Comparing the Treecode with FMM on GPUs for vortex particle simulations of a leapfrogging vortex ring

Rio Yokota *, **L. A. Barba** **

* *Department of Mathematics, University of Bristol, Bristol, U.K.*
** *Department of Mechanical Engineering, Boston University, U.S.A.*

Abstract: We compare the performance of the treecode and the fast multipole method (FMM) on multiple GPUs. These fast algorithms are used to accelerate a vortex particle simulation of two leapfrogging vortex rings. The performance of the treecode and FMM depends strongly on the number of particles, type of hardware, distribution of particles, and the required accuracy. Our results demonstrate that the treecode is faster for less accurate simulations with non-uniform particle distributions, while the FMM is faster for high accuracy simulations with a large number of evenly distributed particles.
Keywords: Vortex Methods; Fast Multipole Methods; Treecodes; GPU clusters; Vortex Rings

1 INTRODUCTION

When two coaxial vortex rings travel in the same direction, it can lead to a process known as the "leapfrogging" of the vortex rings. During this process, the induced velocity of the front ring will cause the rear ring to contract and accelerate. At the same time, the induced velocity of the rear ring causes the front ring to widen and decelerate. Eventually, the rear ring may catch up to the front ring, and be drawn through its center to emerge ahead of the front ring. If the condition is favorable, it is possible for the process to repeat again and again.

Depending on the initial core size and Reynolds number of the vortex rings, the two rings could merge [1, 2] or remain leapfrogging for a few cycles[3, 4]. Shariff *et al.* [5] performed a dynamical systems analysis of the leapfrogging vortex rings and showed that the unstable manifold exhibits striking agreement with even the fine features of smoke visualization photographs, suggesting that fluid elements in the vicinity of the manifold are drawn out along it and begin to reveal its structure. They also predicted that simulations at a higher Reynolds number will lead to a synthesis of inviscid descriptions and Maxworthy's views about the role of viscous diffusion in leapfrogging vortex rings.

For the simulation of leapfrogging vortex rings, the vorticity is confined to a finite region near the rings, and the flow is dominated by unsteady vortical motion of the vortex rings. Such flows have been known to be handled effectively by vortex methods, and several recent applications of high-quality vortex methods have focused on vortex rings [6, 7, 8]. Vortex methods solve the Navier-Stokes equation in the velocity-vorticity form with Lagrangian discretization. Since vortex particles are initially placed only in the region of finite vorticity and can convect along with the vortex ring, it results in an optimized spatial distribution of particles in the field. The Lagrangian treatment of the convection frees the vortex method from numerical stability conditions and numerical diffusion.

The use of particle methods results in an N-body problem. The traditional approach to accelerate N-body problems has been the use of hierarchical algorithms, such as the Barnes & Hut treecode [9], the fast multipole method (FMM) [10], and particle-particle particle-mesh methods (PPPM) [11]. For vortex methods in particular, the FMM [12] and PPPM [13, 14] have been the popular choices.

A more recent trend in the field of vortex methods has been the efficient implementation of such fast N-body algorithms on GPUs. Stock & Gharakhani [15] implemented the treecode on the GPU to accelerate their vortex method calculation. Similarly, Yokota *et al.*[16] realized vortex method simulations of isotropic turbulence with the FMM on GPUs. Rossinelli *et al.* [17] implemented the particle-mesh method on the GPU for their vortex method calculation of a circular cylinder flow.

Since the treecode algorithm has been traditionally used by the astrophysics community, the FMM by the vortex method community, and PPPM by the molecular dynamics community, there is a lack of consensus regarding the relative performance of these methods. Recently, with the introduction of GPUs as a tool for high performance computing, the selection of the fastest method has become even more unclear. As an effort to better understand the relative performance of these methods for different conditions, in the present study we apply the treecode and fast multipole method on a

cluster of GPUs for the calculation of a leapfrogging vortex ring using vortex methods. We believe these types of comparisons are of value to researchers considering the various algorithms available for their applications, and clear some misconceptions that exist because of opinions voiced by scientists about their favorite method that are not backed by experiments.

2 SUMMARY OF THE NUMERICAL METHODS

2.1 Vortex particle method

Since the vortex method is neither a standard method for simulating turbulent flows nor a standard application for FMMs, we will give a brief explanation of the method itself. In the vortex method, the Navier-Stokes equation is solved in the velocity-vorticity form, and discretized with a particle approximation. The velocity is calculated by

$$\mathbf{u}_i = \sum_{j=1}^{N} \boldsymbol{\alpha}_j g_\sigma \times \nabla G, \tag{1}$$

where $\boldsymbol{\alpha}_j$ is the strength of vortex particle j, $G = 1/4\pi r_{ij}$ is the Green's function for the Laplace equation, and

$$g_\sigma = \operatorname{erf}\left(\sqrt{\frac{r_{ij}^2}{2\sigma_j^2}}\right) - \sqrt{\frac{4}{\pi}}\sqrt{\frac{r_{ij}^2}{2\sigma_j^2}} \exp\left(-\frac{r_{ij}^2}{2\sigma_j^2}\right), \tag{2}$$

called the cutoff function. Also, r is the distance between the interacting particles, and σ is the standard deviation of the Gaussian function. The vorticity equation is solved in a fractional step manner by calculating the stretching term,

$$\frac{D\boldsymbol{\alpha}_i}{Dt} = \sum_{j=1}^{n} \boldsymbol{\alpha}_j \nabla(g_\sigma \times \nabla G) \cdot \boldsymbol{\alpha}_i, \tag{3}$$

and the diffusion term,

$$\sigma^2 = 2\nu t \tag{4}$$

separately. We perform a radial basis function interpolation for reinitialized particle distributions every 100 steps to ensure the convergence of the diffusion calculation. The radial basis function interpolation is performed by ***PetRBF*** — A parallel $\mathcal{O}(N)$ algorithm for radial basis function interpolation with Gaussians[18]. This open-source library makes use of the GMRES solver with a restricted additive Schwarz method (RASM) preconditioner in PETSc (Portable Extensible Toolkit for Scientific Computing) [19].

The second-order Adams-Bashforth method is used for all time integration calculations. Equations (1) and (3) involve far-field interactions and can be solved using hierarchical N-body algorithms, such as the treecode and FMM. We execute these N-body calculations on the GPU, while all other components are executed on the CPU.

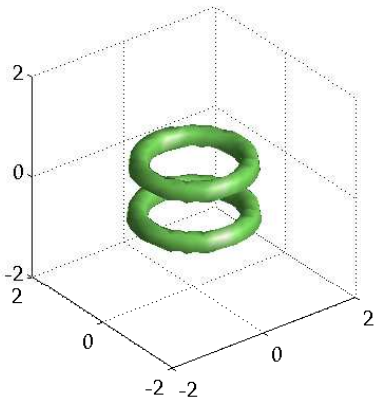
2.2 Treecode and FMM on GPUs

In order to execute the treecode and FMM efficiently on this architecture, the following modifications were made. First, the complex spherical harmonics were transformed to real basis functions, in order to avoid complex arithmetic on the GPU. Second, in order to minimize the memory usage per interaction, all of the translation matrices were generated on-the-fly. Third, the box structure and interaction lists are restructured and renumbered to match the number of threads-per-block on the GPU, so that no threads remain idle. Details of the implementation of the treecode and FMM on GPUs can be found in [20].

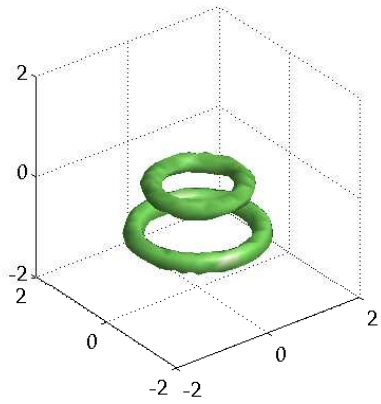
3 NUMERICAL EXPERIMENTS

3.1 Leapfrogging vortex rings

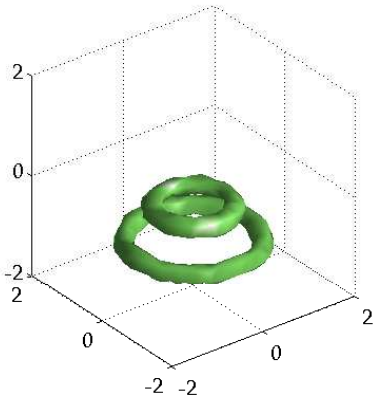
Two vortex rings with radius 1 are placed at a distance of 1 as shown in Figure 1(a). The core radius of the ring is 0.1, and the initial distribution of particles is set to cover the entire vorticity support. The inter-particle spacing was 0.02,



(a) t=0



(b) t=1



(c) t=2

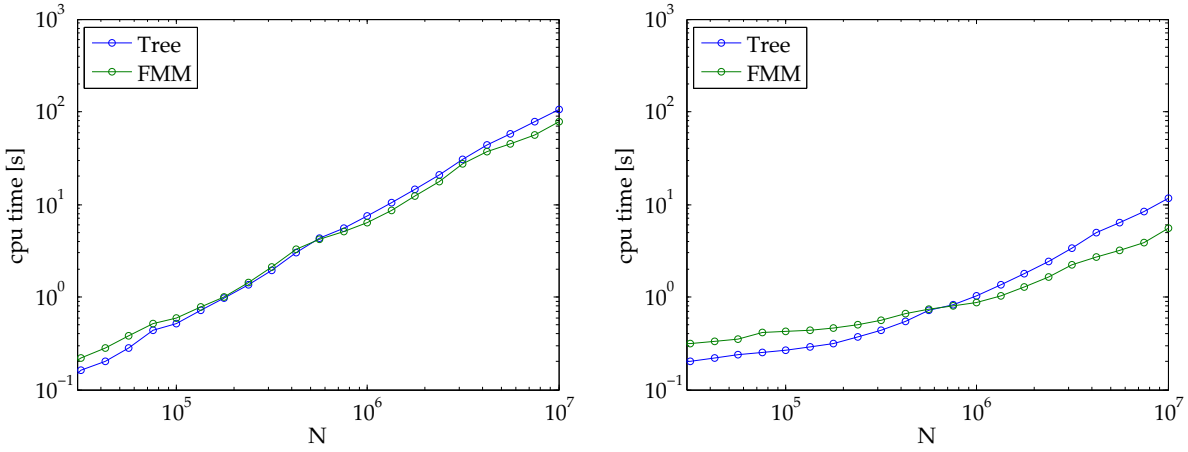
Figure 1. Isosurface of vorticity for different times

which results in a total number of about $1,000,000$ vortex particles. The time step size was 0.005 and the Reynolds number based on the circulation of the ring was $Re_\Gamma = \Gamma/\nu = 2,000$. The isosurface of the vorticity is shown for different time steps in Figure 1. It can be seen from the three sequence of figures that the front ring expands and decelerates while the rear ring shrinks and accelerates.

3.2 Treecode vs. FMM

The test problem of the leapfrogging vortex rings was computed using both the treecode and the FMM algorithms for the N-body interaction of the particles required by the velocity evaluation. The calculation time for the velocity evaluation using the Biot-Savart formula for one time step using different numbers of particles N is shown in Figure 2(a). The computational complexity of both algorithms implies that calculation time for the treecode should scale as $\mathcal{O}(N \log N)$ while for the FMM it should scale as $\mathcal{O}(N)$. For the present calculations, the order of multipole expansions was chosen as $p = 10$. The calculations were performed on a single GPU (NVIDIA GTX 295).

The calculation time of the same calculation on 32 GPUs is shown in Figure 2(b). The treecode for multiple GPUs requires further tuning and does not give expected results at the moment. Note that this is not a limitation of the algorithm itself.



(a) Single GPU (b) 32 GPUs

Figure 2. Calculation time of treecode and FMM on a single GPU

4 CONCLUSIONS AND OUTLOOK

The treecode and fast multipole method (FMM) are implemented on multi-GPUs for the vortex method simulation of two leapfrogging vortex rings. The $\mathcal{O}(N \log N)$ scaling of the treecode causes it to crossover with the $\mathcal{O}(N)$ FMM. The crossover point depends on the type of hardware, distribution of particles, and the required accuracy. For the present case it seems to be in the order of $N = 10^5$. The final manuscript will include longer simulation results which will allow us to examine high Reynolds number leapfrogging. Also, a more thorough investigation of the comparative performance of treecodes and FMM on GPUs will be performed after further tuning of the codes.

REFERENCES

[1] T. Maxworthy, J. Fluid Mech. 51 (1972) 15.
[2] Y. Oshima, T. Kambe, A. Asaka, J. Phys. Soc. Jpn. 38 (1975) 1159.
[3] H. Yamada, T. Matsui, Phys. Fluids 21 (1978) 292.
[4] T. T. Lim, Phys. Fluids 9 (1997) 239.
[5] K. Shariff, A. Leonard, J. H. Ferziger, Phys. Fluids 18 (2006) 047104.
[6] M. J. Stock, W. J. A. Dahm, G. Tryggvason, J. Comput. Phys. 227 (2008) 9021.
[7] F. Schlegel, D. Wee, A. F. Ghoniem, J. Comput. Phys. 227 (2008) 9063.
[8] R. Cocle, G. Winckelmans, G. Daeninck, J. Comput. Phys. 227 (2008) 2263.
[9] J. E. Barnes, P. Hut, Nature 324 (1986) 446.
[10] L. Greengard, V. Rokhlin, J. Comput. Phys. 73 (1987) 325.
[11] R. Hockney, J. Eastwood, Computer Simulation using Particles, Institute of Physics Publishing (1988)
[12] R. Yokota, T. K. Sheel, S. Obi, J. Comput. Phys. 226 (2007) 1589.
[13] I. F. Sbalzarini, J. H. Walther, M. Bergdorf, S. E. Hieber, E. M. Kotsalis, P. Koumoutsakos, J. Comput. Mech. 215 (2006) 566.
[14] P. Chatelain, A. Curioni, M. Bergdorf, D. Rossinelli,W. Andreoni, P. Koumoutsakos, Comput. Meth. Appl. Mech. Eng. 197 (2008) 1296.
[15] M. J. Stock, A. Gharakhani, in: Proc. of 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada (2008).
[16] R. Yokota, T. Narumi, R. Sakamaki, S. Kameoka, S. Obi, K. Yasuoka, Comput. Phys. Comm. 180 (2009) 2066.
[17] D. Rossinelli, M. Bergdorf, G-H. Cottet, P. Koumoutsakos, J. Comput. Phys. (2010) doi:10.1016/j.jcp.2010.01.004
[18] R. Yokota, M. G. Knepley, L. A. Barba, Comput. Meth. Appl. Mech. Eng., accepted (2010)
[19] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. Knepley, L. Curfman-McInnes, B. F. Smith, H. Zhang. PETSc User's Manual. - Revision 3.0.0, Argonne National Laboratory (2008)
[20] T. Hamada, R. Yokota, K. Nitadori, T. Narumi, K. Yasuoka, M. Taiji, K. Oguri, ACM Gordon Bell prize price/perfomance, SC09 (2009)