Computers & Fluids 92 (2014) 274–280

Contents lists available at ScienceDirect

Computers & Fluids

journal homepage: www.elsevier.com/locate/compfluid

The 3D vortex particle method in parallel computations on many GPUs

Andrzej Kosior *, Henryk Kudela

Wroclaw University of Technology, Wybrzeze Wyspianskiego 27, 50-537 Wroclaw, Poland

ARTICLE INFO

Article history:
Received 6 May 2013
Received in revised form 18 September 2013
Accepted 9 October 2013
Available online 20 October 2013

Keywords:
Leapfrogging
Head-on collision
Vortex in cell
Parallel computations
Graphics cards

ABSTRACT

Parallel implementation of the Vortex-in-Cell (VIC) method for 3D flow on many graphics cards was presented. As test problems it was chosen the leapfrogging and head-on collision of two vortex rings for which a well documented visualization exists in the literature. Our aim was to show the great potential of the VIC method for solution of 3D flow problems and that it is very well suited for parallel computation.

1. Introduction

Numerical solution of the 3D Navier – Stokes equations for high Reynolds number, using any method, is a very time consuming process. Recently there is not much increase in the computational power of a single processor. Instead we are forced to use multicore architectures and parallel computation. But to take advantage of their potential one needs to use proper numerical algorithms. One of the multiprocessor hardware that can be efficiently used in scientific computations is the graphics processing unit (GPU). It is built of hundreds of simple streaming processors which altogether give a great computational power. GPUs are relatively cheap and commonly available.

Our first implementation used only one graphics card for computation. We were able to use the computational grid of $128 \times 128 \times 128$ nodes (on a newer GPU we were able to fit computational grid of $224 \times 224 \times 224$ nodes). Our results can be found in [11]. Very quickly we found the limitation of the RAM memory of a single GPU. We were forced to use many graphics cards. For communication we used MPI library.

As a numerical method we chose the 3D Vortex-in-Cell (VIC) method that become more and more important method in numerical investigation of the fluid dynamics phenomena [4–6,20,18]. In this method particles carry information about vorticity. It is well known that the velocity may be calculated from the vorticity distribution. Next the vortex particles are displaced according to local velocity field. Particles intensity is then interpolated back to the grid nodes. To simulate the effect of the viscosity the viscous splitting was used and the diffusion equation was solved in each time step.

The VIC method is very well suited for parallel computation [11,10,19,7]. The displacement and redistribution processes, which have to be done at each time step, have a local character and the computations for each particle can be done independently. So the whole set of particles can be divided into independent groups and operations over these groups can be done concurrently.

In the paper it was presented the numerical results of the interaction between two vortex rings. The phenomena of leapfrogging and head-on collision of vortex rings were simulated. Experimental results for both cases are well documented in the literature [12,13].

The structure of the article is as follows: in the next section a short description of the VIC method is given, in Section 3 it was presented the numerical test cases and its results and the last section are closing remarks.

2. Equations of the motion and description of the vortex particle method

Equations of incompressible and viscous fluid motion have the following form:

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

$$\nabla \cdot \mathbf{u} = 0 \tag{2}$$

where $\mathbf{u} = (u, v, w)$ is velocity vector, ρ is fluid density, p is pressure, ν is kinematic viscosity. Eq. (1) can be transformed to the Helmholtz equation for vorticity evolution [21]:

* Corresponding author. Tel.: +48 713203553; fax: 48 713417708.
E-mail addresses: andrzej.kosior@pwr.wroc.pl (A. Kosior), henryk.kudela@pwr.wroc.pl (H. Kudela).

0045-7930/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compfluid.2013.10.011

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + (\mathbf{u}\cdot\nabla)\boldsymbol{\omega} = (\boldsymbol{\omega}\cdot\nabla)\mathbf{u} + \nu\Delta\boldsymbol{\omega} \tag{3}$$

where $\boldsymbol{\omega} = \nabla \times \mathbf{u}$.

Generally in all vortex particle methods the viscous splitting algorithm is used [8]. The solution is obtained in two steps: first, the inviscid - Euler equation is solved.

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + (\mathbf{u}\cdot\nabla)\boldsymbol{\omega} = (\boldsymbol{\omega}\cdot\nabla)\mathbf{u} \tag{4}$$

Next, the viscosity effect is simulated by solving the diffusion equation

$$\frac{\partial \boldsymbol{\omega}}{\partial t} = \nu\Delta\boldsymbol{\omega} \tag{5}$$

$$\boldsymbol{\omega}(\mathbf{x},0) = \boldsymbol{\omega}_I \tag{6}$$

where $\boldsymbol{\omega}_I$ is vorticity distribution obtained after inviscid step.

For the solution of Eqs. (5) and (6) one can use any suitable method like the Particle Strength Exchange (PSE) method [4] or the Finite Difference method (FDM). Due to the fact that we did the distribution of the particles intensities in each time step to the grid nodes and the positions of the particles are fixed, the most suitable method for that case, seems to be the FDM method with semi-implicit Crank–Nicholson scheme that is of the order $(\mathcal{O}((\Delta t)^2 + (\Delta x_i)^2))$.

The mth component ($m = 1,2,3$) of the vorticity vector was approximated by

$$\frac{\omega_m^{n+1} - \omega_m^n}{\Delta t} = \frac{1}{2}L^{n+1}\omega_m + \frac{1}{2}L^n\omega_m \tag{7}$$

where

$$L^{n+1}\omega_m = \nu\left(\Lambda_i\omega_m^{n+1} + \Lambda_j\omega_m^{n+1} + \Lambda_k\omega_m^{n+1}\right) \tag{8}$$

and

$$\Lambda_i\omega = \frac{\omega_{i+1,j,k} - 2\omega_{i,j,k} + \omega_{i-1,j,k}}{\Delta x_i^2}, \tag{9}$$

where index n related to time level $t_n = n\Delta t$ and (i,j,k) enumerates the grid nodes in x_1, x_2, x_3 directions respectively. The resulting algebraic systems were solved on GPU by the conjugate-gradient method.

In inviscid flow (4), according to the third Helmholtz theorem [21], vorticity lines move as the material fluid particles. Thus the movement of the vortex particles can be described by the infinite set of ordinary differential equations:

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{u}(\mathbf{x}_p, t), \qquad \mathbf{x}(0,\boldsymbol{\alpha}) = \boldsymbol{\alpha} \tag{10}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$ means the Lagrange coordinates of fluid particles. Solution of Eq. (10) gives the particle-trajectory mapping $\Phi(\cdot,t) : \mathbb{R}^3 \to \mathbb{R}^3; \alpha \to \Phi(\alpha,t) = \mathbf{x} \in \mathbb{R}^3$ that is one-to-one and onto. Incompressibility implies that $det(\nabla_\alpha \boldsymbol{\Phi}(\boldsymbol{\alpha},t)) = 1$

The velocity can be expressed through vorticity distribution using the Biot–Savart law

$$\mathbf{u}(\mathbf{x},t) = \int \mathbf{K}(\mathbf{x}-\mathbf{x}')\boldsymbol{\omega}(\mathbf{x}',t)d\mathbf{x}' = (\mathbf{K}^*\boldsymbol{\omega})(\mathbf{x},t) \tag{11}$$

where * denotes convolution and $\mathbf{K}$ is a 3×3 matrix kernel

$$\mathbf{K}(\mathbf{x}) = \frac{1}{4\pi}\frac{1}{|\mathbf{x}|^3}\begin{pmatrix} 0 & x_3 & -x_2 \\ -x_3 & 0 & x_1 \\ x_2 & -x_1 & 0 \end{pmatrix}, \qquad |\mathbf{x}| = \sqrt{x_1^2 + x_2^2 + x_3^2}$$

Eq. (11) are the fundamental formulas for direct vorticity methods [14]. By covering the domain of the flow with a numerical mesh ($N_x \times N_y \times N_z$) with equidistant spacing h, the ith component of the intensity vector particle $\boldsymbol{\alpha}_i$ is defined by the expression:

$$\alpha_i = \int_{V_p} \omega_i(x_1,x_2,x_3)d\mathbf{x} \approx h^3\omega_i(\mathbf{x}_p), \quad \mathbf{x}_p \in V_p, \quad |V_p| = h^3 \tag{12}$$

where V_p is the volume of the cell with index p. The velocity field is found by summing over all of the particles

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}(\mathbf{x}_i,t) = \sum_{p=1}^{N_p}\mathbf{K}(\mathbf{x}_i - \mathbf{x}_p)\boldsymbol{\alpha}_p(t) \tag{13}$$

$$\frac{d\boldsymbol{\alpha}_i}{dt} = (\boldsymbol{\alpha}_i(t)\cdot\nabla)\mathbf{u}(\mathbf{x}_i,t) \tag{14}$$

The disadvantage of the direct vortex particle method given by formula (13) is that the method is very computationally time consuming. It stems from the fact that one should take into account all of the mutual interactions between particles that are in the flow. In practical calculation one should also introduce regularization in the formulas (13) removing the singularity of the kernel $\mathbf{K}$ [17,15,14]. To overcome this difficulty we replaced the Biot–Savart law for velocity calculation with the grid method.

From incompressibility (2) stems the existence of the vector potential $\mathbf{A}$, such that:

$$\mathbf{u} = \nabla\times\mathbf{A} \tag{15}$$

The vector potential $\mathbf{A}$ defines the velocity field with accuracy to some potential field $\nabla\phi$. Adding $\mathbf{A} + \nabla\phi$ we do not change the left side of (15). It is convenient to assume that div$\mathbf{A} = 0$. The vorticity is related to $\mathbf{A}$ as follows

$$\nabla\times\nabla\times\mathbf{A} = \omega = \nabla(\nabla\cdot\mathbf{A}) - \Delta\mathbf{A} \tag{16}$$

So assuming that $\nabla\cdot\mathbf{A} = 0$ the vector potential can be obtained by solving three Poisson equations

$$\Delta A_i = -\omega_i, \quad i = 1,2,3 \tag{17}$$

with periodic boundary conditions in each direction.

The Poisson equation can be solved effectively using the numerical grid and finite difference method. Grid solution (17) permits for calculation of the velocity by formula (15).

Subsequently the velocity from the mesh nodes is interpolated onto the particles positions. Such an approach significantly accelerates (~1000 times quicker [6]) the calculations. Algebraic systems obtained from discretisation of the Poisson Eq. (17) were solved by the Multigrid method. The system of Eq. (13) was solved by the Runge–Kutta method of the 4th order. After solution of the Eq. (13) the intensity of the particles was redistributed onto the grid nodes.

We did the redistribution of the intensities of particles onto grid nodes in each time step, before the solution of the Poisson Eq. (17).

It was done using an interpolation:

$$\omega_j = \sum_p \tilde{\alpha}_{p_n}\varphi\left(\frac{\mathbf{x_j} - \tilde{\mathbf{x}}_p}{h}\right)h^{-3} \tag{18}$$

where j is the index of the numerical mesh node, p is the index of a particle.

Let us assume that $x \in \mathbb{R}$. In this work, we used the following interpolation kernel [4]

$$\varphi(x) = \begin{cases} (2 - 5x^2 + 3|x|^3)/2 & \text{if } 0 \leqslant |x| \leqslant 1 \\ (2-|x|)^2(1-|x|)/2 & \text{if } 1 \leqslant |x| \leqslant 2 \\ 0 & \text{if } 2 \leqslant |x| \end{cases} \tag{19}$$

For the 3D case, $\varphi = \varphi(x)\varphi(y)\varphi(z)$. The φ satisfies the following moment condition [4]: