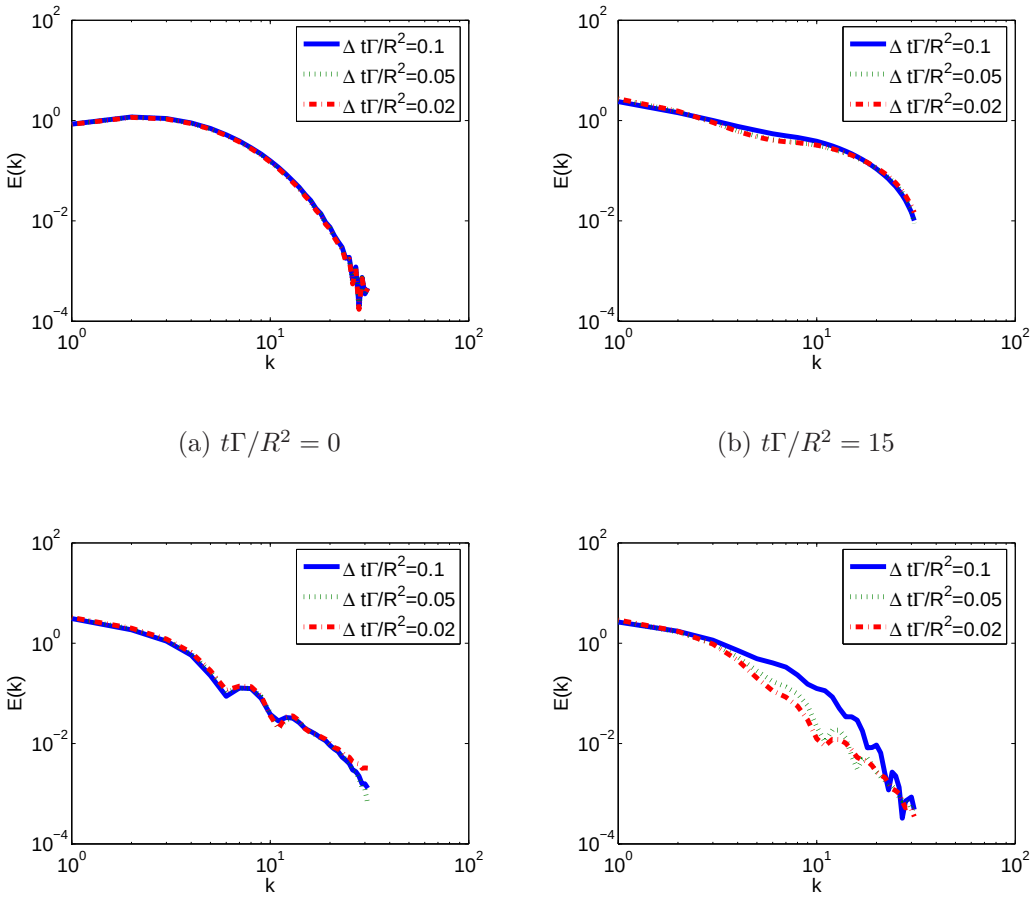
Furthermore, the kinetic energy and enstrophy are now calculated by simply integrating for all particles

$$K = \frac{1}{2}\sum_{i}^{N} \mathbf{u}_i \cdot \mathbf{u}_i \tag{5.3}$$

$$\zeta = \sum_{i}^{N} \boldsymbol{\omega}_i \cdot \boldsymbol{\omega}_i \tag{5.4}$$

5.3.4 Effect of Temporal Resolution

The effect of temporal resolution is investigated for the case of $Re_\Gamma = 400$ and $N \approx 10^5$. The kinetic energy spectra at different times is shown in Fig. 5.15. The energy spectra are calculated in the same way as the previous calculations. The initial energy spectra in Fig. 5.15(a) are similar to those in Fig. 5.14. As the calculation proceeds, the vortex ring collides and causes the energy spectra to change dramatically, as shown in Fig. 5.15(b). At later stages of the calculation, the difference in temporal resolution results in a significant difference between the energy spectra, which is clearly shown in Fig. 5.15(d). The results of $\Delta t\Gamma/R^2 = 0.05$ and $\Delta t\Gamma/R^2 = 0.02$ are closer than that of $\Delta t\Gamma/R^2 = 0.1$.



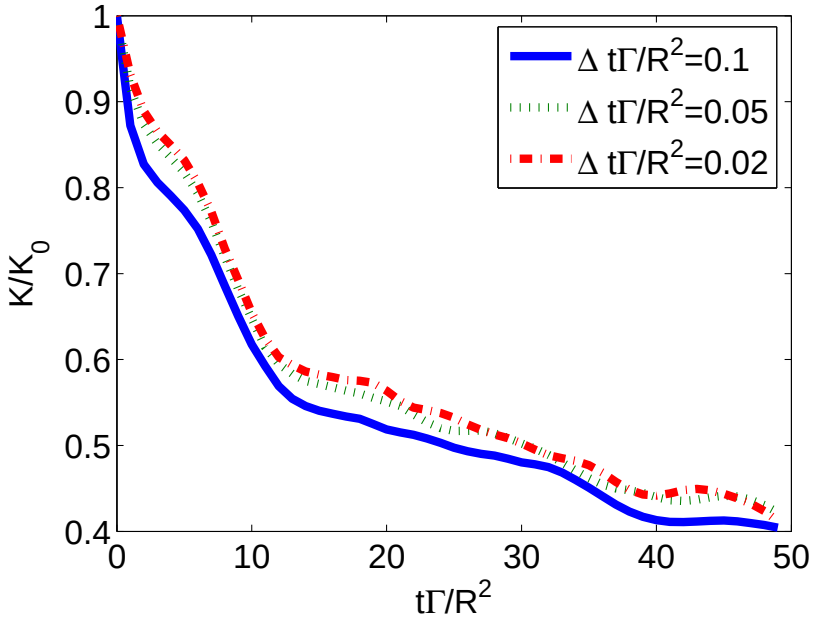
(a) $t\Gamma/R^2 = 0$

(b) $t\Gamma/R^2 = 15$

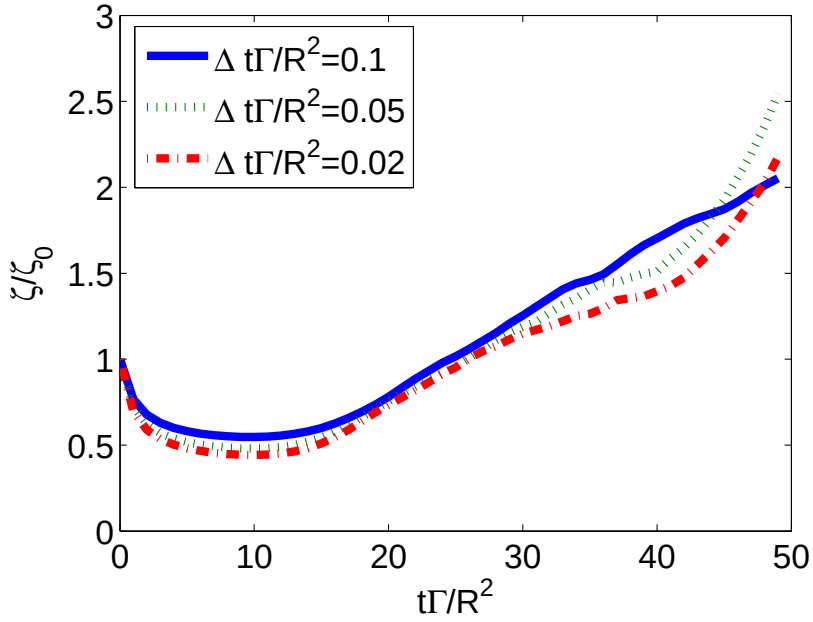
(c) $t\Gamma/R^2 = 30$

(d) $t\Gamma/R^2 = 45$

Figure 5.15: Effect of Temporal Resolution on the Energy Spectra



(a) Kinetic Energy



(b) Enstrophy

Figure 5.16: Effect of Temporal Resolution on the Decay of Kinetic Energy and Enstrophy

The decay of the total kinetic energy and enstrophy are shown in Fig. 5.16. Both of the kinetic energy and enstrophy are normalized by their initial values. Compared to the results of the previous calculations shown in Fig. 5.13, the decay of kinetic energy agrees quite well, where they both decrease to about 0.7 at $t\Gamma/R^2 = 8$. On the other hand, the enstrophy increases after $t\Gamma/R^2 \approx 15$ in the present calculation. It will be shown next that this increase in enstrophy is caused by insufficient spatial resolution.

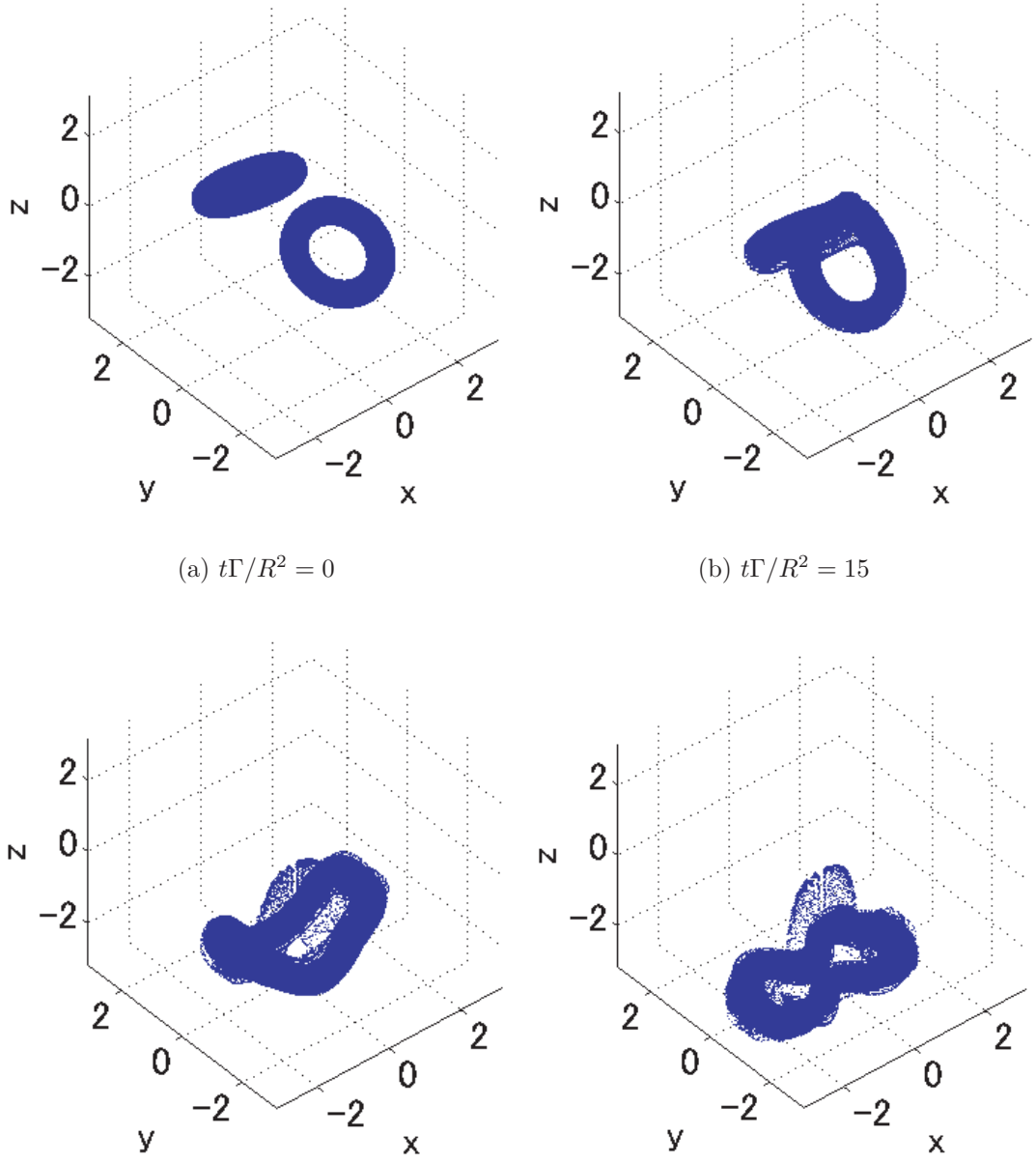
5.3.5 Effect of Spatial Resolution

Before investigating the effect of spatial resolution for turbulent statistics, the movement of vortex elements for different N at different times are shown as follows.

The movement of vortex elements for $Re_\Gamma = 400$ and N for $CaseA$ is plotted in Fig. 5.17. The vortex elements are initially distributed in a ring shape, having a cross sectional radius of 3σ, where σ is the cross sectional radius of the vorticity distribution used in Eq. 5.2. The situation of the rings at $t\Gamma/R^2 = 0$ is the same as the schematic shown in Fig. 4.11. At $t\Gamma/R^2 = 15$, the two rings collide and begin to merge. At $t\Gamma/R^2 = 30$, the vorticity $\boldsymbol{\omega}_y$ near $y = 0$ is mostly canceled, and the two rings move as one. At $t\Gamma/R^2 = 45$ the vortex rings reconnect and form two new rings again.

The movement of vortex elements for $Re_\Gamma = 400$ and N for $CaseB$ is plotted in Fig. 5.18. The plots at $t\Gamma/R^2 = 0$ and 15 are similar to those of Fig. 5.17. The plots at later times seem different at first hand, but the only differences are in the density of the particles. The collision of vortex rings leaves a tail of vortex elements with very small vorticity near the point of origin (0,0,0). This is seen in Figs. 5.17(c) and 5.17(d), but is much more dense in Figs. 5.18(c) and 5.18(d), which causes the figures to appear different.

One of the key features of the present calculation is that the vortex reconnection is reproduced more realistically than the previous calculations by Winckelmans and Leonard [1993] and Fukuda et al. [2005]. These results are supported by the fact that the present method can handle the diffusion more accurately since it considers the diffusion in the region surrounding the rings and also uses



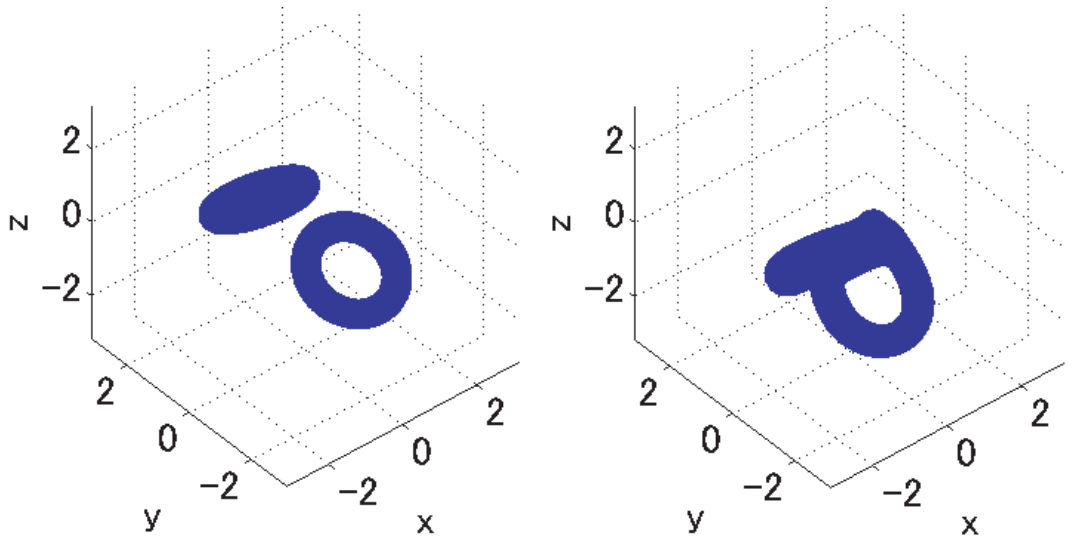
(a) $t\Gamma/R^2 = 0$

(b) $t\Gamma/R^2 = 15$

(c) $t\Gamma/R^2 = 30$

(d) $t\Gamma/R^2 = 45$

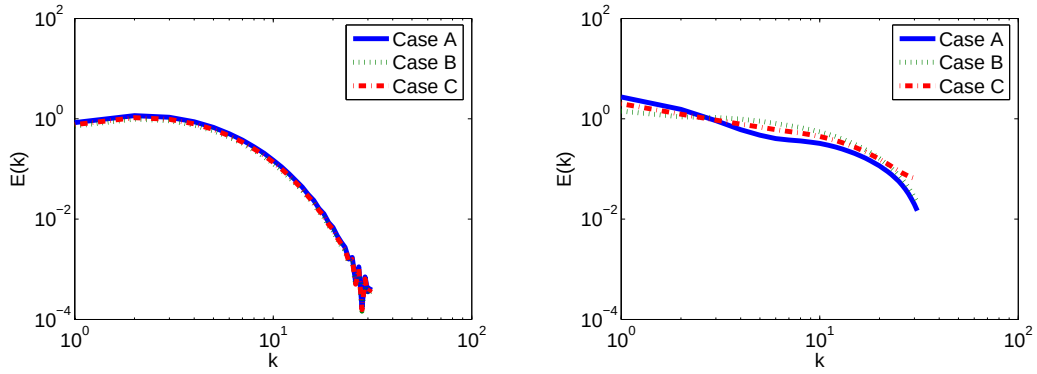
Figure 5.17: Position of Vortex Elements for Case A



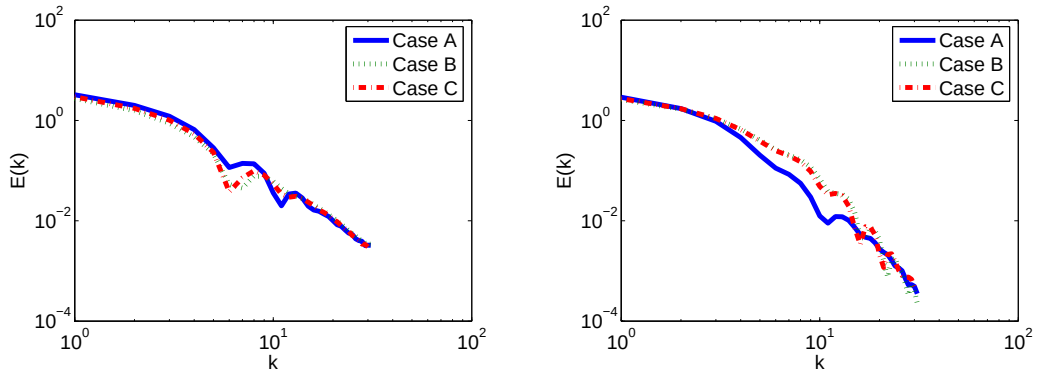
(a) $t\Gamma/R^2 = 0$ (b) $t\Gamma/R^2 = 15$

(c) $t\Gamma/R^2 = 30$ (d) $t\Gamma/R^2 = 45$

Figure 5.18: Position of Vortex Elements for Case B



(a) $t\Gamma/R^2 = 0$ (b) $t\Gamma/R^2 = 15$



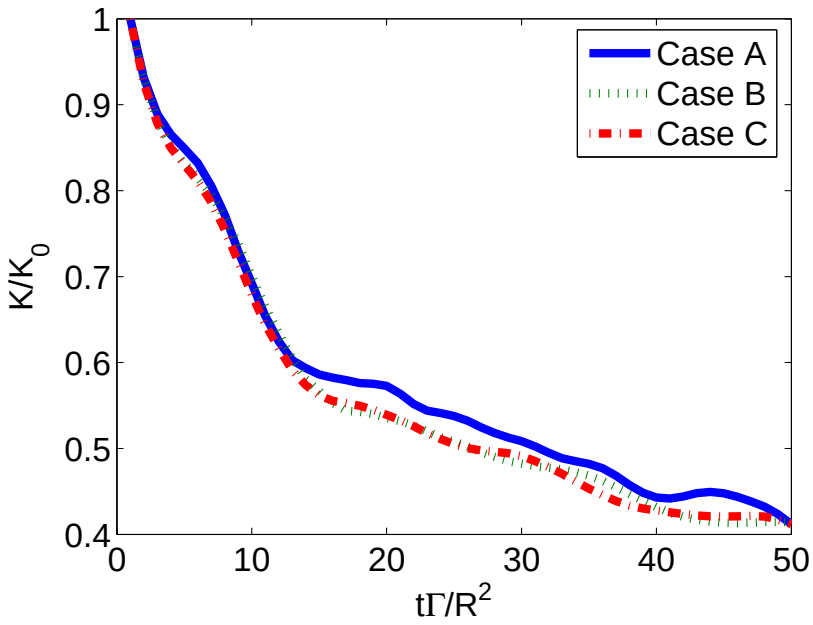
(c) $t\Gamma/R^2 = 30$ (d) $t\Gamma/R^2 = 45$

Figure 5.19: Effect of Spatial Resolution on the Energy Spectra

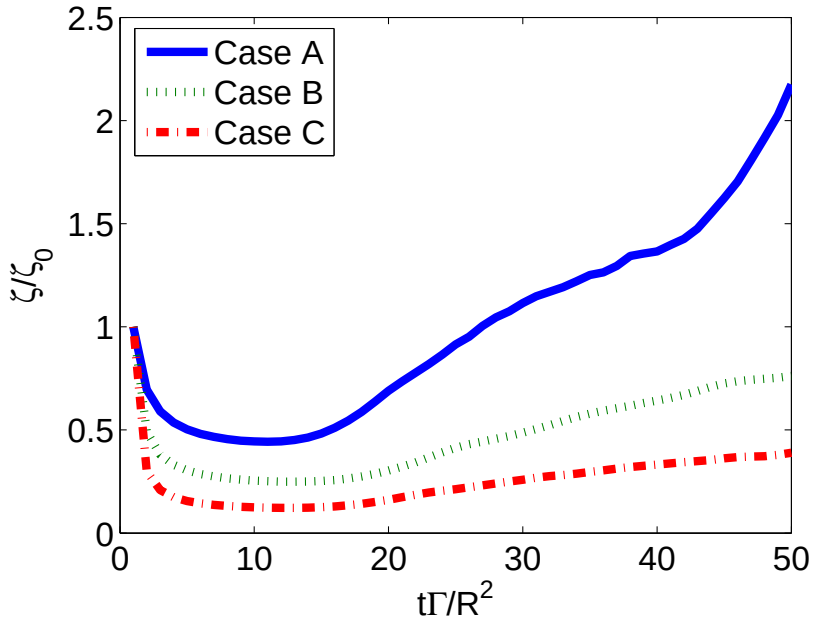
an accurate spatial adaption technique to ensure the convergence of the diffusion scheme for longer calculations. The reconnection has been observed here is also similar to the DNS results of Cottet et al. [2000] (p. 245), experimental results by Kida and Takaoka [1994] and also computations by Chatelain et al. [2003].

The comparison of the position of vortex elements for different N shows little qualitative difference, although the quantitative difference between the turbulence statistics is quite large, which will be shown next.

The Reynolds number $Re_\Gamma = 400$ is kept constant while the number of particles is increased from $N = 10^5$ to $N = 10^7$. The energy spectra for different



(a) Kinetic Energy



(b) Enstrophy

Figure 5.20: Effect of Spatial Resolution on the Decay of Kinetic Energy and Enstrophy

N at various time stages are plotted in Fig. 5.19. The overall behavior of the energy spectra at each time is similar to Fig. 5.15. Initially, there are no notable differences for larger N in Fig. 5.19(a). Significant differences have clearly been observed as time progresses. It has been clearly observed that the energy spectra are closer to each other for larger N in Fig. 5.19(d). The decay of kinetic energy and enstrophy are shown in Fig. 5.20. These plots also show closer results when N is larger. It is clearly shown that the enstrophy in Fig. 5.20(b) does not increases for larger N, as in Fig. 5.16(b). This result shows that the spatial resolution has a large effect on these calculations and for the calculation of $Re_\Gamma = 400$, the vortex method requires $N = 10^7$ particles for an accurate calculation. The above results indicate that for the calculation of further high Reynolds numbers, the vortex method requires significantly larger N, which is possible by using the proposed acceleration method.

5.4 Conclusions

The vortex method calculation is accelerated significantly by the simultaneous use of the FMM and special purpose computers MDGRAPE-2 and MDGRAPE-3. The FMM on MDGRAPE-3 is about 16 times faster than the MDGRAPE-3 itself, and approximately 4 times faster than the FMM on a Xeon 5160 (3.0 GHz) for the Biot-Savart calculation of $N = 10^6$ elements. The errors involved in the use of the MDGRAPE-3 are less than the errors of the FMM, and thus are small enough to perform an accurate vortex method calculation.

The collision of two vortex rings is selected as a test case. The reconnection of the vortex rings in the present calculation is similar to what is seen in experimental and DNS results. This is a result of the high precision of the stretching and diffusion calculations. The results of the calculations using more than 10^6 particles, not only reproduce the qualitative aspects of the reconnection, but also show nearly negligible discretization error. These features support the use of pure Lagrangian vortex methods in fairly complex 3-D flows.

Chapter 6

Conclusions and Outlook

6.1 Fast Vortex Method

A fast vortex method has been developed by using special-purpose computers those were exclusively designed for molecular dynamics simulations. A mathematical formulation has been developed for the 3D vortex method calculation using a special-purpose computer MDGRAPE-2. A rigorous assessment of this hardware has been made for a simple flow calculation using this method. It is found that the generation of appropriate function tables, which are used to call libraries, embedded in MDGRAPE-2 is of primary importance in order to retain accuracy.

This method has been successfully applied to the calculation of two colliding vortex rings in three different configurations. The error arising from the approximation is evaluated by calculating a pair of vortex rings impinging to themselves. The error in the statistical quantities such as kinetic energy and enstrophy remain negligible and have good agreement when compared with others similar work.

MGGRAPE-3 is a successor of MDGRAPE-2 has been applied to the similar calculations and developed fast vortex method. The previous results have been confirmed and further improved are achieved using this new hardware.

The simultaneous use of the FMM with MDGRAPE-2 and MDGRAPE-3 has been successfully applied to the similar calculations. Further acceleration has been achieved by using this new acceleration method. The various forms of FMM and their performance on MDGRAPE-2 and MDGRAPE-3 have been

investigated. The new technique has been successfully applied to the improved initial calcultion conditions after a rigorous validation of previous calculations.

6.2 Optimization

It has been observed that the definition of an optimum range of a function table plays an essential role to control and achieve satisfactory accuracy in MDGRAPE-2. An optimum range of a function table has been determined after a careful observation of computational domain of the collisions of a pair of vortex rings. The similar accuracy has been achieved for Biot-Savart law calculations as of molecular dynamics simulation. The cross product calculations have been handled in a proper manner, which is not considered in the original command set in MDGRAPE-2 library.

The computational domain has been determined by checking the typical distribution of vortex elements for different positions of colliding vortex rings at various non-dimensional times. It has been found that this domain determines the optimal range of a function table.

It is found that the table range is different for different problems. It is necessary to generate a new function table for a new problem. It was carefully observed that the preparation of the function table operation takes only a few minutes, which does not affect the performance of the entire calculations. To simulate high Reynolds number flows using vortex method, it is required to incorporate large number of vortex elements for an accurate calculations. There is no connection between the number of elements and the range of function table. The range is the key factor in maintaining the accuracy and the single precision calculation of MDGRAPE-2 board, which does not have any influence on the calculation of high Reynolds number flows.

An optimum range of a function table has been determined prior to MDGRAPE-3 calculation. The generation procedures are similar as of MDGRAPE-2 calculation and applied to MDGRAPE-3 case.

It has been clearly observed that the most time consuming parts of FMM calculations are multipole to local(M2L) translations for far field particles and direct calculations of neighboring particles. The balance between these two steps

are dependent on the level of box divisions. An optimum level of box division has been determined according to the number of particles being calculated.

The most important issue regarding the simultaneous use of the FMM and MDGRAPE-3 is the balance of the workload between the M2L translation and the direct calculation. MDGRAPE-3 can handled only the direct calculation, hence an optimum level has been determined. It has been observed that the optimum level of box division of the FMM on MDGRAPE-3 is approximately two levels lower than that of the FMM without MDGRAPE-3 because only the direct summation is accelerated.

In addition, an optimum level of box divisions has been determined when the pseudo-particle multipole method implemented on MDGRAPE-3.

6.3 Accelerations

The vortex method calculation is accelerated by the use of a special purpose computer MDGRAPE-2. The collision of two vortex rings is selected as a test case. The calculation cost has been reduced by a factor of 100 when compared with the calculation of a conventional PC (Intel Pentium 4 2.66GHz). This acceleration has been furhter improved when applied another special purpose computer, MDGRAPE-3. MDGRAPE-3 has been applied to the same calculations and the improvement in speed was 1000 times faster when compared with the host PC(Xeon 5160 3.00GHz) and 25 times faster compared with MDGRAPE-2 for $N = 10^6$.

Further acceleration has been achieved for the vortex method calculation with the simultaneous use of the FMM and a special purpose computer MDGRAPE-3. The dynamics of two colliding vortex rings have been studied and the computation time has been reduced by a factor of 2000 compared to a direct calculation on a standard PC(Xeon 5160 3.00GHz). The FMM on MDGRAPE-3 is about 16 times faster than MDGRAPE-3 itself, and approximately 4 times faster than that of the FMM on a Xeon 5160 (3.0 GHz) for the Biot-Savart calculation of $N = 10^6$ elements. From a different perspective, the MDGRAPE-3 can accelerate the calculation 119 times, but the simultaneous use of the FMM allows a 16 fold increase from that. Similarly, the stretching term calculation is 613 times faster

when used the FMM, and another 2.8 times faster when we combine it with the MDGRAPE-3. The MDGRAPE-3 accelerates the stretching term calculation 52 times, and another 33 times if we use the FMM on it.

6.4 Accuracy

The net relative accuracy in the MDGRAPE-2 chip is set to 10^{-7}, since this accuracy is usually satisfactory in MD simulations. The same accuracy has been achieved by calculating the Biot-Savart law in the present study which was also stisfactory for the vortex method calculations.

The scaling errors of function table have been confirmed by comparing with and without the use of MDGRAPE-2.

The convection error has been evaluated by calculating the difference in the position of the same particles between the host and MDGRAPE-2 for the same time step. The error has been checked for different configurations of vortex rings in three different time iterations. In all cases, MDGRAPE-2 error has been increased for larger N but it kept constant in finite range. It has been confirmed that the error is satisfactory for the present vortex method calculations.

It has been confirmed that the errors in MDGRAPE-2 and MDGRAPE-3 are in similar order while the calculation speed of MDGRAPE-3 is faster than that of MDGRAPE-2. The overall errors are satisfactory enough for entire calculations.

Basically, the FMM trades accuracy for speed, and the order of multipole moments p strongly affects the balance between them. The effect of multipole moments has been investigated and found that it has large influence for the accuracy of FMM. The multipole moment was considered $p = 10$ and the error was below 10^{-5} in entire calculations which was also satisfactory for the present vortex method calculation.

The global kinetic energy and enstrophy have been investigated to address the numerical accuracy. The results have good agreement when compared with the previous and referenced work. The results with and without the use of the FMM, MDGRAPE-2, MDGRAPE-3 and the combination of FMM & MDGRAPEs do not show any notable difference.

It has been clearly observed that the errors involved in the use of MDGRAPE-3 are less than the errors of the FMM, and thus are small enough to perform an accurate vortex method calculation.

The reconnection of the vortex rings in the present calculation is similar to what is seen in experimental and DNS results. This is a result of the high precision of the stretching and diffusion calculations. The results of the calculations using more than 10^6 particles, not only reproduce the qualitative aspects of the reconnection, but also show nearly negligible discretization error.

In addition, one-dimensional kinetic energy spectra and corresponding decay of kinetic energy and enstrophy for high Reynolds numbers have been investigated. The effect of spatial and temporal resolution at high Reynolds numbers is investigated by comparing the energy spectrum and decay rate of the kinetic energy and enstrophy.

6.5 Outlook

For the present Reynolds number $Re_\Gamma = 400$, the qualitative behavior of the vortex elements shows little difference between the calculations using $N \approx 10^5$ and $N \approx 10^6$ elements. However, the energy spectrum, decay of kinetic energy, and decay of enstrophy show a large difference between the calculations using $N \approx 10^5$ and $N \approx 10^6$ elements. This difference decreases drastically when the number of elements is increased another order of magnitude to 10^7.

The above discussions indicate that for the calculation of further high Reynolds numbers, the vortex method requires significantly larger N, which is possible by using the proposed acceleration method. Therefore, the proposed method will make a large contribution to the simulations that have been previously difficult to perform with existing methods.

The collision of vortex rings has been simulated using the present acceleration method and the computation time has been reduced significantly without loss of numerical accuracy. The present acceleration rate is not satisfactory enough to calculate highly turbulent flows. There are still some rooms to improve the acceleration rate by reconstructing the subroutines which call MDGRAPE library. In the present routines it is necessary to call library embeded MDGRAPE 18

times for one cross product term calculation which is a threshold for the total calculation cost. The acceleration rate can be increased by reducing the CALLing time in MDRAPE routines. The overall accuracy may be improved further by making more sophisticated function table using high algebraic smoothing function for vortex method calculation.

This method can be applied to calculate the homogeneous shear flow, fully developed channel flow, smoothed particle hydrodynamics (SPH), dissipative particle dynamics (DPD) for mesoscale polymer description, charged particles in plasma physics etc. The above-mentioned achievements indicate that the present method will be an alternative to conventional methods in computational fluid dynamics in near future.

Appendix A

Vortex Methods

A.1 Biot-Savart Law

The velocity field on a three-dimensional problem is

$$\mathbf{u}(\mathbf{x}) = -\frac{1}{4\pi}\int \frac{(x-x')\times\omega(x')}{|x-x'|^3}dV(x') \tag{A.1}$$

Transfer to summation form in three directions are as follows

$$\mathbf{u}(x) = -\frac{1}{4\pi}\sum_{j=0}^{N}\frac{(x-x')\times\Gamma}{|x-x'|^3} \tag{A.2}$$

The discretized form as follows:

$$u_i = -\frac{1}{4\pi}\sum_{j}^{N}\frac{(y_i-y_j)\gamma_j^z-(z_i-z_j)\gamma_j^y}{\{(x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2\}^{1.5}} \tag{A.3}$$

$$v_i = -\frac{1}{4\pi}\sum_{j}^{N}\frac{(z_i-z_j)\gamma_j^x-(x_i-x_j)\gamma_j^z}{\{(x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2\}^{1.5}} \tag{A.4}$$

$$w_i = -\frac{1}{4\pi}\sum_{j}^{N}\frac{(x_i-x_j)\gamma_j^y-(y_i-y_j)\gamma_j^x}{\{(x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2\}^{1.5}} \tag{A.5}$$

Winckelmans and Leonard [1993] model as a cutoff function: $\zeta(\xi) = \frac{15}{8\pi(\xi^2+1)^{7/2}}$

Using cutoff function the Biot-Savart law becomes:

$$\mathbf{u}(x) = -\frac{1}{4\pi}\sum_{j=0}^{N}\frac{(x-x')\times\Gamma}{|x-x'|^3}\times\zeta \tag{A.6}$$

$$\begin{aligned}\mathbf{u}(x) &= -\frac{1}{4\pi}\sum_{j=0}^{N}\frac{|\mathbf{x}_i-\mathbf{x}_j|^2+(5/2)\sigma_j^2}{\left(|\mathbf{x}_i-\mathbf{x}_j|^2+\sigma_j^2\right)^{5/2}}(\mathbf{x}_i-\mathbf{x}_j)\times\boldsymbol{\gamma_j}\\ &= -\frac{1}{4\pi}\sum_{j=0}^{N}\frac{\mathbf{r}_{ij}^2+(5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\mathbf{r}_{ij}\times\boldsymbol{\gamma_j}\end{aligned} \tag{A.7}$$

Second order Adams-Bashforth method has been used for particle convection with time advancement:

$$x^{n+1}=x^n+\left(\frac{3}{2}u^n-\frac{1}{2}u^{n-1}\right)dt \tag{A.8}$$

Core radius has been expanded at the following rate:

$$\frac{d\sigma^2}{dt} = 4\nu \tag{A.9}$$

$$\begin{aligned}\Rightarrow 2\sigma\frac{d\sigma}{dt} &= 4\nu\\ \Rightarrow \frac{d\sigma}{dt} &= \frac{2\nu}{\sigma}\\ \Rightarrow \frac{\sigma^{n+1}-\sigma^n}{\Delta t} &= \frac{2\nu}{\sigma}\\ \Rightarrow \sigma^{n+1} &= \sigma^n+\frac{2\nu}{\sigma^n}dt\end{aligned} \tag{A.10}$$

Finally the Biot-Savart law has been discretized as follows:

$$\mathbf{u}=(u_i\ \ v_i\ \ w_i)\,;\quad \mathbf{r}_{ij}=(x_{ij}\ \ y_{ij}\ \ z_{ij})\,;\quad \gamma_j=\left(\gamma_j^x\ \ \gamma_j^y\ \ \gamma_j^z\right)$$

$$u_i = -\frac{1}{4\pi}\sum_{j=0}^{N}\frac{\mathbf{r}_{ij}^2+(5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\left(y_j\gamma_j^z-z_j\gamma_j^y\right) \tag{A.11}$$

$$v_i = -\frac{1}{4\pi}\sum_{j=0}^{N}\frac{\mathbf{r}_{ij}^2+(5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\left(z_j\gamma_j^x-x_j\gamma_j^z\right) \tag{A.12}$$

$$w_i = -\frac{1}{4\pi}\sum_{j=0}^{N}\frac{\mathbf{r}_{ij}^2+(5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\left(x_j\gamma_j^y-y_j\gamma_j^x\right) \tag{A.13}$$

A.2 Stretching term

The stretching term of vorticity equation (Eq. 2.1) has been discretized as follows using the same cutoff function.

$$\frac{d\omega}{dt} = (\boldsymbol{\omega} \cdot \nabla)\, \mathbf{u}_i \tag{A.14}$$

$$\gamma_i = \omega_i d^3 x_i; \quad \frac{d\gamma_i}{dt} = (\boldsymbol{\gamma}_i \cdot \nabla)\, \mathbf{u}_i$$

$$\begin{aligned} \frac{d\gamma_i}{dt} &= \frac{1}{4\pi} \sum_j \{ -\frac{|\mathbf{x}_i - \mathbf{x}_j|^2 + (5/2)\sigma_j^2}{\left(|\mathbf{x}_i - \mathbf{x}_j|^2 + \sigma_j^2\right)^{5/2}} \boldsymbol{\gamma}_i \times \boldsymbol{\gamma}_j \\ &+ 3\frac{|\mathbf{x}_i - \mathbf{x}_j|^2 + (7/2)\sigma_j^2}{\left(|\mathbf{x}_i - \mathbf{x}_j|^2 + \sigma_j^2\right)^{7/2}} (\boldsymbol{\gamma}_i \cdot (x_i - x_j))((x_i - x_j) \times \boldsymbol{\gamma}_j) \} \end{aligned} \tag{A.15}$$

To make the simplicity and clarification, Eq. A.15 has been separated in two parts as follows.

$$\frac{d\gamma_i}{dt} = \mathbf{stx} + \mathbf{tx}; \quad \Rightarrow \boldsymbol{\gamma}_{i+1} = \boldsymbol{\gamma}_{i-1} + (\mathbf{stx} + \mathbf{tx})\Delta t \tag{A.16}$$

Where

$$\begin{aligned} \mathbf{stx} &= \frac{1}{4\pi} \sum_j -\frac{|\mathbf{x}_i - \mathbf{x}_j|^2 + (5/2)\sigma_j^2}{\left(|\mathbf{x}_i - \mathbf{x}_j|^2 + \sigma_j^2\right)^{5/2}} \boldsymbol{\gamma}_i \times \boldsymbol{\gamma}_j \\ &= -\frac{1}{4\pi} \sum_j \frac{\mathbf{r}_{ij}^2 + (5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2 + \sigma_j^2\right)^{5/2}} \boldsymbol{\gamma}_i \times \boldsymbol{\gamma}_j \end{aligned} \tag{A.17}$$

$$\begin{aligned} \mathbf{tx} &= \frac{3}{4\pi} \sum_j \frac{|\mathbf{x}_i - \mathbf{x}_j|^2 + (7/2)\sigma_j^2}{\left(|\mathbf{x}_i - \mathbf{x}_j|^2 + \sigma_j^2\right)^{7/2}} (\boldsymbol{\gamma}_i \cdot (\mathbf{x}_i - \mathbf{x}_j))((\mathbf{x}_i - \mathbf{x}_j) \times \boldsymbol{\gamma}_j) \\ &= \frac{3}{4\pi} \sum_j \frac{\mathbf{r}_{ij}^2 + (7/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2 + \sigma_j^2\right)^{7/2}} (\boldsymbol{\gamma}_i \cdot (\mathbf{x}_i - \mathbf{x}_j))((\mathbf{x}_i - \mathbf{x}_j) \times \boldsymbol{\gamma}_j) \end{aligned} \tag{A.18}$$

The above equations are discretized in the following forms.

$$stx_i = -\frac{1}{4\pi}\sum_j \frac{\mathbf{r}_{ij}^2 + (5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\left(\gamma_i^y\gamma_j^z - \gamma_i^z\gamma_j^y\right) \tag{A.19}$$

$$sty_i = -\frac{1}{4\pi}\sum_j \frac{\mathbf{r}_{ij}^2 + (5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\left(\gamma_i^z\gamma_j^x - \gamma_i^x\gamma_j^z\right) \tag{A.20}$$

$$stz_i = -\frac{1}{4\pi}\sum_j \frac{\mathbf{r}_{ij}^2 + (5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\left(\gamma_i^x\gamma_j^y - \gamma_i^y\gamma_j^x\right) \tag{A.21}$$

$$\begin{aligned} tx_i = {} & \frac{3}{4\pi}\sum_j \frac{\mathbf{r}_{ij}^2 + (7/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{7/2}}\left(\gamma_i^x(x(i)-x(j)) + \gamma_i^y(y(i)-y(j)) + \gamma_i^z(z(i)-z(j))\right) \\ & \left((y(i)-y(j))\gamma_j^z - (z(i)-z(j))\gamma_j^y\right) \end{aligned} \tag{A.22}$$

$$\begin{aligned} ty_i = {} & \frac{3}{4\pi}\sum_j \frac{\mathbf{r}_{ij}^2 + (7/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{7/2}}\left(\gamma_i^x(x(i)-x(j)) + \gamma_i^y(y(i)-y(j)) + \gamma_i^z(z(i)-z(j))\right) \\ & \left((z(i)-z(j))\gamma_j^x - (x(i)-x(j))\gamma_j^z\right) \end{aligned} \tag{A.23}$$

$$\begin{aligned} tz_i = {} & \frac{3}{4\pi}\sum_j \frac{\mathbf{r}_{ij}^2 + (7/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{7/2}}\left(\gamma_i^x(x(i)-x(j)) + \gamma_i^y(y(i)-y(j)) + \gamma_i^z(z(i)-z(j))\right) \\ & \left((x(i)-x(j))\gamma_j^y - (y(i)-y(j))\gamma_j^x\right) \end{aligned} \tag{A.24}$$

Appendix B

Mathematical Formulations of MDGRAPE-2

B.1 Biot-Savart Law

The velocity field on a two-dimensional problem is

$$\mathbf{u}(\mathbf{x}) = \frac{1}{2\pi} \int \frac{(x - x') \times \omega(x')}{|x - x'|^2} ds(x) \tag{B.1}$$

Transfer to summation form as follows

$$\mathbf{u}_i = \frac{1}{2\pi} \sum_{j=0}^{N} \frac{(x - x') \times \Gamma_j}{|x - x'|^2} = \sum_{j=0}^{N} \frac{\mathbf{r}_{ij} \times \Gamma_j}{2\pi |\mathbf{r}_{ij}|^2} \tag{B.2}$$

Where: $\Gamma = \int \omega ds$, ω is the vorticity, $\mathbf{r}_{ij} = r_i - r_j$ stands the destance of position vectors.

Now using the Gaussian smoothing as a cutoff function

$$\zeta = \left(1 - exp\left(-\left(\frac{|\mathbf{r}_{ij}|}{\sigma_j}\right)^2\right)\right)$$

Then Biot-Savart law becomes:

$$\mathbf{u}_i^x = \sum_{j=0}^{N} \frac{\mathbf{r}_{ij} \times \Gamma_j}{2\pi |\mathbf{r}_{ij}|^2} \times \left(1 - exp\left(-\left(\frac{|\mathbf{r}_{ij}|}{\sigma_j}\right)^2\right)\right) \tag{B.3}$$

Where σ_j is the core radius of vortex element.

The Biot-Savart law contains cross-product term and MDGRAPE-2 can not handle this term directly. A special treatment is necessary to make it compatible for MDGAPE-2 calculation as follows.

$$\mathbf{r}_{ij} = (x_{ij}, y_{ij}, z_{ij}), \mathbf{\Gamma}_j = (\Gamma_j^x, \Gamma_j^y, \Gamma_j^z);$$

The total moment is:

$$\sum_j \mathbf{r}_{ij} \times \mathbf{\Gamma}_j = \sum_j (y_{ij}\Gamma_j^z - z_{ij}\Gamma_j^y,\ \ z_{ij}\Gamma_j^x - x_{ij}\Gamma_j^z,\ \ x_{ij}\Gamma_j^y - y_{ij}\Gamma_j^x) \tag{B.4}$$

In 2-D case: $z_{ij} = 0$, $\Gamma_j^x = 0$, $\Gamma_j^y = 0$

$$\begin{aligned}
\mathbf{u}_i^x &= \sum_{j=0}^{N} \frac{y_{ij}\Gamma_j^z - z_{ij}\Gamma_j^y}{2\pi|\mathbf{r}_{ij}|^2} \times \left(1 - exp\left(-\left(\frac{|\mathbf{r}_{ij}|}{\sigma_j}\right)^2\right)\right) \\
&= \sum_{j=0}^{N} \frac{y_{ij}\Gamma_j^z}{2\pi|\mathbf{r}_{ij}|^2} \times \left(1 - exp\left(-\left(\frac{|\mathbf{r}_{ij}|}{\sigma_j}\right)^2\right)\right) \\
&- \underbrace{\sum_{j=0}^{N} \frac{z_{ij}\Gamma_j^y}{2\pi|\mathbf{r}_{ij}|^2} \times \left(1 - exp\left(-\left(\frac{|\mathbf{r}_{ij}|}{\sigma_j}\right)^2\right)\right)}_{0} \\
\mathbf{u}_i^x &= \sum_{j=0}^{N} \frac{\frac{\Gamma_j^z}{\sigma_j^2} \cdot \frac{1}{\sigma_j^2}}{2\pi|\mathbf{r}_{ij}|^2 \cdot \frac{1}{\sigma_j^2} \cdot \frac{1}{\sigma_j^2}} \times \left(1 - exp\left(-\frac{\mathbf{r}_{ij}^2}{\sigma_j^2}\right)\right) y_{ij} \\
OR &= \sum_{j=0}^{N} \frac{\frac{\Gamma_j^z}{\sigma_j} \cdot \frac{1}{\sigma_j}}{2\pi|\mathbf{r}_{ij}|^2 \cdot \frac{1}{\sigma_j} \cdot \frac{1}{\sigma_j}} \times \left(1 - exp\left(-\frac{\mathbf{r}_{ij}^2}{\sigma_j^2}\right)\right) y_{ij} \\
&= \frac{1}{2\pi} \sum_{j=0}^{N} \frac{\Gamma_j^z}{\sigma_j^2} \cdot \frac{\left(1 - exp\left(-\frac{\mathbf{r}_{ij}^2}{\sigma_j^2}\right)\right)}{\frac{|\mathbf{r}_{ij}|^2}{\sigma_j^2}} \cdot \frac{1}{\frac{1}{\sigma_j^2}} \cdot \frac{|\mathbf{r}_{ij}|^2}{|\mathbf{r}_{ij}|^2} \cdot \frac{1}{\sigma_j^2} y_{ij} \\
OR &= \frac{1}{2\pi} \sum_{j=0}^{N} \frac{\Gamma_j^z}{\sigma_j} \cdot \frac{\left(1 - exp\left(-\frac{\mathbf{r}_{ij}^2}{\sigma_j^2}\right)\right)}{\frac{|\mathbf{r}_{ij}|^2}{\sigma_j^2}} \cdot \frac{1}{\sigma_j} \cdot y_{ij} \\
&= \frac{1}{2\pi} \sum_{j=0}^{N} \frac{\Gamma_j^z}{\sigma_j^2} \cdot \frac{\left(1 - exp\left(-\frac{\mathbf{r}_{ij}^2}{\sigma_j^2}\right)\right)}{\frac{|\mathbf{r}_{ij}|^2}{\sigma_j^2} \cdot \frac{|\mathbf{r}_{ij}|^2}{\sigma_j^2}} \cdot \frac{1}{\sigma_j^2} \cdot |\mathbf{r}_{ij}|^2 y_{ij} \\
OR &= \frac{1}{2\pi} \sum_{j=0}^{N} \frac{\Gamma_j^z}{\sigma_j} \cdot \frac{\left(1 - exp\left(-\frac{\mathbf{r}_{ij}^2}{\sigma_j^2}\right)\right)}{\frac{|\mathbf{r}_{ij}|^2}{\sigma_j^2}} \cdot \frac{1}{\sigma_j} \cdot y_{ij}
\end{aligned}$$

$$\begin{aligned}
\mathbf{u}_i^x &= \frac{1}{2\pi}\sum_{j=0}^{N} B_j^z \cdot \frac{(1-exp(-w))}{w.w} \cdot A_j \cdot |\mathbf{r}_{ij}|^2 y_{ij} \\
&= \frac{1}{2\pi}\sum_{j=0}^{N} B_j^z g\left(A_j\left(|\mathbf{r}_{ij}|^2+\epsilon_{ij}^2\right)\right) y_{ij} \qquad \text{(B.5)} \\
\mathbf{u}_i^x &= \frac{1}{2\pi}\mathbf{u}_i^{yz} - \frac{1}{2\pi}\mathbf{u}_i^{zy} \qquad \text{(B.6)}
\end{aligned}$$

In 2-D case:

$$\mathbf{u}_i^x = \frac{1}{2\pi}\mathbf{u}_i^{yz} \qquad \text{(B.7)}$$

Similarly we can calculate:

$$\begin{aligned}
\mathbf{u}_i^y &= \frac{1}{2\pi}\sum_{j=0}^{N} B_j^z \cdot \frac{(1-exp(-w))}{w.w} \cdot A_j \cdot |\mathbf{r}_{ij}|^2 x_{ij} \\
&= \frac{1}{2\pi}\sum_{j=0}^{N} B_j^z g\left(A_j\left(|\mathbf{r}_{ij}|^2+\epsilon_{ij}^2\right)\right) x_{ij} \qquad \text{(B.8)} \\
\mathbf{u}_i^y &= \frac{1}{2\pi}\mathbf{u}_i^{zx} - \frac{1}{2\pi}\mathbf{u}_i^{xz} \qquad \text{(B.9)}
\end{aligned}$$

In 2-D case:

$$\mathbf{u}_i^y = -\frac{1}{2\pi}\mathbf{u}_i^{xz} \qquad \text{(B.10)}$$

Where:

$B_j^z = \frac{\Gamma_j^z}{\sigma_j^2}$, $g(w) = \frac{(1-exp(-w))}{w}$, $A_j = \frac{1}{\sigma_j^2}$, $\epsilon_{ij}^2 = 0$, $w = \left(\frac{|\mathbf{r}_{ij}|}{\sigma_j}\right)^2$

Finally Biot-Savart law becomes

$$\mathbf{u}_i = \sum_{j=0}^{N} B_j g\left(A_j\left(|\mathbf{r}_{ij}|^2+\epsilon_{ij}^2\right)\right)\mathbf{r}_{ij} \qquad \text{(B.11)}$$

It can be written in discretized forms as follows:

$$\begin{aligned}
u_i &= \sum_{j=0}^{N} B_j^x g\left(A_j\left(|\mathbf{r}_{ij}|^2+\epsilon_{ij}^2\right)\right)\mathbf{r}_{ij} \qquad \text{(B.12)} \\
v_i &= \sum_{j=0}^{N} B_j^y g\left(A_j\left(|\mathbf{r}_{ij}|^2+\epsilon_{ij}^2\right)\right)\mathbf{r}_{ij} \qquad \text{(B.13)} \\
w_i &= \sum_{j=0}^{N} B_j^z g\left(A_j\left(|\mathbf{r}_{ij}|^2+\epsilon_{ij}^2\right)\right)\mathbf{r}_{ij} \qquad \text{(B.14)}
\end{aligned}$$

Where

$$g(w) = \frac{w+5/2}{(w+1)^{5/2}};\ \ A_j = \frac{1}{\sigma_j^2};\ \ B_j = \frac{\gamma_j}{\sigma_j^3}\left(B_j^x = \frac{\gamma_j^x}{\sigma_j^3},\ \ B_j^y = \frac{\gamma_j^y}{\sigma_j^3},\ \ B_j^z = \frac{\gamma_j^z}{\sigma_j^3}\right) \tag{B.15}$$

B.2 Stretching Term

The stretching term of vorticity transport equation (Eq. 2.1) has been derived as follows to make it compatible to MDGRAPE-2 calculation.

$$\begin{aligned}
\mathbf{stx} &= \frac{1}{4\pi}\sum_j -\frac{|\mathbf{x}_i-\mathbf{x}_j|^2+(5/2)\sigma_j^2}{\left(|\mathbf{x}_i-\mathbf{x}_j|^2+\sigma_j^2\right)^{5/2}}\boldsymbol{\gamma}_i\times\boldsymbol{\gamma}_j \\
&= -\frac{1}{4\pi}\sum_j \frac{\mathbf{r}_{ij}^2+(5/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{5/2}}\boldsymbol{\gamma}_i\times\boldsymbol{\gamma}_j \\
&= -\frac{1}{4\pi}\sum_j \frac{\left(\mathbf{r}_{ij}/\sigma_j\right)^2+5/2}{\left(\left(\mathbf{r}_{ij}/\sigma_j\right)^2+1\right)^{5/2}}\left(\boldsymbol{\gamma}_i\times\boldsymbol{\gamma}_j\right)\frac{1}{\sigma_j^3} \\
&= -\frac{1}{4\pi}\sum_j g1(w)\left(\boldsymbol{\gamma}_i\times\boldsymbol{\gamma}_j\right)\frac{1}{\sigma_j^3} \\
&= -\frac{1}{4\pi}\sum_j g1(w)\left(\gamma_i^y\gamma_j^z-\gamma_i^z\gamma_j^y,\ \ \gamma_i^z\gamma_j^x-\gamma_i^x\gamma_j^z,\ \ \gamma_i^x\gamma_j^y-\gamma_i^y\gamma_j^x\right)\frac{1}{\sigma_j^3} \\
&= -\frac{1}{4\pi}\left(\gamma_i^y K_i-\gamma_i^z J_i,\ \ \gamma_i^z I_i-\gamma_i^x K_i,\ \ \gamma_i^x J_i-\gamma_i^y I_i\right)
\end{aligned} \tag{B.16}$$

Where I_i, J_i, K_i are calculated in potential mode at MDGRAPE-2 as follows.

$$g1(w) = \frac{\left(\mathbf{r}_{ij}/\sigma_j\right)^2+5/2}{\left(\left(\mathbf{r}_{ij}/\sigma_j\right)^2+1\right)^{5/2}} \tag{B.17}$$

$$I_i = \sum_j g1\left(\mathbf{r}_{ij}^2\right)\frac{\gamma_j^x}{\sigma_j^3} = \sum_j g1(w)B_j^x; \tag{B.18}$$

$$J_i = \sum_j g1\left(\mathbf{r}_{ij}^2\right)\frac{\gamma_j^y}{\sigma_j^3} = \sum_j g1(w)B_j^y; \tag{B.19}$$

$$K_i = \sum_j g1\left(\mathbf{r}_{ij}^2\right)\frac{\gamma_j^z}{\sigma_j^3} = \sum_j g1(w)B_j^z; \tag{B.20}$$

$$
\begin{aligned}
\mathbf{tx} &= \frac{3}{4\pi}\sum_j \frac{\mathbf{r}_{ij}^2 + (7/2)\sigma_j^2}{\left(\mathbf{r}_{ij}^2+\sigma_j^2\right)^{7/2}}(\boldsymbol{\gamma}_i\cdot(\mathbf{x}_i-\mathbf{x}_j))((\mathbf{x}_i-\mathbf{x}_j)\times\boldsymbol{\gamma}_j) \\
&= \frac{3}{4\pi}\sum_j \frac{\left(\mathbf{r}_{ij}/\sigma_j\right)^2+7/2}{\left(\left(\mathbf{r}_{ij}/\sigma_j\right)^2+1\right)^{7/2}}\left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij}\right)\left(\mathbf{r}_{ij}\times\boldsymbol{\gamma}_j\right)\frac{1}{\sigma_j^5} \\
&= \frac{3}{4\pi}\sum_j g2\left(r_{ij}^2\right)\left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij}\right)\left(\mathbf{r}_{ij}\times\boldsymbol{\gamma}_j\right)\frac{1}{\sigma_j^5} \qquad \text{(B.21)} \\
&= \frac{3}{4\pi}\sum_j g2\left(r_{ij}^2\right)\left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij}\right)\left(y_{ij}\gamma_j^z - z_{ij}\gamma_j^y,\ \ , z_{ij}\gamma_j^x - x_{ij}\gamma_j^z,\ \ x_{ij}\gamma_j^y - y_{ij}\gamma_j^x\right)\frac{1}{\sigma_j^5}
\end{aligned}
$$

Where,

$$
g2\left(\mathbf{r}_{ij}^2\right) = \frac{\left(\mathbf{r}_{ij}/\sigma_j\right)^2+7/2}{\left(\left(\mathbf{r}_{ij}/\sigma_j\right)^2+1\right)^{7/2}} \qquad \text{(B.22)}
$$

γ_j^x can be expressed in vector $\mathbf{I}_i$ as follows

$$
\begin{aligned}
\mathbf{I}_i &= \left(I_i^x,\ I_i^y,\ I_i^z\right) \qquad \text{(B.23)} \\
&= \sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij}\right)\left(\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij} \\
&= \sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(\boldsymbol{\gamma}_i\cdot(\mathbf{r}_i-\mathbf{r}_j)\right)\left(\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij} \\
&= \sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(\boldsymbol{\gamma}_i\mathbf{r}_i-\boldsymbol{\gamma}_i\mathbf{r}_j\right)\left(\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij} \\
&= (\boldsymbol{\gamma_i}\cdot\mathbf{r}_i)\sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij} - \sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(\boldsymbol{\gamma_i}\cdot\mathbf{r}_j\right)\left(\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij} \\
&= (\boldsymbol{\gamma_i}\cdot\mathbf{r}_i)\sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij} \\
&- \{\gamma_i^x\sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(x_j\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij} + \gamma_i^y\sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(y_j\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij} \\
&+ \gamma_i^y\sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(z_j\gamma_j^x/\sigma_j^5\right)\cdot\mathbf{r}_{ij}\} \qquad \text{(B.24)} \\
&= (\boldsymbol{\gamma_i}\cdot\mathbf{r}_i)\,\mathbf{S} - (\gamma_i^x\mathbf{T1}+\gamma_i^y\mathbf{T2}+\gamma_i^z\mathbf{T3}) \qquad \text{(B.25)} \\
&= (\gamma_i^x x_i+\gamma_i^y y_i+\gamma_i^z z_i)\left(S_x,\ \ S_y,\ \ S_z\right) \\
&- \{\gamma_i^x(T1_x,\ \ T1_y,\ \ T1_z)+\gamma_i^y(T2_x,\ \ T2_y,\ \ T2_z)+\gamma_i^z(T3_x,\ \ T3_y,\ \ T3_z)\}
\end{aligned}
$$

$$
\begin{aligned}
\Rightarrow I_i^x &= (\gamma_i^x x_i + \gamma_i^y y_i + \gamma_i^z z_i)\,(S_x,\ S_y,\ S_z) \\
&- (\gamma_i^x T1_x + \gamma_i^y T2_x + \gamma_i^z T3_x,\ \gamma_i^x T1_y + \gamma_i^y T2_y + \gamma_i^z T3_y,\ \gamma_i^x T1_z + \gamma_i^y T2_z + \gamma_i^z T3_z) \\
&= \sum_j g\left(\mathbf{r}_{ij}^2\right)(\boldsymbol{\gamma_i}\cdot\mathbf{r}_{ij})\left(x_{ij}\gamma_j^x,\ y_{ij}\gamma_j^x,\ z_{ij}\gamma_j^x\right) \qquad \text{(B.26)}
\end{aligned}
$$

Similarly

$$I_i^y = \sum_j g\left(\mathbf{r}_{ij}^2\right)(\boldsymbol{\gamma_i}\cdot\mathbf{r}_{ij})\left(x_{ij}\gamma_j^y,\ y_{ij}\gamma_j^y,\ z_{ij}\gamma_j^y\right) \qquad \text{(B.27)}$$

$$I_i^z = \sum_j g\left(\mathbf{r}_{ij}^2\right)(\boldsymbol{\gamma_i}\cdot\mathbf{r}_{ij})\left(x_{ij}\gamma_j^z,\ y_{ij}\gamma_j^z,\ z_{ij}\gamma_j^z\right) \qquad \text{(B.28)}$$

Where

$$\mathbf{S} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)B_j^x\cdot\mathbf{r}_{ij} \qquad \text{(B.29)}$$

$$\mathbf{T1} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{x_j\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)x_jB_j^x\cdot\mathbf{r}_{ij} \qquad \text{(B.30)}$$

$$\mathbf{T2} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{y_j\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)y_jB_j^x\cdot\mathbf{r}_{ij} \qquad \text{(B.31)}$$

$$\mathbf{T3} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{z_j\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)z_jB_j^x\cdot\mathbf{r}_{ij} \qquad \text{(B.32)}$$

Finally $\mathbf{I}_i$ becomes,

$$
\begin{aligned}
\mathbf{I}_i &= (\boldsymbol{\gamma}_i\cdot\mathbf{r}_i)\,\mathbf{S} - (\gamma_i^x\mathbf{T1} + \gamma_i^y\mathbf{T2} + \gamma_i^z\mathbf{T3}) \\
&= \sum_j g2\left(\mathbf{r}_{ij}^2\right)(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij})\frac{\gamma_j^x}{\sigma_j^5}\cdot\mathbf{r}_{ij} \qquad \text{(B.33)} \\
&= \sum_j g2\,(w)\,(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij})\,B_j^x\cdot\mathbf{r}_{ij} \qquad \text{(B.34)}
\end{aligned}
$$

Similary it can be calculated

$$\mathbf{J}_i = (\boldsymbol{\gamma}_i\cdot\mathbf{r}_i)\,\mathbf{S} - (\gamma_i^x\mathbf{T1} + \gamma_i^y\mathbf{T2} + \gamma_i^z\mathbf{T3}) \qquad \text{(B.35)}$$

Where,

$$\mathbf{S} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{\gamma_j^y}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)B_j^y\cdot\mathbf{r}_{ij} \tag{B.36}$$

$$\mathbf{T1} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{x_j\gamma_j^y}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)x_jB_j^y\cdot\mathbf{r}_{ij} \tag{B.37}$$

$$\mathbf{T2} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{y_j\gamma_j^y}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)y_jB_j^y\cdot\mathbf{r}_{ij} \tag{B.38}$$

$$\mathbf{T3} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{z_j\gamma_j^y}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)z_jB_j^y\cdot\mathbf{r}_{ij} \tag{B.39}$$

Finally $\mathbf{J}_i$ becomes,

$$\begin{aligned}\mathbf{J}_i &= \sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij}\right)\frac{\gamma_j^y}{\sigma_j^5}\cdot\mathbf{r}_{ij}\\ &= \sum_j g2\left(w\right)\left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij}\right)B_j^y\cdot\mathbf{r}_{ij}\end{aligned} \tag{B.40}$$

Similary it can be calculated

$$\mathbf{K}_i = \left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_i\right)\mathbf{S} - \left(\gamma_i^x\mathbf{T1} + \gamma_i^y\mathbf{T2} + \gamma_i^z\mathbf{T3}\right) \tag{B.41}$$

Where,

$$\mathbf{S} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{\gamma_j^z}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)B_j^z\cdot\mathbf{r}_{ij} \tag{B.42}$$

$$\mathbf{T1} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{x_j\gamma_j^z}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)x_jB_j^z\cdot\mathbf{r}_{ij} \tag{B.43}$$

$$\mathbf{T2} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{y_j\gamma_j^z}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)y_jB_j^z\cdot\mathbf{r}_{ij} \tag{B.44}$$

$$\mathbf{T3} = \sum_j g2\left(\mathbf{r}_{ij}^2\right)\frac{z_j\gamma_j^z}{\sigma_j^5}\cdot\mathbf{r}_{ij} = \sum_j g2(w)z_jB_j^z\cdot\mathbf{r}_{ij} \tag{B.45}$$

Finally $\mathbf{K}_i$ becomes,

$$\begin{aligned}\mathbf{K}_i &= \sum_j g2\left(\mathbf{r}_{ij}^2\right)\left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij}\right)\frac{\gamma_j^z}{\sigma_j^5}\cdot\mathbf{r}_{ij}\\ &= \sum_j g2\left(w\right)\left(\boldsymbol{\gamma}_i\cdot\mathbf{r}_{ij}\right)B_j^z\cdot\mathbf{r}_{ij}\end{aligned} \tag{B.46}$$

Therefore **tx** becomes

$$\mathbf{tx} = \frac{3}{4\pi}\left(K_i^y - J_i^z,\ \ I_i^z - K_i^x,\ \ J_i^x - I_i^y\right) \tag{B.47}$$

B.3 Cut-off Function

The following cut-off functions have been used to generate an optimum function table for vortex method calculation with the use of MDGRAPE-2 and MDGRAPE-3.

$$\zeta_0 = -\frac{1}{\sqrt{x}} \tag{B.48}$$

$$\zeta_1 = -\frac{1}{x^{3/2}} \tag{B.49}$$

$$\zeta_2 = -\frac{1}{x^{5/2}} \tag{B.50}$$

$$\zeta_3 = -\frac{x+5/2}{(x+1)^{5/2}} \tag{B.51}$$

$$\zeta_4 = -\frac{x+7/2}{(x+1)^{7/2}} \tag{B.52}$$

$$\zeta_5 = -\frac{15}{2\,(x+1)^{7/2}} \tag{B.53}$$

$$\zeta_6 = -\frac{1}{x^{3/2}} erf\left(\sqrt{x/2}\right) - \sqrt{\frac{x}{cos\theta}}e^{-x/2} \tag{B.54}$$

$$\zeta_7 = -3\ erf(\sqrt{x/2}) - \frac{(x+3)\sqrt{\frac{x}{cos\theta}}e^{-x/2}}{3x^{5/2}} \tag{B.55}$$

$$\zeta_8 = \sqrt{\frac{1}{cos\theta}}e^{-x/2} \tag{B.56}$$

3D cutoff function

Gaussian:

$$f(\rho) = \frac{1}{(2\pi)^{3/2}}e^{-\rho^2/2} \tag{B.57}$$

$$g(\rho) = \frac{1}{4\pi}\left(erf\left(\frac{\rho}{\sqrt{2}}\right) - \sqrt{\frac{2}{\pi}}\rho e^{-\rho^2/2}\right) \tag{B.58}$$

Low Algebra:

$$f(\rho) = \frac{3}{4\pi}\frac{1}{(\rho^2+1)^{5/2}} \tag{B.59}$$

$$g(\rho) = \frac{1}{4\pi}\frac{\rho^3}{(\rho^2+1)^{3/2}} \tag{B.60}$$

High Algebra:

$$f(\rho) = \frac{15}{8\pi}\frac{1}{(\rho^2+1)^{7/2}} \tag{B.61}$$

$$g(\rho) = \frac{1}{4\pi}\frac{\rho^3\,(\rho^2+5/2)}{(\rho^2+1)^{5/2}} \tag{B.62}$$

Appendix C

Calculation Algorithm and Sample Program of MDGRAPE

C.1 Formation of ring

The collision of two identical vortex rings has been used as a test case to validate the present acceleration methods. Here I will introduce initial condition and other parameters which have been used for entire calculations.

Ring1

Outer:

$$
\begin{aligned}
x &= R\ cos\theta + r\ cos\theta\ cos\phi \\
y &= R\ sin\theta\ cos\psi + r\ sin\theta\ cos\phi\ cos\psi + r\ sin\phi\ sin\psi \\
z &= R\ sin\theta\ sin(-\psi) + r\ sin\theta\ cos\phi\ sin(-\psi) + r\ sin\phi\ cos\psi \\
gx &= -gb\ sin\theta \\
gy &= gb\ cos\theta\ cos\psi \\
gz &= gb\ cos\theta\ sin(-\psi)
\end{aligned}
$$

Inner:

$$\begin{aligned} x &= R\ cos\theta \\ y &= R\ sin\theta\ cos(-\psi) \\ z &= R\ sin\theta\ sin(-\psi) \\ gx &= -gb\ sin\theta \\ gy &= gb\ cos\theta\ cos(-\psi) \\ gz &= gb\ cos\theta\ sin(-\psi) \end{aligned}$$

Ring2

Outer:

$$\begin{aligned} x &= R\ cos\theta + r\ cos\theta\ cos\phi \\ y &= R\ sin\theta\ cos\psi + r\ sin\theta\ cos\phi\ cos\psi + r\ sin\phi\ sin(-\psi) \\ z &= R\ sin\theta\ sin\psi + r\ sin\theta\ cos\phi\ sin\psi + r\ sin\phi\ cos\psi \\ gx &= -gb\ sin\theta \\ gy &= gb\ cos\theta\ cos\psi \\ gz &= gb\ cos\theta\ sin\psi \end{aligned}$$

Inner:

$$\begin{aligned} x &= R\ cos\theta \\ y &= R\ sin\theta\ cos\psi \\ z &= R\ sin\theta\ sin\psi \\ gx &= -gb\ sin\theta \\ gy &= gb\ cos\theta\ cos\psi \\ gz &= gb\ cos\theta\ sin\psi \end{aligned}$$

Where

$$gb = \frac{\Gamma}{2\pi\sigma^2} \exp\left(\frac{-r^2}{2\sigma^2}\right) \tag{C.1}$$

$$
\begin{aligned}
R &= Radius\ of\ vortex\ ring \\
r &= Radius\ of\ cross-section\ of\ a\ vortex\ ring \\
dx &= Inter-particle\ space \\
n\theta &= 2R\pi/dx \\
\theta &= 2\pi_i/n\theta \quad (i = 1, 2, \cdots n\theta) \\
nr &= r/dx \\
n\phi &= r/2\pi nr/dx \\
\phi &= 2\pi_k/nj \quad (k = 1, 2, \cdots n\phi) \\
nj &= j \times n\phi \quad (j = 1, 2, \cdots nr) \\
ncr &= \frac{1}{2} n\phi \times nr\,(nr + 1) + 1 \\
nri &= n\theta \times ncr \\
nr &= k + \frac{1}{2} j(j-1)n\phi + (i-1)ncr \\
np &= 2 \times nri \\
\psi &= \pi/6 \\
\sigma &= core\ radius = 2 \times dx \\
sr &= number\ of\ cross\ section \\
\Gamma_0 &= initial\ strength = 1 \cdot 0d0
\end{aligned}
$$

C.2 MDGRAPE Calculation

The vortex method calculation has been accelerated with the use of special-purpose computers; MDGAPE-2 and MDGRAPE-3. There are some critical issues have been solved to make mathematical formulations to call MDGRAPE libraries. Here is a sample FORTRAN program has been used to simulate the collision of two identical vortex rings. Other programs are almost similar except some parameters and communication between Host and MDGRAPEs. The API's of MDGRAPE-2 and MDGRAPE-3 are basically same except the CALLing variable names. Here I will show a sample program of Biot-Savart calculation on MDGRAPE-2.

```
Subroutine for Biot-Savart law calculation for MDGRAPE
subroutine bs3m2(n0,n1,xi,yi,zi,ui,vi,wi, * n2,n3,xj,yj,zj,gxj,gyj,gzj,sj)
implicit real* 8(a-h,o-z)
include 'memory.f'
include 'm2_unit.h'
character skip

dimension xi(npmax), yi(npmax), zi(npmax)
dimension ui(npmax), vi(npmax), wi(npmax)
dimension xj(npmax), yj(npmax), zj(npmax)
dimension gxj(npmax),gyj(npmax),gzj(npmax),sj(npmax)
dimension amd(npmax), bmd(npmax)
dimension bxmd(npmax), bymd(npmax), bzmd(npmax)
dimension xmd(3,npmax),ymd(3,npmax),zmd(3,npmax)
dimension pos(3,npmax),gmd(3,npmax)
common/mdg/amd,bmd,bxmd,bymd,bzmd,xmd,ymd,zmd,pos,gmd

pi = 2* acos(0.0)
do i = n0,n1
    ui(i) = 0
    vi(i) = 0
    wi(i) = 0
end do
ncall = (n3-n2+1)/mdmax+1
do icall = 1,ncall
    iwork1 = (n3-n2+1)/ncall
    iwork2 = mod(n3-n2+1,ncall)
    jsta = (icall-1)*iwork1+n2+min(icall-1,iwork2)
    jend = jsta+iwork1-1
    if(iwork2.gt.icall-1) jend = jend+1
    do i = jsta,jend
        nmd = i-jsta+1
        amd(nmd) = 1/sj(i)** 2
```

```
        pos(1,nmd) = xj(i)
        pos(2,nmd) = yj(i)
        pos(3,nmd) = zj(i)
        bxmd(nmd) = gxj(i)/sj(i)** 3
        bymd(nmd) = gyj(i)/sj(i)** 3
        bzmd(nmd) = gzj(i)/sj(i)** 3
    enddo
    n_unit = m2_allocate_unit('force.table',m2_force, xminf,xmaxf,null_integer)
    call m2_set_positions(n_unit,pos,nmd)
    call m2_set_rscales(n_unit,amd,nmd)
    do i = n0,n1
        nmdd = i-n0+1
        pos(1,nmdd) = xi(i)
        pos(2,nmdd) = yi(i)
        pos(3,nmdd) = zi(i)
    end do
    call m2_set_charges(n_unit,bxmd,nmd)
    call m2_calculate_forces(n_unit,pos,nmdd,xmd)
    call m2_set_charges(n_unit,bymd,nmd)
    call m2_calculate_forces(n_unit,pos,nmdd,ymd)
    call m2_set_charges(n_unit,bzmd,nmd)
    call m2_calculate_forces(n_unit,pos,nmdd,zmd)
    call m2_free_unit(n_unit)
    do i = n0,n1
        nmd = i-n0+1
        ui(i) = ui(i)-0.25/pi* (ymd(3,nmd)-zmd(2,nmd))
        vi(i) = vi(i)-0.25/pi* (zmd(1,nmd)-xmd(3,nmd))
        wi(i) = wi(i)-0.25/pi* (xmd(2,nmd)-ymd(1,nmd))
    end do
end do
return
end
```

Appendix D

Fast Multipole Method with Special-Purpose Computer

D.1 General Idea of FMM

FMM is a numerical method which has found wide acceptance in the scientific community. It is a fast summation method for potentials in $1/r$ and has applications in many areas such as Laplace and Poisson equations, particle simulations, molecular dynamics, etc. Another application of the FMM to vortex method calculations (Cheng et al. [1999]; Yokota et al. [2007]). FMM is an algortihm for achieving fast products of particular dense matrices with vectors. It is similar to the Fast Fourier Transform. FMM achieves product in $O(N)$ or $O(NlogN)$ time and memory. Here I will introduce an example to get an idea of FMM.

If ones want to solve the series as follows

$$\sum_{j=1}^{N} \frac{1}{\mathbf{x_i} - \mathbf{x_j}} = \sum_{j=1}^{N} \frac{1}{\mathbf{x_i} - \mathbf{x}_*} \frac{1}{\left(1 - \frac{\mathbf{x_j} - \mathbf{x}_*}{\mathbf{x_i} - \mathbf{x}_*}\right)} \tag{D.1}$$

and Maclaurin expansion

$$\frac{1}{1-t} = \sum_{k=0}^{p-1} t^k \tag{D.2}$$

gives

$$\sum_{j=1}^{N} \frac{1}{\mathbf{x_i} - \mathbf{x_j}} \longleftarrow N \times N \tag{D.3}$$

It can be seen from Eq. (D.3) that it requires $O(N^2)$ memory for general calculation. It can be discretized as follows.

$$\begin{aligned} \sum_{j=1}^{N} \frac{1}{\mathbf{x_i} - \mathbf{x_j}} &= \sum_{j=1}^{N} \frac{1}{\mathbf{x_i} - \mathbf{x}_*} \left\{ \sum_{k=0}^{p-1} \left(\frac{\mathbf{x_j} - \mathbf{x}_*}{\mathbf{x_i} - \mathbf{x}_*} \right)^k \right\} \\ &= \sum_{k=0}^{p-1} (\mathbf{x_i} - \mathbf{x}_*)^{-k-1} \underbrace{\left\{ \sum_{j=1}^{N} (\mathbf{x_j} - \mathbf{x}_*)^k \right\}}_{Does\ \ not\ \ involve\ \ i} \leftarrow P \times N \end{aligned} \tag{D.4}$$

It is easily observed from Eq. (D.4) that it takes only $O(P \times N)$ time and memory which is lower and faster than Eq. (D.3).

D.2 Hot-spot of FMM calculation

Here I focus on the hot-spot of FMM calculation and the possibility to use MD-GRAPE. Figure D.2 represents the hot spots of FMM calculation. If I want to calculate for light gray box, far particles are solved by FMM and the neighbor particles are solved directly. First step is to calculate Multipole to Multipole (called M2M) translation that is the summation for all particles in each box and translate the multipole expansion to the center of larger boxes. Then it performs the multipole to local (M2L) translation. Here red is source particle and blue is target. Note that M2L cannot be preformed for neighbors. The next step is to translate the local expansion to the center of smaller boxes. Then again preform M2L calculation for the remaining boxes. Calculate the induced velocity up to this step. Many sources acting on one target in this step and this is one of the hot-spots of the FMM calculation. Finally calculate the remaining induced velocity by direct calculation for all particles in the light gray box. If the box is too coarse this could also be the hot-spot of the FMM calculation.

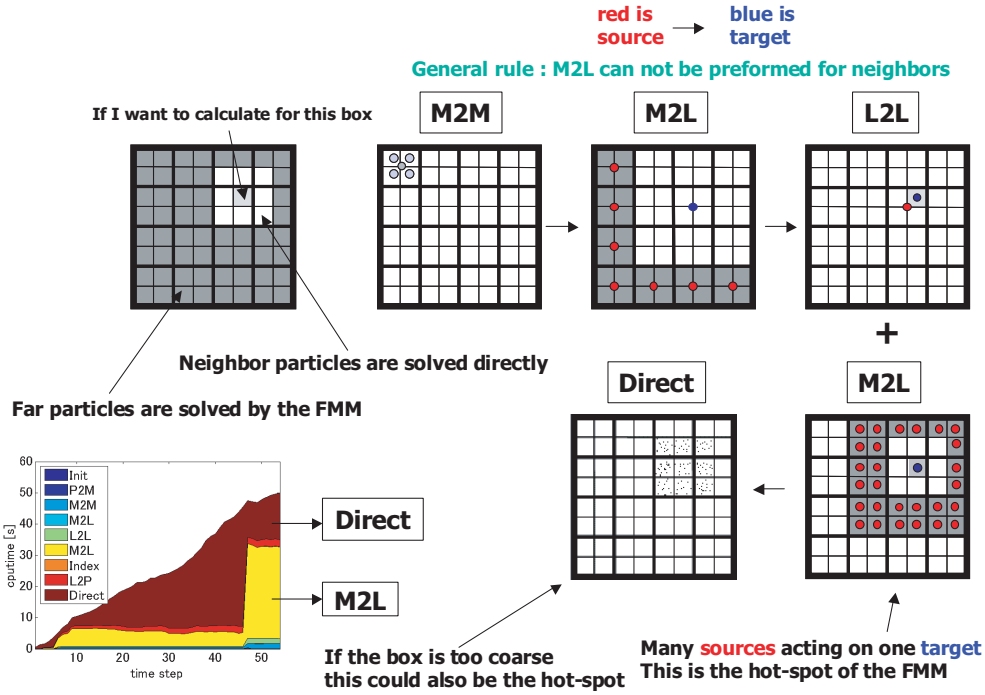


Figure D.1: Hot-spot of FMM calculation

The most time consuming parts of the FMM are the multipole to local (M2L) translation and the direct calculation. The balance between these two steps is dependent on the level of box divisions. Dividing the particles into excessively small boxes will result in an enormous amount of multipole to local translations, whereas not dividing them enough would result in a large amount of direct calculation of neighboring particles. These two steps must be balanced by changing the level of box divisions according to the number of particles being calculated.

The last plot (most left of below) of Fig. D.2 shows the cputime for different steps of FMM calculations. It can be easily observed that the Direct and M2L calculations consumed most of the time. The cputime of the rest steps is negligible. It is necessary to balance the cost of direct and M2L calculations.

From these hot spots of FMM I have decided to apply MDGRAPE to solve the same problems for further acceleration which is the main focus of the present research.

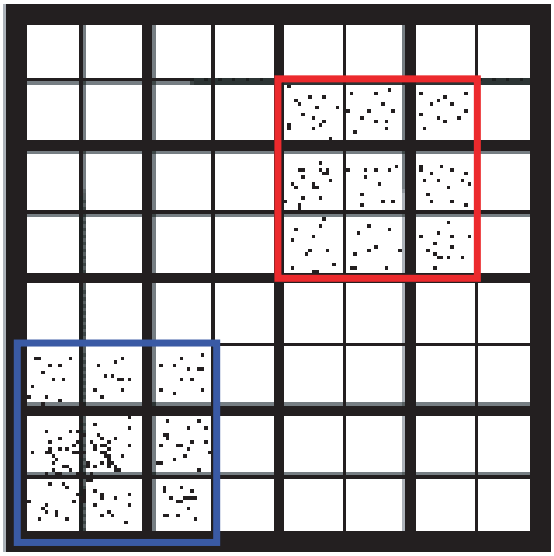


Figure D.2: Direct calculation of FMM in MDGRAPE

D.3 MDGRAPE for direct calculation

Here I will discuss how the MDGRAPE board calculate the direct part of FMM calculation which reduced the total computation cost with the simultaneous use of FMM and MDGRAPE.

In my calculations, I have used the FMM by Cheng et al. [1999]. The most time consuming parts of the FMM are the multipole to local translation and the direct calculation. The balance between these two steps is dependent on the level of box divisions. Dividing the particles into excessively small boxes will result in an enormous amount of multipole to local translations, whereas not dividing them enough would result in a large amount of direct calculation of neighboring particles. These two steps must be balanced by changing the level of box divisions according to the number of particles being calculated.

The mutipole and local expansions and their translations are impossible to calculate on the GRAPE architecture. Therefore, in a straightforward implementation of the FMM, MDGRAPE can only be used for the final step of the FMM where the direct interaction of the particles is calculated (Figure D.3).

In the Figure D.3 the source acting on the target is different for each box

$$\mathbf{f_i} = \sum_{j=1}^{N} b_j g\left(w\right) \mathbf{r_{ij}} \tag{D.5}$$

so j is different for each box in ablove equation. The number of boxes is defined by 8^{level}

D.4 Mathematical Formulations of FMM

The necessary mathematical formulations of FMM have been introduced here. Further details can be found in Carrier et al. [1988]; Schmidt et al. [1991]; Cheng et al. [1999].

The following coordinates and functions are used to derive mathematical formulation of FMM.

The spherical harmonics of degree n and order m according to the formula

$$Y_n^m\left(\theta, \phi\right) = \sqrt{\frac{(n-|m|)!}{(n+|m|)!}} P_n^{|m|}\left(cos\theta\right) e^{im\phi} \tag{D.6}$$

Here, the special functions P_n^m are the associated Legendre functions, which can be defined by Rodrigues' formula

$$P_n^m(x) = (-1)^m \left(1-x^2\right)^{m/2} \frac{d^m}{dx^m} P_n(x) \quad (Associated) \tag{D.7}$$

$$P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} \left(x^2-1\right)^n \quad (Unassociated) \tag{D.8}$$

where $P_n(x)$ denotes the Legendre polynomial of degree n as follows

$$P_n(u) = \frac{4\pi}{2n+1} \sum_{m=-n}^{n} Y_n^{-m}\left(\alpha,\ \beta\right)\ Y_n^m\left(\theta,\ \phi\right) \tag{D.9}$$

Also the recurrence relation and identities of P_n^m are defined as

$$(n-m)\, P_n^m(x) = x\left(2n-1\right) P_{n-1}^m(x) - \left(n+m-1\right) P_{n-2}^m(x) \tag{D.10}$$

$$P_n^m(x) = (-1)^m \left(2m-1\right)! \left(1-x^2\right)^{m/2} \tag{D.11}$$

$$P_{m+1}^m(x) = x\left(2m+1\right) P_m^m(x) \tag{D.12}$$

The fast multipole method requires three transformations of the expansions of the potential. The transformations begin with a multipole expansion of the potential

$$\Phi(X) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \frac{M_n^m}{r^{n+1}} \cdot Y_n^m(\theta,\ \phi), \tag{D.13}$$

where

$$M_n^m = \sum_{i=1}^{N} q_i \cdot \rho_i^n \cdot Y_n^{-m}(\alpha_i,\ \beta_i). \tag{D.14}$$

or a local expansion

$$\Phi(X) = \sum_{j=0}^{\infty} \sum_{k=-j}^{j} L_j^k \cdot Y_j^k(\theta,\ \phi) \cdot r^j, \tag{D.15}$$

where

$$L_j^k = \sum_{l=1}^{N} q_l \cdot \frac{Y_j^{-k}(\alpha_l,\ \beta_l)}{\rho_l^{j+1}}. \tag{D.16}$$

Finally the potential function Φ and three transformations $M2M$, $M2L$, and $L2L$ are defined as follows.

$$\Phi = \frac{1}{|p_i - q_i|} \tag{D.17}$$

$$= \sum_{n=0}^{\infty} \sum_{m=-n}^{n} r_i^{-n-1} Y_n^m(\theta_i,\ \phi_i) \{ \sum_{j=1}^{N} \rho_j^n Y_n^{-m}(\alpha_j,\ \beta_j) \} \tag{D.18}$$

$$= \sum_{n=0}^{\infty} \sum_{m=-n}^{n} r_i^n Y_n^m(\theta_i,\ \phi_i) \{ \sum_{j=1}^{N} \rho_j^{-n-1} Y_n^{-m}(\alpha_j,\ \beta_j) \} \tag{D.19}$$

$$M2M = \sum_{n=0}^{j} \sum_{m=-n}^{n} \frac{O_{j-n}^{k-m} \cdot i^{|k|-|m|-|k-m|} \cdot A_n^m \cdot A_{j-n}^{k-m} \cdot \rho^n \cdot Y_n^{-m}(\alpha,\ \beta)}{(-1)^n A_j^k} \tag{D.20}$$

$$M2L = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \frac{O_n^m \cdot i^{|k-m|-|k|-|m|} \cdot A_n^m \cdot A_j^k \cdot Y_{j+n}^{m-k}(\alpha,\ \beta)}{(-1)^j A_{j+n}^{m-k} \rho^{j+n+1}} \tag{D.21}$$

$$L2L = \sum_{n=j}^{\infty} \sum_{m=-n}^{n} \frac{O_n^m \cdot i^{|m|-|k|-|m-k|} \cdot A_{n-j}^{m-k} \cdot A_j^k \cdot Y_{n-j}^{m-k}(\alpha,\ \beta)\, \rho^{n-j}}{(-1)^{n+j} A_n^m} \tag{D.22}$$

Where

$$A_n^m = \frac{(-1)^n}{\sqrt{(n-m)!\,(n+m)!}}$$

D.4.1 Vortex Methods on FMM

The common form of the Biot-Savart equation is

$$u_i = \sum_j \gamma_j \times \boldsymbol{G}_\sigma \tag{D.23}$$

It can also be written in the form of

$$u_i = \sum_j \gamma_j g_\sigma \times \nabla G \tag{D.24}$$

where $\boldsymbol{G_\sigma} = g_\sigma \nabla G$ and

$$G = \frac{1}{4\pi r_{ij}} \tag{D.25}$$

$$g_\sigma = \frac{r_{ij}^2 + 5/2\sigma^2}{(r_{ij}^2 + \sigma^2)^{5/2}} r_{ij}^3 \tag{D.26}$$

Now using the high order algebraic smoothing function, the stretching term can be written as

$$\begin{aligned} \frac{D\gamma_i}{Dt} &= \sum_{j=1}^{N} \gamma_i \cdot \nabla G_\sigma \times \gamma_j \\ &= \sum_{j=1}^{N} \gamma_i \cdot \nabla (g_\sigma \nabla G) \times \gamma_j \end{aligned} \tag{D.27}$$

The Eqs. (D.24) and (D.27) are calculated using the FMM in order to reduce the complexity from $O(N^2)$ to $O(N)$. G is the Green's function of the Laplace equation, which is defined in Eq. (D.25). g_σ is the cutoff function, which is defined in Eq. (D.26). For the FMM equations, I will adopt the conventions used in Cheng et al. [1999]. By doing so, the Green's function can be expressed by the multipole expansion

$$\sum_j^N G \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \underbrace{r_i^{-n-1} Y_n^m(\theta_i, \phi_i)}_{S_i} \left\{ \sum_{j=1}^{N} \underbrace{\rho_j^n Y_n^{-m}(\alpha_j, \beta_j)}_{M_j} \right\} \tag{D.28}$$

and the local expansion

$$\sum_j^N G \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \underbrace{r_i^n Y_n^m(\theta_i, \phi_i)}_{R_i} \left\{ \sum_{j=1}^{N} \underbrace{\rho_j^{-n-1} Y_n^{-m}(\alpha_j, \beta_j)}_{L_j} \right\}. \tag{D.29}$$

I define the operators S, M, R, L to simplify the equations in the following steps. Using these operators, Eq. (D.24) can be written as

$$\mathbf{u}_i \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \left\{ \sum_{j=1}^{N} \boldsymbol{\gamma}_j M_j \right\} \times \nabla S_i \tag{D.30}$$

$$\mathbf{u}_i \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \left\{ \sum_{j=1}^{N} \boldsymbol{\gamma}_j L_j \right\} \times \nabla R_i. \tag{D.31}$$

Similarly, Eq. (D.27) can be written as

$$\frac{D\boldsymbol{\gamma}_i}{Dt} \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \left\{ \sum_{j=1}^{N} \boldsymbol{\gamma}_j \times \nabla M_j \right\} (\boldsymbol{\gamma}_i \cdot \nabla S_i) \tag{D.32}$$

$$\frac{D\boldsymbol{\gamma}_i}{Dt} \approx \frac{1}{4\pi} \sum_{n=0}^{p} \sum_{m=-n}^{n} \left\{ \sum_{j=1}^{N} \boldsymbol{\gamma}_j \times \nabla L_j \right\} (\boldsymbol{\gamma}_i \cdot \nabla R_i). \tag{D.33}$$

The cutoff function does not appear in these equations since they are used to calculate the effect of the far field, for which it would have negligible effect. For details mathematical formulations, see Yokota et al. [2007]

D.5 Sample ForTran Code

A sample fortran program of Biot-Savart law calculation when the FMM implement on MDGRAPE-3. Here only MDGRAPE-3 API's are presented in details and the other parts of FMM calculations are similar as standard 3D FMM calculations.

```
subroutine bs5(n0,n1,n2,n3,mp,nge,tfmm,npb)
implicit real*8(a-h,o-z)
include 'm3_unit.h'
```

```
dimension nlev(9),tfmm(9)
C Initialization
C Box structure
C Step 1. S-expansion
    Calculation for multipole expansion
C Step 2. S—S-translation
    Calculation for multipole to multipole translation (M2M)
C Step 3. S—R-translation
    Calculation for multipole to local translation (M2L)
C Step 4. R—R-translation
    Calculation for local to local translation (M2L)
C S—R-translation
    Calculation for multipole to local translation (M2L) again
C Step 5. R-expansion and Final summation
C This part has been calculated by MDGRAPE-3
nmd = 0
nbase = n0
nicall = 1
ista(1) = 1
do i = 1,lbi
    call e2b(nfi(i),lbj,li,lev)
    do j = 1,li
        nnjm(j,i) = nnj(j)
    end do
    lim(i) = li
    nmd = nmd+ndi(i,2)-ndi(i,1)+1
    if(nmd.gt.mimax)then
        nbase = ndi(i,1)
        nicall = nicall+1
        iend(nicall-1) = i-1
        ista(nicall) = i
        nmd = ndi(i,2)-ndi(i,1)+1
    end if
```

```
    ibase(i) = ndi(i,1)-nbase+1
    isize(i) = ndi(i,2)-ndi(i,1)+1
end do
iend(nicall) = lbi
nmd = 0
njcall = 1
jsta(1) = 1
do i = 1,lbj
    nmd = nmd+ndj(i,2)-ndj(i,1)+1
    if(nmd.gt.mjmax)then
        njcall = njcall+1
        jend(njcall-1) = i-1
        jsta(njcall) = i
        nmd = ndj(i,2)-ndj(i,1)+1
    end if
end do
jend(njcall) = lbj
do jcall = 1,njcall
    ic = 0
    do i = jsta(jcall),jend(jcall)
        jfil(i,1) = ic
        n = ndj(i,2)-ndj(i,1)+1
        icn = 0
        if(n.lt.12)then
            ic = ic+12-n
            icn = 12-n
        end if
        jfil(i,2) = icn
    end do

    do i = 1,lbi
        ic = 0
        do ij = 1,lim(i)
```

```
            do j = jsta(jcall),jend(jcall)
                if(j.eq.nnjm(ij,i))then
                    ic = ic+1
                    jbase(ic,i) = ndj(j,1)-ndj(js ta(jcall),1)+jfil(j,1)
                    jsize(ic,i) = ndj(j,2)-ndj(j, 1)+1+jfil(j,2)
                end if
            end do
        end do
        njsize(i) = ic
    end do
nmd = 0
do i = jsta(jcall),jend(jcall)
    do j = ndj(i,1),ndj(i,2)
        nmd = nmd+1
        pos(1,nmd) = xj(j)/sj(1)
        pos(2,nmd) = yj(j)/sj(1)
        pos(3,nmd) = zj(j)/sj(1)
        bxmd(nmd) = gxj(j)/sj(1)**2
        bymd(nmd) = gyj(j)/sj(1)**2
        bzmd(nmd) = gzj(j)/sj(1)**2
    end do
    do j = 1,jfil(i,2)
        nmd = nmd+1
        pos(1,nmd) = 0.0d0
        pos(2,nmd) = 0.0d0
        pos(3,nmd) = 0.0d0
        bxmd(nmd) = 0.0d0
        bymd(nmd) = 0.0d0
        bzmd(nmd) = 0.0d0
    end do
end do
n_unit = m3_allocate_unit('force.table',m3_force, xminn,xmaxn,null_integer)
call m3_set_positions(n_unit,pos,nmd)
```

```
do icall = 1,nicall
    do i = ndi(ista(icall),1),ndi(iend(icall),2)
        nmdd = i-ndi(ista(icall),1)+1
        pos(1,nmdd) = xi(i)/sj(1)
        pos(2,nmdd) = yi(i)/sj(1)
        pos(3,nmdd) = zi(i)/sj(1)
    end do
    call m3_set_charges(n_unit,bxmd,nmd)
    do i = ista(icall),iend(icall)
        if(njsize(i).ne.0)then
            call m3_set_cells(n_unit,jbase(1,i),j size(1,i),njsize(i))
            call m3_calculate_forces(n_unit,pos(1 ,ibase(i)),isize(i), xmd(1,ibase(i)))
        else
            do j = ibase(i),ibase(i)+isize(i)-1
                do k = 1,3
                    xmd(k,j) = 0.0d0
                end do
            end do
        end if
    end do
    call m3_set_charges(n_unit,bymd,nmd)
    do i = ista(icall),iend(icall)
        if(njsize(i).ne.0)then
            call m3_set_cells(n_unit,jbase(1,i),j size(1,i),njsize(i))
            call m3_calculate_forces(n_unit,pos(1 ,ibase(i)),isize(i), ymd(1,ibase(i)))
        else
            do j = ibase(i),ibase(i)+isize(i)-1
                do k = 1,3
                    ymd(k,j) = 0.0d0
                end do
            end do
        end if
    end do
```

```
    call m3_set_charges(n_unit,bzmd,nmd)
    do i = ista(icall),iend(icall)
        if(njsize(i).ne.0)then
            call m3_set_cells(n_unit,jbase(1,i),j size(1,i),njsize(i))
            call m3_calculate_forces(n_unit,pos(1 ,ibase(i)),isize(i), zmd(1,ibase(i)))
        else
            do j = ibase(i),ibase(i)+isize(i)-1
                do k = 1,3
                    zmd(k,j) = 0.0d0
                end do
            end do
        end if
    end do
    do i = ndi(ista(icall),1),ndi(iend(icall),2)
        nmdd = i-ndi(ista(icall),1)+1
        ui(i) = ui(i)-2.5d-1/pi*(ymd(3,nmdd)-zmd(2,nmdd))
        vi(i) = vi(i)-2.5d-1/pi*(zmd(1,nmdd)-xmd(3,nmdd))
        wi(i) = wi(i)-2.5d-1/pi*(xmd(2,nmdd)-ymd(1,nmdd))
    end do
end do
call m3_free_unit(n_unit)
end do
return
end
```

Bibliography

Anderson CR. An Implementation of the Fast Multipole Method without Multipoles, SIAM J. Sci. Stat. Comput., 1992;13:923-947. 21, 55, 90

Anderson CR, Greengard C. On vortex methods, SIAM J. Numer. Anal., 1986;22;413-429. 1, 2, 11

Appel AW. An efficient program for many-body simulation, SIAM J. Sci. Stat. Comput., 1985;6:85- . 17

Athanassoula E, Bosma A, Lambart JC, Makino J. Performance and accuracy of a GRAPE-3 system for collisionless N-body simulations, Monthly Notices of the Royal Astronomical Society, 1998;293:369-380. 18

Barba LA. Vortex Method for computing high-Reynolds number flows: Increased accuracy with a fully mesh-less formulation, PhD Thesis, California Institute of Technology, 2004. 1, 11

Barba LA, Leonard A, Allen CB. Advances in Viscous Vortex Methods- Meshless Spatial Adaption Based on Radial Basis Function Interpolation, Int. J. Num. Meth. Fluids 2005;47(5):387-421. 14, 16

Barba LA. Discussion: "Three-Dimensional Vortex Method for Gas-Particle Two-Phase Compound Round Jet"(Uchiyama, T., and Fukase, A., 2005, ASME J. Fluids Eng., 127, pp. 32-40), ASME J. Fluids Eng., 2006;128:643-645. 14

Barnes J, Hut P. A hierarchical $O(NlogN)$ force calculation algorithm. Nature 1986;324:446-449. 4, 11, 17

Barnes JE. A modified tree code: Don't laugh; It runs, J. Comput. Phys., 1990;87:161-170. 11

Beale JT, Majada A. Rates of convergence for viscous splitting of the Navier-Stokes equations, Math Comput, 1981;37:243-59. 2

Beale JT, Majada A. Vortex methods I: Convergence in three dimensions, Math Comput, 1982;39:1-27. 2

Bedorf J. High Performance Direct Gravitational N-body Simulations on Graphics Processing Units (An implementation in CUDA), MSc Thesis, Universiteit van Amsterdam, November 16, 2007. 6

Belleman RG, Be/dorf J, Zwart SFP. High performance direct Gravitational N-body simulations on graphics processing units II: An implementation in CUDA, New Astronomy, 2008;13(2):103-112. 6

Board JA, Hakura ZS Jr., Elliott WD, Rankin WT. Scalable Variants of multipole-based algorithms for molecular dynamics applications, Proceedings of the 7th SIAM conference on parallel processing for scientific computing (edited by Bailey et al.), SIAM, Philadelphia, 1995;295-300. 19

Board J, Schulten K. The fast multipole algorithm, Comp. Sci. Eng., 2000;2(1):76-79. 19

Carrier J, Greengard L, Rokhlin V. A fast adaptive multipole algorithm for particle simulations, SIAM J Sci. Stat. Comput., 1988;9:669-86. 150

Chatelain P, Leonard A. Face-centred cubic lattices and particle redistribution in vortex methods, J. Turb., 2002;3:046. 14

Chatelain P, Kivotides D, Leonard A. Reconnection of Colliding Vortex Rings, Phys. Rev. Lett. 2003;90(5):054501-1. 8, 76, 82, 119

Chatelain P. Contributions to the Three-Dimensional Vortex Element Method and Spinning Bluff Body Flows, PhD Thesis, California Institute of Technology, 2005. 12

Chau NH, Kawai A, Ebisuzaki T. Implementation of Fast Multipole Algorithm on Special-Purpose Computer MDGRAPE-2, PProc. Of the 6th World Multiconference on Systematics, Cybernetics and Informatics SCI 2002, July 14-18, 2002, Orlando, Colorado, USA, Vol. XVI, pp. 477-481. 8, 88, 90

Chau NH, Kawai A, Ebisuzaki T. A new Implementation of Fast Multipole Algorithm on Special-Purpose Computer MDGRAPE-2; Proceedings of second meeting on simulation and modeling physics, Hanoi, Nov 28-29, 2002. 8, 88, 90

Cheng H, Greengard L and Rokhlin V. A Fast Adaptive Multipole Algorithm in Three Dimensions, J. Comp. Phys 1999;155:468-498. 3, 11, 19, 21, 54, 55, 88, 89, 146, 149, 150, 152

Chorin AJ. Numerical Study of Slightly Viscous Flow, J. Fluid Mech. 1973;57:785-96. 1, 14

Chorin AJ. On the convergence of discrete approximations to the Navier-Stokes equations. SIAM J Sci Stat Comput, 1978;27:341-53. 2

Chorin AJ. Vortex Methods, PAM Report 593, Department of Mathematics, University of California, Berkeley, 1993 11

Collins JP, Dimas AA, Bernard PS. A Parallel Adaptive Fast Multipole Method for High Performance Vortex Method Based Simulations, Proc. ASME FED Vol. 250, 1999 4

Cottet GH, Koumoutsakos PD. Vortex Methods (Theory and Practice), Cambridge University Press, 2000. 1, 2, 119

Cottet GH, Michaux B, Ossia S, VanderLinden G. A Comparison of Spectral and Vortex Methods in Three-Dimensional Incompressible Flows, J. Comp. Phys 2002;175:702-712. 4, 8

Degond P, Mas-Gallic S. The Weighted Particle Method for Convection-Diffusion Equations. Part 1: The Case of an Isotropic Viscosity, Math. Comput., 1989;53(188):485-507. 2, 14

Elsen E, Vishal V, Houston M, Pande V, Hanrahan P, Darve E. N-Body Simulations on GPUs, 2007. 6

Elmegreen BG, Koch R, Yasuoka K, Furusawa H, Narumi T, Susukita R, Ebisuzaki T. Numerical simulations of magnetic materials with MD-GRAPE: curvature induced anisotrophy. J Magnetism and Magnetic Materials 2002;250:39-48. 9, 59

Elmegreen BG, Koch R, Schabes ME, Crawford T, Ebisuzaki T, Furusawa H, Narumi T, Susukita R, Yasuoka K. Simulations of magnetic materials with MDGRAPE-2. IBM J RES. & DEV. 2004;48:199-207. 9, 59

Fishelov D. A new vortex scheme for viscous flows, J. Comput. Phys., 1990;86:211-24. 2

Fukuda K, Kamemoto K. Application of a Redistribution Model Incorporated in a Vortex Method to Turbulent Flow Analysis, Proc. ICVFM2005, Yokohama, Japan, pp. 131-136. 116

Fukushige T, Taiji M, Makino J, Ebisuzaki T, Sugimoto D. A highly parallelized special-purpose computer for many-body simulations with an arbitrary central force: MD-GRAPE, Astrophys. J., 1996;468:51. 24

Gingold RA, Monaghan JJ. Smoothed Particle Hydrodynamics: Theory and Application to Non-Sperical Stars, Mon. Not. R. Astr. Soc., 1977;181:375-389.

Grant JR, Marshall JS. Diffusion Velocity for a Three-Dimensional Vorticity Field, Theor. Comput. Fluid Dyn., 2005;19(6):377-390. 15

Greengard L, Rokhlin V. A Fast algorithm for particle simulations. J Comput Phys 1987;73:325-348. 5, 11, 18

Greengard L, Rokhlin V. Rapid evaluation of potential fields in three dimensions, Vortex Methods (edited by C. Anderson and C. Greengard), number 1360 in Lecture Notes in Mathematics (1998), Springer-Verlag, Berlin, 121-141. 18

Greengard L, Rokhlin V. A new version of the fast multiplole method for the Laplace equation in three dimensions, Acta Numerica, 1997;6:229-269. 19

Gumerov NA, Duraiswami R. Fast Multipole Methods for the Helmholtz Equation in Three Dimensions, Elsevier (2004). 19, 21

Hald OH. Convergence of Vortex Methods for Euler's Equations III, SIAM J. Numer. Anal., 1987;24:538-82. 16

Hardin RH, Sloane NJA. McLaren's Improved Snub Cube and Other New Spherical Design in Three Dimensions, Discrete Comput. Geom., 1996;15:429-441. 21, 90

Hernquist L. Performance characteristics of tree codes, The Astrophys. J. Suplement Series, 1987;64:715-34. 11, 18

Hernquist L and Katz N. TREESPH - a unificationof SPH with the hierarchical tree method, Astrophysical Journal Supplement, 1989;70:419-46. 11, 18

Hess JL. Panel Methods in Computational Fluid Dynamics, Annu. Rev. Fluid Mech., 1990;22:255-274 2

Hockney RW, Eastwood JW. Computer Simulations Using Particles. New York, McGraw-Hill, 1981. 4

Hockney RW, Goel SP, Eastwood JW. A 10000 particle molecular dynamics model with long range forces, Chem. Phys. Lett., 1973;21:589-91. 4

Hu y, Jonsson SL, Teng SH. A data-parallel adaptive N-body method, Proceedings of the 8th SIAM conference on parallel processing for scientific computing, SIAM, 1997, (CD-ROM). 19

Huang MJ. Diffusion via Splitting and Remeshing via Merging in Vortex Methods, Int. J. Num. Meth. Fluids 2005;48(2):521-536 14

Ito T, Makino J, Fukushige T, Ebisuzaki T, Okumura SK, Sugimoto D. A special-purpose computer for N-body simulations: GRAPE-2A, Publ. Astron. Soc. Japan, 1993;45:339. 24

Ito T, Makino J, Fukushige T, Ebisuzaki T, Okumura SK, Sugimoto D, Miyagawa H, Kitamura K. A special-purpose computer for molecular dynamics: GRAPE-2A, Proteins, 1994;20:139. 24

Kawai A, Makino J. Pseudoparticle Multipole Method: A simple method to implement a high-accuracy tree code; The Astrophysical Journal, 2001;550:L143-L146. 22

Kawai A, Makino J, Ebisuzaki T. Performance Analysis of High-Accuracy Tree Code Based on the Pseudoparticle Multipole Method. The Astrophysical Journal Suplement Series 2004;151:13-33. 8, 88

Kida S, Takaoka M. Vortex Reconnection. Annu. Rev. Fluid Mech. 1994;26:169-189. 119

Komeiji Y, Uebayashi M, Takata R, Shimuzu A, Itsukashi K, Taiji M. Fast and accurate molecular dynamics simulation of a protein using a special-purpose computer, J. Comput. Chem., 1997;18:1546. 24

Koshizuka S, Tamako H, Oka Y. A Particle Method for Incompressible Viscous Flow with Fluid Fragmentation, Comput. Fluid Dyn. J., 1995;4:29-46. 15

Koumoutsakos P, Leonard A. High-Resolution Simulations of the Flow Around an Impulsively Started Cylinder Using Vortex Methods. J. Fluid Mech. 1995;296:1-38. 1

Kuwahara K, Takami H. Numerical Studies of Two-Dimensional Vortex Motion by a System of Point Vortices, J. Phys. Soc. Japan, 1973;34(1):247-253. 2, 14

Leonard A. Vortex Methods for Flow Simulations. J Comput Phys 1980;37:289-335. 1, 2, 11, 14, 78

Leonard A. Computing Three-Dimensional Incompressible Flows with Vortex Elements, Ann. Rev. Fluid Mech. 1985;17:523-59. 2, 11

Liu S, Wang Z, Gong Z, Peng Q. Real time simulation of a tornado, The Visual Computer, 2007;8:559-567. 6

Liu CH, Doorly DJ. Vortex particle-in-cell method for three-dimensional viscous undbounded flow computations, Int. J. Numer. Meth. Fluids, 2000;32:29-50. 7, 23

Makino J. Treecode with a special-purpose processor. Pub of the Astronomical Society of Japan 1991;43:621-638 8, 18, 88

Makino J, Taiji M. Scientific Simulations with Special-Purpose Computers - the GRAPE systems. John Wiley & Sons Ltd., England, 1998. 6

Makino J. Yet another fast multipole method without multipoles-Pseudo-particle multipole method. J Comput Phys 1999;151:910-920. 21, 22, 55, 88, 90

Mammetti M, Verzicco R, and Orlandi P. The study of vortex ring/wall interaction for artificial nose improvement, ESAIM: Proceedings, 1999;7:258-269. 8

Mansfield JR, Knio MO, Meneveau C. Dynamic LES of Colliding Vortex Rings Using a 3D Vortex Method. J Comput Phys 1999;152:305-345. 1, 80

Moin P. Fundamentals of Engineering Numerical Analysis, Cambridge University Press, Cambridge, UK, 2001. 78, 109

Narumi T. Special-Purpose Computer for Molecular Dynamics Simulations, Doctoral Thesis, College of Arts and Sciences, University of Tokyo, 1997. xiii, 25, 30

Narumi T, Kawai A and Koishi T. An 8.61 Tflop/s molecular dynamics simulation for NaCI with a special-purpose computer: MDM, Proc. Supercomputing 2001 (2001), in CD-ROM. 25

Narumi T, Ohno Y, Okimoto N, Koishi T, Suenaga A, Futatsugi N, Yanai R, Himeno R, Fujikawa S, Ikei M, and Taiji M. A 55 TFLOPS Simulation of Amyloid-forming Peptides from Yeast Prion Sup35 with the Specialpurpose Computer System MDGRAPE-3, Proceedings of the SC06 (High Performance Computing, Networking, Storage and Analysis), CDROM, Tampa, USA, Nov 11-17, 2006 6, 9, 24, 40, 49, 88

Ogami Y, Akamatsu T. Viscous Flow Simulation Using the Discrete Vortex Model-the Diffusion Velocity Method, Comput. Fluids, 1991;19:433-441. 14

Orszag SA, Patterson GS. Numerical simulation of three-dimensional homogeneous isotropic turbulence, Phys. Rev. Lett., 1972;28:76-79. 7

Pfalzner S, and Gibbon P. Many-body tree methods in physics, Cambridge University Press, New York, 1996. 18

Ploumhans P, Winckelmans GS, Salmon JK, Leonard A, and Warren MS. Vortex Methods for Direct Numerical Simulation of Three-Dimensional Bluff Body Flows: Application to the Sphere at Re = 300, 500, and 1000. J. Comp. Phys. 2002;178:427-463. 3, 5, 6, 23

Puckett EG. Vortex Methods: An introduction and survey of selected research topics, in Incompressible computational Fluid Dynamics, edited by M. D. Gunzburger and R. A. Nicolaides, (Cambridge University Press, Cambridge, 1993), pp. 335-407. 11

Raviart PA. An Analysis of Particle Methods, in *Numerical Methods in Fluid Dynamics*, Lecture Notes in Math., Vol. 1127, edited by F. Brezzi, (Springer-Verlag, New York/Berlin, 1985), pp. 243-324. 11

Rogallo RS. Numerical experiments in homogeneous turbulence, NASA TM-81315 (1981). 7

Rokhlin V. Rapid solution of integral equations of classical potential theory, J. Comput. Phys., 1985;60:187-207. 18

Rossi LF. Resurrecting Core Spreading Vortex Methods: A New Scheme that is Both Deterministic and Convergent, SIAM J. Sci. Comput. 1996;17(2):370-397. 14

Rosenhead L. The Formation of Vorticies from a Surface of Discontinuity, Proc. Roy. Soc. London Ser. A, 1931;134:170-192. 1

Salmon JK, Winckelmans GS, Warren MS. Fast parallel tree codes for gravitational and fluid dynamical N-body problems, International J. Super. Computer Applications, 1994;8:129-43. 3, 18

Saito M. Molecular dynamics simulations of proteins in water without truncation of long-range Coulomb interactions, Mol. Sim., 1992;8:321-33. 24

Sbalzarini IF, Walther JH, Bergdorf M, Hieber SE, Kotsalis EM, Koumoutsakos P. PPM - A Highly Efficient Parallel Particle-Mesh Library for the Simulation of Continuum Systems. J. Comp. Phys. 2006;215:566-588. 4, 23

Schmidt KE, Lee MA. Implementing the Fast Multipole Method in Three Dimensions; J. Stat. Phys., 1991;63:1223-1235. 11, 19, 150

Shankar S. A new mesh-free vortex method. Ph.D. Thesis, The Florida State University, 1996 (unpublished). 1, 12

Shankar S, Dommelen LV. A New Diffusion Procedure for Vortex Methods, J. Comput. Phys., 1996;127(2):88-109. 14

Shariff K, Leonard A. Vortex Rings, Annu. Rev. Fluid Mech. 1992;24:235-79. 7

Shariff K, Verzicco R, Orlandi P. A Numerical Study of Three-Dimensional Vortex Ring Instabilities: Viscous Corrections and Early Nonlinear Stage. J. Fluid Mech. 1994;279:351-75. 107, 113

Sheel TK, Yasuoka K, Obi S. Fast vortex method calculation using a special-purpose computer, Computers and Fluids, 2007;36:1319-26. 9, 28, 59, 107

Sheel TK, Yokota R, Yasuoka K, Obi S. The study of colliding vortex rings using a special-purpose computer and FMM, Transactions of the Japan Society for Computational Engineering and Science, in press. 10

Stanely YC Li, Cheuk Gap CK, Lee KH, and Leong Philip HW. FPGA-based SIMD processor, Proc. 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'03). 6

Stock MJ, Summary of vortex methods literature (A living document rife with opinion), *http* : *//mark.technolope.org/research/vortex$_m$ethods$_l$iterature.pdf* 3

Stock MJ, Gharakhani A. Toward efficient GPU-accelerated N-body simulations, 46th AIAA Aerospace Sciences Meeting and Exhibit, 7-10 January 2008, Reno, Noveda, AIAA 2008-608. 6

Sugimoto D, Chikada Y, Makino J, Ito T, Ebisuzaki T, Umemura M. A special-purpose computer for gravitational many-body problems. Nature 1990;345:33-35. 5, 6, 88

Susukita R, Ebisuzaki T, Elmegreen B G, Furusawa H, Kato K, Kawai A, Kobayashi Y, Koishi T, McNiven GD, Narumi T, Yasuoka K. Hardware accelerator for molecular dynamics: MDGRAPE-2. Comput Phys Comm 2003;155:115-131. 5, 24, 25, 30, 35

Taiji M, Narumi T, Ohno Y, Futatsugi N, Suenaga A, Takada N, Konagaya A. Protein Explorer: A Petaflops Special-Purpose Computer System for Molecular Dynamics Simulations, Proc. Supercomputing 2003, in CD-ROM. xiii, 5, 24, 40, 41, 42

Takahashi T, Kawai A, Ebitsuzaki T. Accelerating Boundary Integral Equation Method using a Special-purpose Computer, Int. J. Numer. Meth. Eng., 2006;66(3):529-48. 6

Totsuka Y, Obi S. Viscous Dissipation Model for Fast Vortex Method in Simulation of Decaying Turbulence. Transactions of the Japan Society of Mechanical Engineers (in Japanese) 2005;71 (701):23-29. 87

Totsuka Y, Obi S. A validation of the viscous dissipation model for fast vortex methods in simulations of decaying turbulence, J. Fluid Science and Technology, 2007;2:248-57. 87

Toyoda S, Miyagawa H, Kitamura K, Amisaki T, Hashimoto E, Ikeda H, Kusumi A, Miyakwa N. Development of MD Engine: high-speed accelerator with parallel processor design for molecular dynamics simulations, J. Comput. Chem., 1999;20:185. 24

Warren MS and Salmon JK. A portable parallel particle program, Computer Physics Communications, 1995;87:266-90. 5

Warren MS and Salmon JK. A fast tree code for many-body problems, Los Alamos Science, No. 22, 1994. 3, 5, 18

Winckelmans GS, Leonard A. Contributions to Vortex Particle Methods for the Computation of Three-Dimensional Incompressible Unsteady Flows. J Comput Phys 1993;109:247-273. 7, 13, 16, 76, 78, 80, 87, 109, 112, 116, 128

Winckelmans GS, Salmon JK, Warren MS, Leonard A, Jodoin B. Application of Fast Parallel and Sequential Tree Codes to Computing Three-Dimensional Flows with the Vortex Element and Boundary Element Methods, ESAIM Proceedings, 1996;1:225-40. 3, 6

Winckelmans GS. "Vortex Methods", The Encyclopedia of Computational Mechanics, Edited by Erwin Stein, Rene' de Borst and Thomas J. R. Hughes. Vol. 3: Fluids, 2004:Chapter 5. xvi, 16

Yatsuyanagi Y, Ebisuzaki T, Hatori T, Kato T. Filamentary magnetohydrodynamic simulation model, current vortex method. Physics of Plasmas 2003;10:3181-3187. 8, 9, 24, 58, 59

Yatsuyanagi Y, Kiwamoto Y, Ebisuzaki T, Hatori T, Kato T. Simulation of diocotron instability using a special-purpose computer, MDGRAPE-2. Physics of Plasmas 2003;10:3188-3195. 9, 59

Yokota R, Sheel TK, Obi S, Calculation of Isotropic Turbulence Using Pure Lagrangian Vortex Method, Journal of Computational Physics, 2007;226:1589-1606. 16, 19, 54, 146, 153