 CMES, vol.88, no.1, pp.17-28, 2012

Particle-based Fluid Flow Simulations on GPGPU Using CUDA

Kazuhiko Kakuda[1], **Tsuyoki Nagashima**[1], **Yuki Hayashi**[1]
Shunsuke Obara[1], **Jun Toyotani**[1], **Nobuya Katsurada**[2]
Shunji Higuchi[2] **and Shohei Matsuda**[2]

Abstract: An acceleration of the particle-based incompressible fluid flow simulations on GPU using CUDA is presented. The particle method is based on the MPS (Moving Particle Semi-implicit) scheme using logarithmic-type weighting function to stabilize the spurious oscillatory solutions for the pressure fields which are governed by Poisson equation. The standard MPS scheme is widely utilized as a particle strategy for the free surface flow, the problem of moving boundary, multi-physics/multi-scale ones, and so forth. Numerical results demonstrate the workability and the validity of the present approach through dam-breaking flow problem.

Keywords: particle method, MPS, logarithmic weighting function, dam-breaking flow problem, GPGPU, CUDA.

1 Introduction

From a simulation-based practical point of view, it is important to compute massively multi-physics /multi-scale problem and moving boundary/obstacle one in the wide fields of engineering and science. It is not easy to simulate such problems by using the finite grid/element-based scheme. There are various gridless/meshless-based particle methods, such as SPH (Smoothed Particle Hydrodynamics) method [Lucy (1977); Gingold and Monaghan (1977); Monaghan (1994); Sakai, Yang and Jung (2004)], MPS (Moving Particle Semi-implicit) one [Koshizuka and Oka (1996); Kondo and Koshizuka (2011)], MLPG (Meshless Local Petrov-Galerkin) one [Atluri and Zhu (1998); Lin and Atluri (2000); Lin and Atluri (2001)], and so forth to simulate effectively such complicated problems.

[1] College of Industrial Technology, Nihon University, Narashino, Chiba 275-8575, Japan
[2] Product Engineering Department, Tsurumi Manufacturing Co., LTD., Tsurumi Tsurumi-ku, Osaka 538-8585, Japan

Recently, the physics-based computer simulations on the GPGPU (General-Purpose Computing on Graphics Processing Units) [nVIDIA (2012)] have increasingly become an important strategy for solving efficiently various problems, such as fluid dynamics [Harris (2004); Höfler (2006); Crane, Llamas and Tariq (2008); Harada, Masaie, Koshizuka and Kawaguchi (2008); Ogawa and Aoki (2009); Rossinelli, Bergdorf, Cotter and Koumoutsakos (2010); Hori, Gotoh, Ikari and Khayyer (2011)], rigid body dynamics [Harada (2008)], and so forth. Harris[Harris (2004)] has presented effectively 2D fluid flow simulation on the GPGPU, and the GPU implementations for 3D fluids have been demonstrated by Crane et al. [Crane, Llamas and Tariq (2008)] using CUDA (Compute Unified Device Architecture). Harada [Harada (2008)] has proposed the implementation of rigid body dynamics on the GPU, and the approach was also applied to particle-based fluid simulations. The accelerating DEM (Distinct Element Method) simulations on the GPU have been performed by using a sliced grid and a block transition sort proposed by Harada et al. [Harada, Masaie, Koshizuka and Kawaguchi (2008)]. They obtained about 3 times speed-up for the neighboring particle search and about 1.5 times speed-up for the overall computation. Hori et al. [Hori, Gotoh, Ikari and Khayyer (2011)] have presented a GPU-accelerated MPS code using CUDA and achieved about $3 \sim 7$ times speed-up.

The purpose of this paper is to present an acceleration of the particle-based incompressible fluid flow simulations on the GPU using CUDA. The standard/original MPS method leads to specially the unphysical numerical oscillation of pressure fields which are described by the discretized Poisson equation. To improve some shortcomings of the standard MPS method, Khayyer and Gotoh have proposed the modified MPS method for the prediction of wave impact pressure on a coastal structure to ensure more exact momentum conservation [Khayyer and Gotoh (2009)]. More recently, the improvement of stability in the original MPS method [Koshizuka and Oka (1996)] has been achieved by adding some source terms into Poisson pressure equation [Kondo and Koshizuka (2011)]. To overcome such spurious oscillations in the standard MPS method, we have proposed to utilize the logarithmic weighting function and also take into consideration the reduction of *ad hoc* influence radius for solving an auxiliary Poisson equation for the pressure field [Kakuda, Obara, Toyotani, Meguro and Furuichi (2012)]. The workability and validity of the present approach are demonstrated through dam-breaking flow [Martin and Moyce (1952); Hirt and Nichols (1981); Ramaswamy and Kawahara (1987)], and compared through the accelerating performance of GPU to the CPU time.

Throughout this paper, the summation convention on repeated indices is employed. A comma following a variable is used to denote partial differentiation with respect to the spatial variable.

2 Statement of the problem

Let Ω be a bounded domain in Euclidean space with a piecewise smooth boundary Γ. The unit outward normal vector to Γ is denoted by $\boldsymbol{n}$. Also, $\Im$ denotes a closed time interval.

The motion of an incompressible viscous fluid flow is governed by the following Navier-Stokes equations :

$$\frac{Du_i}{Dt} = -\frac{1}{\rho}p_{,i} + \nu u_{i,jj} + f_i \qquad in\ \Im \times \Omega \tag{1}$$

$$\frac{D\rho}{Dt} = 0 \qquad in\ \Im \times \Omega \tag{2}$$

where u_i is the velocity vector component, ρ is the density, p is the pressure, f_i is the external force, ν is the kinematic viscosity, and D/Dt denotes the Lagrangian differentiation.

In addition to Eq. 1 and Eq. 2, we prescribe the initial condition $u_i(\boldsymbol{x},0) = u_i^0$, where u_i^0 denotes the given initial velocity, and the Dirichlet and Neumann boundary conditions.

3 MPS formulation using a logarithmic-type weighting function

The particle interaction models of the MPS as illustrated in Fig. 1(a) are prepared with respect to differential operators, namely, gradient, divergence and Laplacian [Koshizuka and Oka (1996)]. The incompressible viscous fluid flow is calculated by a semi-implicit algorithm, such as SMAC (Simplified MAC) scheme [Amsden and Harlow (1970)]. For the standard MPS formulation, the selection of a weighting function is a key factor in the particle-based framework. If the distance r between the coordinates $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ is close, then there is a possibility that the computation fails suddenly with unphysical numerical oscillations. Therefore, in order to stabilize such spurious oscillations generated by the standard MPS strategy, we adopt the following logarithmic-type weighting function as shown in Fig. 1(b), and also consider the reduction of *ad hoc* influence radius, r_e, for solving the pressure fields (see Fig. 2).

$$w(r) = \begin{cases} log(\dfrac{r_e}{r}) & (r < r_e) \\ 0 & (r \geq r_e) \end{cases} \tag{3}$$

The logarithmic-type weighting function is generally similar to the profile of the weighting function proposed by Kondo and Koshizuka to stabilize the pressure calculations [Kondo and Koshizuka (2011)](see Fig. 1(b)).

The particle number density n at particle i with the neighboring particles j is defined as

$$< n >_i = \sum_{j \neq i} w(|\boldsymbol{r}_j - \boldsymbol{r}_i|) \tag{4}$$

The model of the gradient vectors at particle i between particles i and j is weighted with the kernel function and averaged as follows :

$$< \nabla \phi >_i = \frac{d}{n^0} \sum_{j \neq i} [\frac{\phi_j - \phi_i}{|\boldsymbol{r}_j - \boldsymbol{r}_i|^2} (\boldsymbol{r}_j - \boldsymbol{r}_i) w(|\boldsymbol{r}_j - \boldsymbol{r}_i|)] \tag{5}$$

where d is the number of spatial dimensions, ϕ_i and ϕ_j denote the scalar quantities at coordinates $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$, respectively, and n^0 is the constant value of the particle number density.

The Laplacian model at particle i is also given by

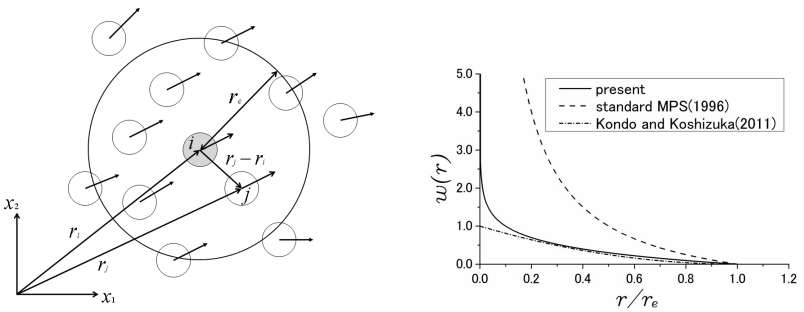
$$< \nabla^2 \phi >_i = \frac{2d}{n^0 \lambda} \sum_{j \neq i} (\phi_j - \phi_i) w(|\boldsymbol{r}_j - \boldsymbol{r}_i|) \tag{6}$$

where λ is an *ad hoc* coefficient.

The Poisson equation for solving implicitly the pressure field at particle i is given as follows:

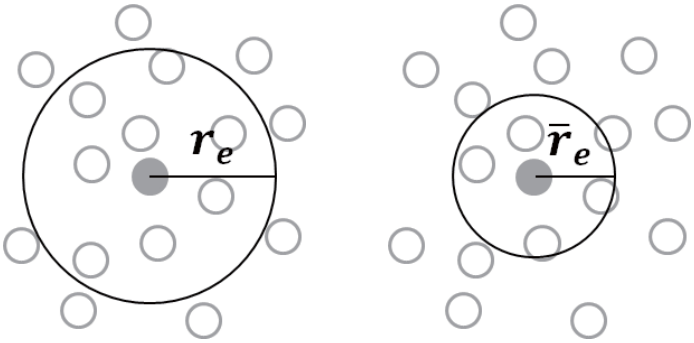
$$< \nabla^2 p >_i = -\frac{\rho}{\Delta t^2} \frac{< n^* >_i - n^0}{n^0} \tag{7}$$

where $< n^* >_i$ denotes the particle number at particle i.



(a) Particle interaction models (b) Profiles of weighting functions

Figure 1: Particle interaction models and weighting functions



(a) Velocity field (b) Pressure field

Figure 2: Basic idea for taking the influence radius r_e

4 GPU implementation using CUDA

The specification of CPU and GPU using CUDA is summarized in Tab. 1. Fig. 3 shows also the architecture of *NVIDIA GeForce GTX*580 as the graphics hardware supporting CUDA [nVIDIA (2012)].

A physical value at particle position is calculated as a weighted sum of the values of neighboring particles in the influence area. Therefore, we have to search for neighboring particles. The difficulty in implementing MPS on the GPU is that the neighborhood relationship among particles dynamically changes during the simulation.

The flow chart based on the MPS algorithm is illustrated in Fig. 4. The GPU implementation consists mainly of the following five steps, which are shown in the flow chart in Fig. 4:
1) search for neighboring particles in the influence area;
2) calculation of the particle number density;
3) solving the Poisson equation of the pressure fields by using Jacobi method;
4) calculation of the pressure gradient;
5) modification of velocities and positions of the particles.

5 Numerical example

In this section we present numerical results obtained from applications of the above-mentioned numerical method to dam-breaking incompressible flow problem involving free surface and gravity. The dam-breaking flow problem has been used widely to verify the applicability and validity of the numerical methods [Hirt and

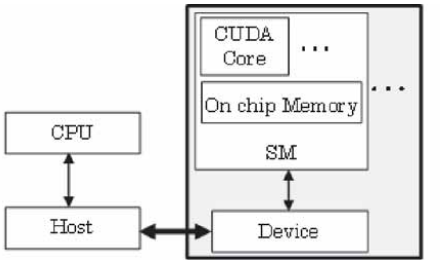


Figure 3: Architecture of NVIDIA's GPU

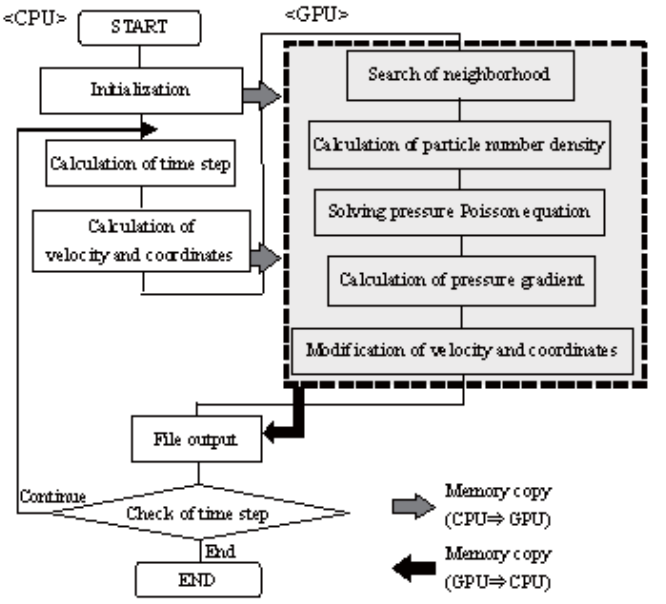


Figure 4: Flow chart based on the MPS algorithm

Table 1: A summary of the specification of CPU and GPU

CPU	Intel Core i7, 3.50GHz
Memory	DDR3 PC3-10600 16GB
OS	Cent OS 6.0 64bit
Bus	PCI Express 2.0x16
GPU	NVIDIA GeForce GTX580
Global Memory	1.5GB
Processor Clock	1544MHz
Streaming Multiprocessor (SM)	16
CUDA core	512
Memory Transfer Rate	192.4GB/s
Memory Interface	384bit
CUDA Driver	Version 4.10
Tool kit & SDK	Version 4.0

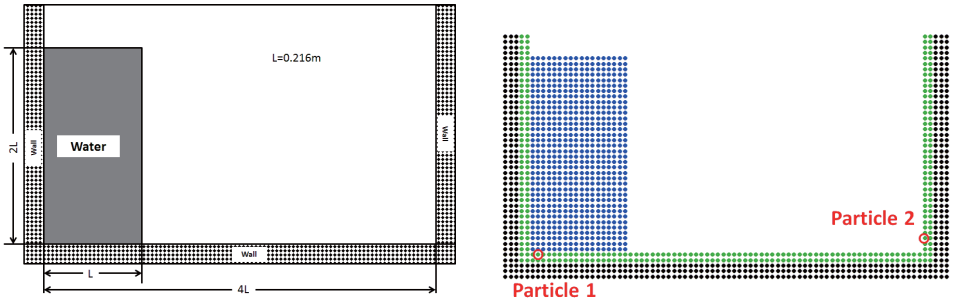
Nichols (1981); Ramaswamy and Kawahara (1987); Koshizuka and Oka (1996); Khayyer and Gotoh (2009); Kondo and Koshizuka (2011)]. The initial velocities are assumed to be zero everywhere in the interior domain.

Fig. 5 shows the geometry and the initial state of particles for flow in the dam-breaking problem. In two-dimensional simulation, we set several types of particle number in the initial configuration, and the CFL condition $u_{max}\Delta t/l_{min} \leq C$, where C is the Courant number $(= 0.1)$. The kernel sizes for the particle number density and the gradient/Laplacian models are $r_e = 4.0l_0$ and $\bar{r}_e = 2.0l_0$ for velocity and pressure calculations, respectively, in which l_0 is the distance between two neighboring particles in the initial state. In this case, we set $l_0 = 0.008m$.

Fig. 6 shows the instantaneous particle behaviors with $16,221$ particles at different time on the GPU using CUDA. The comparisons of CPU and GPU particle-based simulations at each time are also shown in Fig. 7, and the agreement between both results appears very satisfactory. Tab. 2 gives the comparisons of the computational time on CPU and GPU simulations up to $t \approx 0.5$. Fig. 8 shows also the accelerating performance of GPU to the CPU time. We can see from Fig. 8 that the performance with about $120,000$ particles leads to approximately 12 times speed-up.

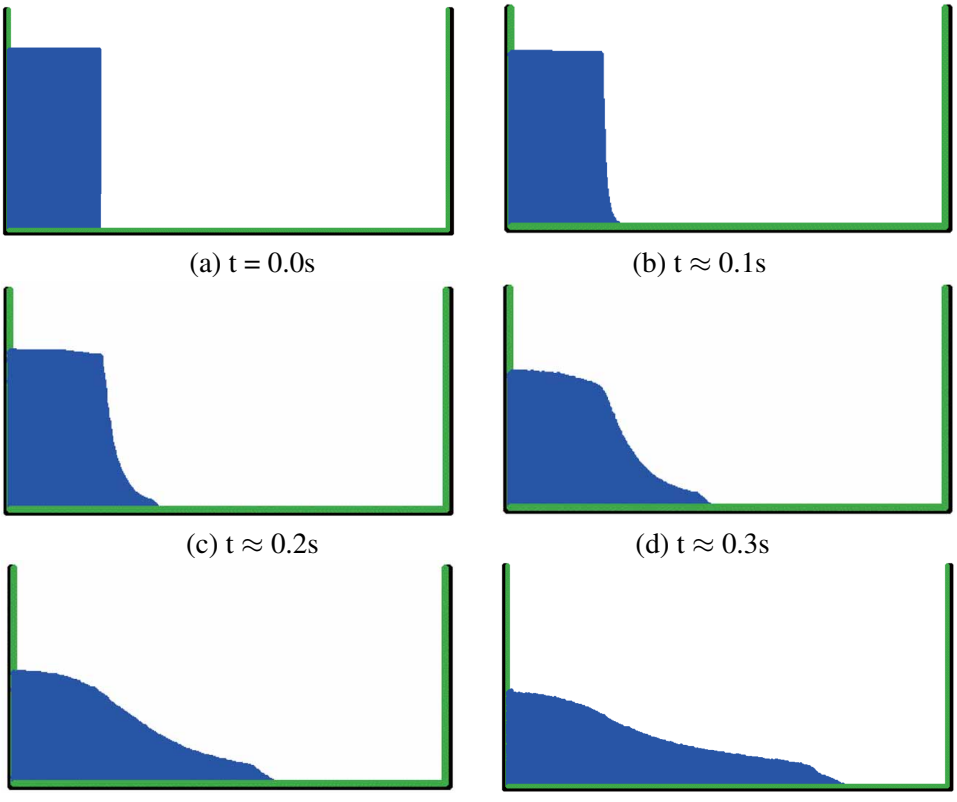
6 Conclusions

We have presented the GPU-based MPS simulation using logarithmic-type weighting function for solving numerically incompressible viscous fluid flow involving free surfaces and gravity. The standard MPS method has been widely utilized as a



(a) Geometry (b) Initial state of particles

Figure 5: Geometrical configuration and initial state of particles

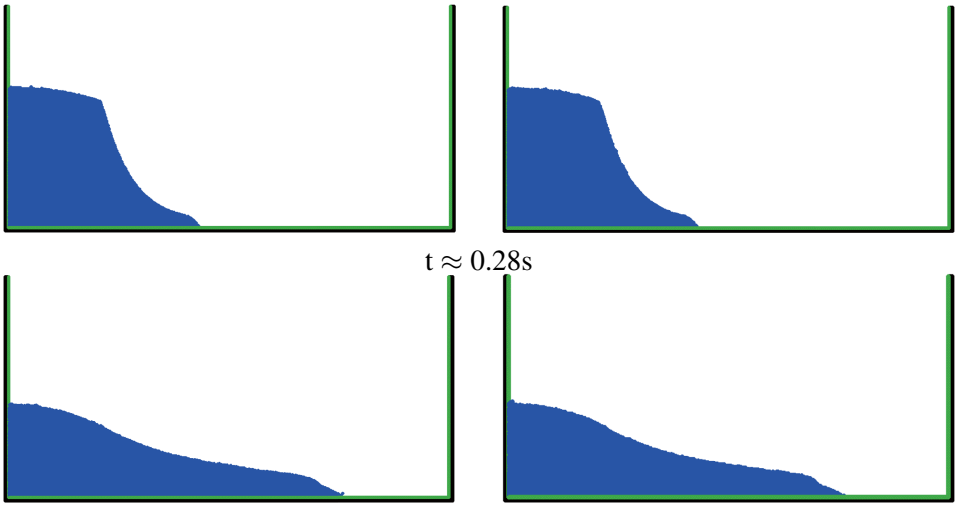


(a) t = 0.0s (b) t ≈ 0.1s

(c) t ≈ 0.2s (d) t ≈ 0.3s

(e) t ≈ 0.4s (f) t ≈ 0.5s

Figure 6: Computational results on GPU using CUDA



(a) CPU simulation (b) GPU simulation

Figure 7: Comparisons of CPU (left) and GPU (right) particle simulations

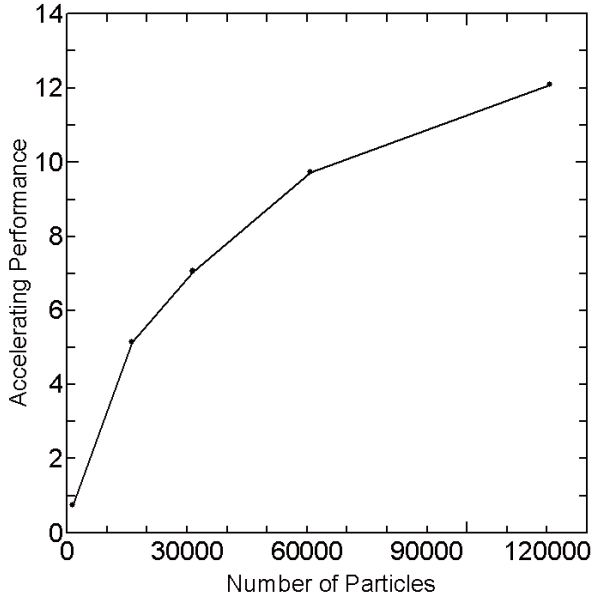


Figure 8: Accelerating performance of GPU to the CPU time

Table 2: Comparisons of the computational time on CPU and GPU [sec]

Number of particles	1,458	16,221	31,465	60,875	120,788
GPU 1st run	19.22	269.51	692.75	1801.92	5653.20
GPU 2nd run	17.25	267.12	698.04	1842.50	5538.14
GPU 3rd run	19.58	267.15	695.64	1788.60	5663.54
GPU 4th run	17.13	267.95	694.38	1781.98	5664.27
GPU 5th run	19.07	267.94	696.06	1782.27	5671.92
CPU 1st run	13.65	1384.57	4942.27	17486.74	68619.45
CPU 2nd run	13.76	1368.84	4905.48	17514.14	68035.42
CPU 3rd run	13.71	1377.11	4878.58	17494.31	68197.82
CPU 4th run	13.65	1382.20	4890.58	17629.67	68469.14
CPU 5th run	13.77	1367.53	4940.00	17439.36	67890.69
GPU Average	18.45	267.93	695.37	1799.45	5638.21
CPU Average	13.71	1376.05	4911.38	17512.84	68242.50
CPU-Ave./GPU-Ave.	0.7430	5.136	7.063	9.732	12.10

particle strategy for free surface flow, the problem of moving boundary, and multi-physics/multi-scale ones. To overcome spurious oscillations in the standard MPS method, we have utilized the logarithmic-type weighting function and also take into the influence radius reduction for solving an auxiliary Poisson equation of the pressure field. As the numerical example, the well-known dam-breaking flow problem has been carried out and compared with the CPU-based particle simulation. The results obtained herein are summarized as follows:

(1) The GPU implementation consists mainly of the five steps, namely, the search for neighboring particles in the influence area, the calculation of the particle number density, solving the Poisson equation of the pressure fields by using Jacobi method, the calculation of the pressure gradient, and the modification of velocities and positions of the particles.

(2) The qualitative agreement between CPU and GPU particle-based simulations appears very satisfactory.

(3) We can see that the performance on GPU with about $120,000$ particles leads to approximately 12 times speed-up.

(4) The numerical results demonstrate that the approach is capable of solving rapidly and in a stable manner the complicated flow phenomena involving free surfaces.

Acknowledgement: This work was supported by Grant-in-Aid for Scientific Research (C) (KAKENHI:No.23560075) in the MEXT.

References

Amsden, A. A.; Harlow, F. H. (1970): A simplified MAC technique for incompressible fluid flow calculations. *J. Comput. Phys.*, vol.6, pp.322–325.

Atluri, S. N.; Zhu, T. (1998): A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics. *Comput. Mech.*, vol.22, pp.117–127.

Crane, K.; Llamas, I.; Tariq, S. (2008): Real-time simulation and rendering of 3d fluids. *In GPU Gems 3*, (Ed. H. Nguyen), Addison-Wesley, pp.633–675.

Gingold, R. A.; Monaghan, J. J. (1977): Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Mon. Not. R. astr. Soc.*, vol.181, pp.375–389.

Harada, T. (2008): Real-time rigid body simulation on GPUs. *In GPU Gems 3*, (Ed. H. Nguyen), Addison-Wesley, pp.611–632.

Harada, T.; Masaie, I.; Koshizuka, S; Kawaguchi, Y. (2008): Accelerating particle-based simulations utilizing spatial locality on the GPU. *Trans. JSCES*, (in Japanese), Paper No.20080016.

Harris, M. J. (2004): Fast fluid dynamics simulation on the GPU. *In GPU Gems*, (Ed. R. Fernando), Addison-Wesley, pp.637–665.

Hirt, C. W.; Nichols, B. D. (1981): Volume of fluid (VOF) method for the dynamics of free boundaries. *J. Comput. Phys.*, vol.39, pp.201–225.

Höfler, M. (2006): Real-time visualization of unstructured volumetric CFD data sets on GPUs. *CESCG*.

Hori, C.; Gotoh, H.; Ikari, H; Khayyer, A. (2011): GPU-acceleration for moving particle semi-implicit method. *Computers & Fluids*, vol.51, pp.174–183.

Kakuda, K.; Obara, S.; Toyotani, J.; Meguro, M.; Furuichi, M. (2012): Fluid flow simulation using particle method and its physics-based computer graphics. *CMES: Computer Modeling in Engineering and Sciences*, vol.83, no.1, pp.57–72.

Khayyer, A.; Gotoh, H. (2009): Modified moving particle semi-implicit methods for the prediction of 2d wave impact pressure. *Coastal Engineering*, vol.56, pp.419–440.

Kondo, M.; Koshizuka, S. (2011): Improvement of stability in moving particle semi-implicit method. *Int. J. Numer. Meth. Fluids*, vol.65, pp.638–654.

Koshizuka, S.; Oka, Y. (1996): Moving-particle semi-implicit method for fragmentation of incompressible fluid. *Nucl. Sci. Eng.*, vol.123, pp.421–434.

Lin, H.; Atluri, S. N. (2000): Meshless local Petrov-Galerkin method (MLPG) for convection-diffusion problems. *CMES: Computer Modeling in Engineering and Sciences*, vol.1, no.2, pp.45–60.

Lin, H.; Atluri, S. N. (2001): The meshless local Petrov-Galerkin (MLPG) method for solving incompressible Navier-Stokes equations. *CMES: Computer Modeling in Engineering and Sciences*, vol.2, no.2, pp.117–142.

Lucy, L. B. (1977): A numerical approach to the testing of the fission hypothesis. *The Astronomical J.*, vol.82, no.12, pp.1013–1024.

Martin, J. C.; Moyce, W. J. (1952): An experimental study of the collapse of liquid columns on a rigid horizontal plane. *Philos. Trans. R. Soc. London Ser. A*, vol.244, pp.312–334.

Monaghan, J. J. (1994): Simulating free surface flows with SPH. *J. Comput. Phys.*, vol.110, pp.399–406.

nVIDIA (2012): CUDA zone. *http://developer.nvidia.com/category/zone/cuda-zone*

Ogawa, S.; Aoki, T (2009): GPU computing for 2-dimensional incompressible-flow simulation based on multigrid method. *Trans. JSCES*, (in Japanese), Paper No.20090021.

Ramaswamy, B.; Kawahara, M. (1987): Lagrangian finite element analysis applied to viscous free surface fluid flow. *Int. J. Numer. Meth. Fluids*, vol.7, pp.953–984.

Rossinelli, D.; Bergdorf, M.; Cotter, G. H.; Koumoutsakos, P. (2010): GPU accelerated simulations of bluff body flows using vortex particle methods. *J. Comput. Phys.*, vol.229, pp.3316–3333.

Sakai, Y.; Yang, Z. Y.; Jung, Y. G. (2004): Incompressible viscous flow analysis by SPH. *Trans. Soc. Japan Mech. Engrs.* (in Japanease), vol.70,696 , pp.1949–1956.