Fluid Dynamic Simulation for Cutting in Virtual Environment

Sugeng Rianto
Medical Imaging and Applied Science
Curtin University of Technology
Perth, Western Australia 6102
Sugeng.rianto@postgrad.curtin.edu.au

Ling Li
Department of Computing
Curtin University of Technology
Perth, Western Australia 6102
L.Li@curtin.edu.au

Bruce Hartley
Department of Exploration Geophysics
Curtin University of Technology
Perth, Western Australia 6102
B.Hartley@curtin.edu.au

ABSTRACT

In this paper, we introduce a 3D fluid dynamics solver for real-time interactions in virtual environment. We approach the solution of differential equations based on the cubic interpolated propagation (CIP) technique on GPU. Since the CIP combine the solution for fluid equations and their interactions with the environment together, the Navier-Stokes equation can be solved efficiently. Furthermore, to achieve high performance results without involving a supercomputer, we take advantage of the parallelism and programmability of the GPU. Simulation is performed on pixels that can be considered to be a grid of cells; therefore processing on multiple vertices and pixels can be done simultaneously in parallel. This strategy is effective enough to simulate fluid dynamic model for real-time virtual cutting in 3D computer graphic. Experimental results demonstrate that the skin cutting followed by blood flowing over the anatomical surface run smoothly in a real-time interaction and realistic visual effect is achieved.

Keywords

CIP, cutting simulation, fluid dynamic solver, GPU, virtual environment

1. INTRODUCTION

Virtual reality (VR) is widely used as a training tool from science to engineering or social sciences [Ros00, Sto00]. The VR techniques provide the potential for a realistic, safe, controllable environment for novice doctors to practice surgical operations, allowing them to make mistakes without serious consequences [Ano01, Hig02]. Virtual reality is also widely known as virtual (or synthetic) environments (VEs). These are three dimensional computer generated environments that the user is not only able to experience interactively but also able to manipulate in real time [Was01]. The way humans interact with their physical environments are artificially imitated in VEs.

VEs provide natural interface between humans and computers in virtual reality. A VE consists of an interfacing system with humans through output of sensory information and input of commands. Several examples of such input and output devices are 3D space mouse, data gloves, 3D navigation devices, and haptic feedback.

The major goal of computer graphics and virtual environment simulation is to provide *realistic* methods to allow an easy creation of digital equivalents for natural phenomena such as fluid dynamic behavior and its interactions. The common problem in a fluid simulation and animation, on the other hand, is how to solve the underlying laws that govern fluids motions. The fluid dynamic simulation is known to be difficult to solve in an efficient way, sometimes even too difficult to be solved at all.

Furthermore, a real time rendering in immersive virtual environment with haptic interaction demand high frequency processing (1000 Hz) and the visual aspects need frame rates in the order of 30 fps.

This study introduces novel framework utilizing the immersive virtual environment for fluid interaction in cutting procedures and haptic rendering. The fluid dynamic computations rely on the Navier-Stokes equation in three-dimension (3D) is solved using cubic interpolated propagation based on GPU programming. The framework is expected to provide

efficient computation which is applicable for haptic rendering.

2. PREVIOUS WORK

Several methods have been developed to simulate fluids and its interaction to objects based on the Navier-Stokes equations to exhibit physically realistic fluid behaviours [Carl04, Che95]. In consideration to computation cost some studies attempted to solve the NS equation with simplifications [Fos01, Sta99].

The cubic interpolated propagation (CIP) methods was firstly introduced in [Yab91c] as a universal solver for hyperbolic equation by cubic interpolation. It is then used for solving fluid dynamic equation as a conservative semi-Lagrangian solver for solid, liquid and gas [Yab91b, Yab02]. Some studies prove that this method is efficient enough to solve dynamic fluid equation in all states with motions [Yab04, Yab02] .

Although the CIP has been accepted as an alternative method for fluid solver that fits the need of computer animations, there has been no study that uses this method in immersed virtual environments with a haptic interaction. More advance studies [Liu04, Li03] attempted to conduct fluid dynamic computation on the GPU to get parallelisms for more efficiency computations. The GPU applications for study fluid erosions [Anh07, Ben06, Mei07, Nei05] also become a current research interest in computer graphics and animation. These study show promising results to increase the interactivity visual cue; however, most of them are not involving kinesthetic interactions.

This paper introduce a framework which combines CIP and parallelism in GPU, trying to achieve most efficient solution of fluid dynamic equations for real time rendering in virtual environment with haptic interaction.

The rest of the paper is organized as follows: CIP as the fluid dynamic solvers is described in Section 3; computational fluid dynamics implemented in GPU is described in Section 4; Section 5 described the experimental set-ups and the results; with Section 6 concludes the paper.

3. FLUID DYNAMIC SOLVER

3.1. Cubic interpolated propagation

The Navier-Stokes equation (NSe) for incompressible flow can be described as a *momentum*

$$\frac{\partial u}{\partial t} = -(u \bullet \nabla)u - \frac{1}{\rho}\nabla p + \frac{\mu}{\rho}\nabla^2 u + F_c$$

and *mass conservation* function

$$\nabla \bullet u = 0 \quad (2)$$

where u is a velocity field of fluid, p is a pressure, ρ is the density of the fluid, μ is a viscosity coefficient, F_c represents external force (gravity, haptic motion), and ∇ is the differential operator, respectively.

The behaviors of a fluid can be predicted by solving equations (1) and (2). Then, a flow dynamics simulation can be done using staggered grids that define velocity components at cell faces and scalar variables in cell centers. The fraction of volume of fluid in each cell is transported by advection equation as formulated below.

$$\frac{\partial f}{\partial t} = -(u \bullet \nabla) f \quad (3)$$

where f is the function of volume of fluid fraction (VOF). By using the CIP method [Tak03, Yab02, Yok02] and the advection form ($-(u \bullet \nabla)u$), the equation (3) can be solved.

The interface between liquid and solid is traced by distributing Eqn. (4) into advection phase shown in Eqn. (5) and non-advection in Eqn. (6) as follows:

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = G \quad (4)$$

$$\frac{\partial f}{\partial t} = G \quad (5)$$

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0 \quad (6)$$

The CIP scheme calculates the advection phase by shifting a cubic interpolated profile into space according to the total derivative equations. Since the interpolation profile uses the cubic polynomial and the spatial derivative value at each grid point, it can be derived that:

$$\frac{\partial f'}{\partial t} + u\frac{\partial f'}{\partial x} = G' - f'\frac{\partial u}{\partial x} \quad (7)$$

$$\frac{\partial f'}{\partial t} = G' - f'\frac{\partial u}{\partial x} \quad (8)$$

$$\frac{\partial f'}{\partial t} + u\frac{\partial f'}{\partial x} = 0 \quad (9)$$

The exact solution of Equation (6) is

$$f(x,t) = f(x - ut, 0) \quad (10)$$

(1)

If the velocity is assumed to have a constant value within short time, we get

$$f(x,t+\Delta t) \cong f(x-u\Delta t,t) \qquad (11)$$

Since $x\text{-}u\Delta t$ is not always located on grid points, this point is interpolated locally by Hermit polynomials within the calculation grid point as the value for the next time step. Within the discretised function a cubic-Hermite polynomial can be expressed as follow:

$$F_j(x) = [a_j X - b_j)X + \Delta x f_j^{'}]X + f_j \qquad (12)$$

where

$$a_j = \Delta x(f_j^{'} + f_{j+1}^{'}) - 2(f_{j+1} - f_j)$$

$$b_j = \Delta x(f_j^{'} + 2f_{j+1}^{'}) - 3(f_{j+1} - f_j)$$

and

$$X = \frac{x - x_j}{\Delta x}$$

$$\Delta x = x_j - x_{j-1}$$

Furthermore, Equation (1) can also be used to approximate the other fluid states such as a fluid pressure. The pressure equation is solved using the Poisson's equation with assumption that the liquid has no viscosity and has the external force that is determined by the force released from/to the haptic interaction. After the VOF of each cell is obtained in the simulation, a fluid surface is created and it is smoothened by subdivision, if necessary.

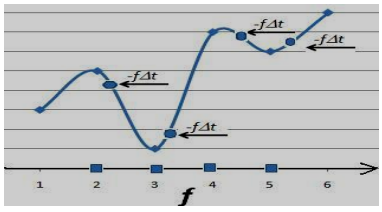


Figure 1. Estimation of the physical value on the departure point of the arbitrary contour in advection phase.

The CIP scheme uses a spatial profile that rely on different interpolations to propagate the solution along the characteristics [Uts91a]. The physical value on f_t is calculated by a Hermite interpolation constructed locally within the calculation cell. Then, an estimate value is directly advected toward the grid point calculation at the next time step value (figure 1).

By employing a cubic-Hermite polynomial within the discretised function (a grid cell) and its approximation in continuous manner of an explicit finite-difference form, and by assuming that both a physical quantity (f) and its spatial derivative (f') satisfy the master advection equations, the CIP method provides a stable result, less diffusive, and has third order accuracy [Uts97, Uts04, Yok02]. Furthermore, the numerical solvers do not need the solution for each interface, because the solver yields the solutions at once.

When using the CIP method as fluid solver, the highly accurate result in coarse grid can be obtained; however, the volume of fluid is not properly conserved; which may be caused by isosurface extraction during rendering and it is tolerable for a short simulation period.

3.2. Fluid interaction and simulation

Although CIP can combine the solution of fluid dynamics and the fluid-solid interaction together in one algorithm, the dynamic interaction between fluid and solid, and between a haptic and fluid, need to be studied in order to achieve realistic visual result. Such interactions can be described as collision between two objects. For haptic rendering, a collision detection and the impulse response are employed using the method introduced in [Gre00].

The interface between fluid and solid are simulated with a volume of solid (VOS) value. The VOS defines the fraction of a solid contained in a fluid cell and approximate the shape of rigid body in the fluid solver. The cells with VOS values less than or equal to 0.5 are considered as fluid otherwise a solid (figure 2).

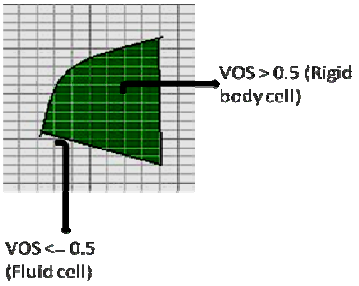


Figure 2. Approximation of fluid and solid cells.

The dynamics of rigid bodies are approximated based on force Fc acting on the centre of mass and the torque Tc from the centre of mass as follow:

$$F_C = Mg + \iint_s (p.n)ndS \qquad (13)$$

$$T_C = \iint_s (r - r_C)(p.n)dS \qquad (14)$$

where g is the gravity, M is mass of the object, p is the pressure of a boundary between a solid cell and a fluid cell, n is the normal vector to the boundary, r is

a point on the boundary, r_c is the object's centre of gravity, and s is the boundary between the fluid and the solid objects.

A spilling fluid from the cut is assumed as a droplet liquid that travel over the mesh. When the dropped fluid flows on a surface, some amount of fluid remains behind because of surface absorption and wetting. This remaining fluid will merges into the other fluid that spill later on.

3.3. Simulation and computation of the fluid dynamic on a GPU

Although, the real-time fluid computation system involving in this simulation can be categorised into the process that is done in a CPU (central processing unit) and in a GPU (graphic processor unit), the fluid simulation presented in the flowchart do not clearly show these divisions (figure 3). Beside controls the hardware (peripherals, haptic and 3D space navigator), the CPU computes the collision detections, detects or/and delivers the forces from/to a user, and maintains the haptic rendering rates. The GPU, on the other hand, responsible to render the image into a display and to compute a fluid dynamic numerical solver.

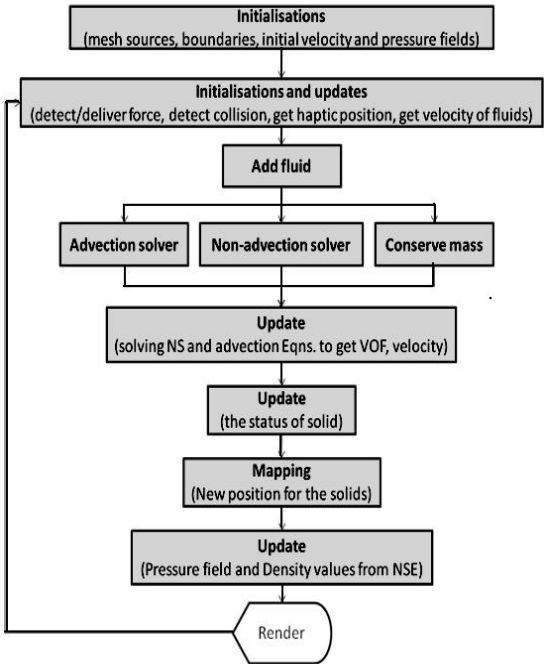


Figure 3. Flowchart of the real-time fluid simulation.

The fluid solver for the simulation that is developed here, is adapted from [Har04] with novelties on the numerical methods and the use of parallelism on the GPU for fluid dynamic computations. As explained above the CIP numerical method to solve the NSe is used. The solver is mainly divided into four fragment programs in the GPU: an advection, a non-advection, divergences and gradients, and boundary conditions.

Besides the fluid solver computations, the GPU is employed as a tree lookup generator in the simulation. The simulation is initialized by creating an octree around the mesh surface in the GPU. The algorithm for the texture mesh is developed based on the octree texture from the fragment program using the tree lookup as suggested in [Lev05]. Then, the density values are updated using a render-to-texture operation during the simulation and stored in 2D texture map. The octree's leafs store the index of a pixel in the form of the three 8-bit values (in RGB Channels). The simulation will also access the density of the neighbors to recode the neighboring information.

All the fragment programs are written in Cg [Fer03, Wil03]. The fragment codes for octree has the call *tex2D(Dmap, I)*, where *I* is the index stored within a leaf *L* of the octree and *Dmap* is the density map. This call returns the density value associated with leaf *L*. The call *tex2D(N ,I)* yields the index within the density map of a 3D space neighbor. To get all neighbor information, we require 26 textures in 3D or 9 textures in 2D. When all textures have been created, the density map can be render on the texture mesh and the location of each density can be retrieve from the octree. The texture density values are updated based on the fluid solver computation, the current VOF, the velocity, and the solid status respectively.

3.4. Force feedback design

Even though force feedback design is not a main discussion in this paper, this section describes briefly our force feedback design to show how to manage the haptic device. This section will also not explain in detail of each force involved in the discussion. This project employ two different threads: the first one is for visual rendering updated in 30Hz and the other for haptic rendering in the rate of about 1000Hz.

A force computation generating feedback into a haptic usually is called as a *force model*. This force has a means of mapping from the haptic device position to a force vector[Ano05]. The force model also deliver constrain or reaction forces that are sensed by the user through a device. Although a force model is updated in the scene-graph loop at about 60 Hz, the current force model will be used extensively by the real-time loop at about 1000 Hz by repeated calls to its evaluate function.

In this paper the goals of force models design are: to fell a skin surface, to create cutting path, and to

prompt bleeding. The formulation of forces can be expressed as is shown in equation (15) as follows:

$$F_t = kF_p - \sum F_{xy} + F_d \qquad (15)$$

where F_t is total forces felled by a user, F_p is pressure and displacement forces released from the user, k is a blade sharpness constant, F_{xy} is force reactions from surface returned through haptic device, and F_d is a damper force added to reduce haptic jerkiness respectively.

4. EXPERIMENTAL SETUP AND IMPLEMENTATION

The proposed framework is implemented on a personal computer (PIV Dual Core 3.2 GHz, 2 GB RAM, 80GB HD), the PHANTOM Haptic premium 1.5 (from Sensable) in Reachin display with stereo LCD shutter glasses (Figure 4). The visual and haptic rendering is controlled using Reachin API 4.2 in combination with Visual Studio.net and Python.

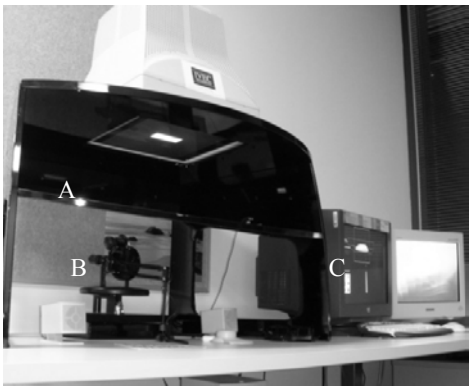
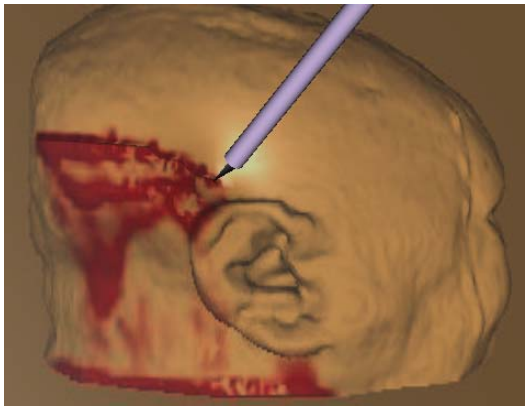


Figure 4. Haptic workbench with Reachin display (A), haptic (B), and work station (C).

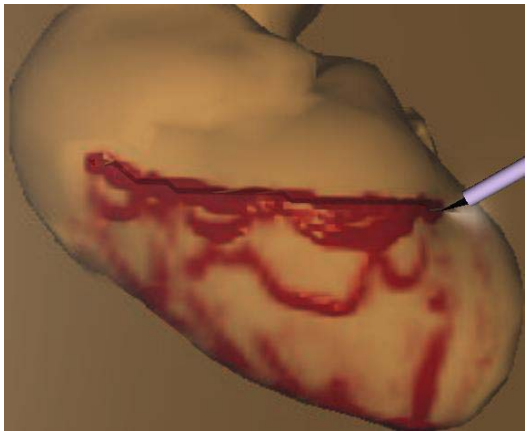
In this study we employ mesh cutting techniques by combining adaptive re-meshing and mesh sub-division for creating surface cutting paths. The cutting paths are created when the following conditions are met: the scalpel collides with a surface, the stylus button is pressed, and the force is applied greater than the surface stiffness.

The force here represents pressure force and the sharpness constant of the blade. The cutting process will be followed by flowing of the fluid. The fluid, representing blood, flows over the surface toward lower parts on the surface or toward the centre of the gravity force.

Several examples of snapshot simulations are shown in figure 4. These examples describe the cutting simulation followed by blood flowing over the surface in the cutting path in a head (figure 5a) and in a heart (figure 5b).



a) A cutting simulation on the head's skin



b) Heart cutting simulation

Figure 5. Interactive cutting over the mesh surface with liquid flowing along the surface paths.

This paper examines several different mesh sizes versus the observed frame rate per second (fps) in the simulation as presented in table (1). As can be seen from the table, the visual/graphic (G) frame rate remain stable for various mesh sizes either with fluid dynamic inclusion (f) or not; however, when a mesh cutting through haptic in combination with fluid dynamic interaction (cf) is included in the simulation the frame rate fall into 18fps for the triangles greater than 29000. We can see that the cutting process reduce the interaction speed. This may be influenced by the re-meshing processes and mesh deformations that need more computation on the CPU. It is also proved from the decreasing frame rate (fr in Hz) on haptic rendering.

The fluid simulation presented here will not go through the detail of bleeding released from the vein or artery when they are cut, since this is beyond the scope of this project. At this time, we only concentrate to control the fluid movement released from the skin when they are cut, regardless the existing vein or artery.

Mesh	Triangles	G_{fps}	$G_{fps(f)}$	$G_{fps(cf)}$	H_{fr}
Full Head	37096	58	58	18	980
Chevy	29304	58	58	18	989
Half Head	23587	58	58	28	989
Bunny	2915	59	59	28	1000
Heart	1619	59	59	28	1000
Torus	576	59	59	28	1000

Table 1. Mesh sizes versus frame rate pers second (fps)

Furthermore, although this paper does not present quantitative data to evaluate the effect of the haptic in user immersion, we verify that the user can communicate kinesthetically in 3D space using this device. Not as in ordinary computer animation that actually only has two and half dimension visual presentation, using haptic device in combination with 3D stereo shutter glasses can also improve user immersions, where the movement can be generated in six degree of freedom with sense of touch and force feedbacks.

5. CONCLUTION AND FUTURE WORK

The computation and simulation of fluid for haptic rendering and cutting procedure can be done at reasonable efficiency using CIP fluid solver that combines the solution of fluid dynamics and its interaction with solid object together. By that using CIP fluid solver and GPU programming, the computation for solving Navier-Stokes equation can be done efficiently without involving supercomputer.

Our results confirm that by combining the CIP solver and GPU programming, the haptic interaction for skin cutting followed by releasing blood from the paths and blood flowing over the anatomical surface can be simulated in a real-time interaction with visually realistic display.

More detail results such as involving two or more fluid and their interactions will be gathered in the next study. This may investigate the fluid dynamic interaction during surgical simulation. The cutting and bleeding involving blood flow from the vein or artery may also be studied in the future.

6. ACKNOWLEDGMENT

The authors wish to thank IVEC (Central TAFE) that provide the haptic equipment, Reachin display, and haptic workbench. The Medical Imaging Department of Curtin University for providing other facilities, and the Exploration Geophysics department that provide spaces for work and equipments. This work is part of the PhD project research in the department of imaging and applied science Curtin University of Technology.

7. REFERENCES

[Anh07],N. H. Anh, A. Sourin and P. Aswani, "Physically based hydraulic erosion simulation on graphics processing unit," in Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia Perth, Australia: ACM New York, NY, USA, 2007.

[Ano01],Anonimous, "Virtual Simulation of Temporal Bone Dissection," in http://www.osc.edu/Biomed/temporal.html (1 of 2) [4/5/2001 8:33:54 AM], 2001.

[Ano05],Anonimous, ReachinAPI 4 Programmer's Guide, Revision No: 27 ed.: Reachin Technology AB, 2005.

[Ben06], B. Benes, V. Tensinsky, J. Hornys and S.K.Bhatia, "Hydraulic erosion," Computer Animation and Virtual Worlds, vol. 17, pp. 99-108, 2006.

[Carl04],M. T. Carlson, "Rigid, Melting, and Flowing Fluid," in College of Computing: Georgia Institute of Technology, 2004.

[Che95], J. X. Chen and N. d. V. Lobo, "Toward Interactive-Rate Simulation of Fluids with Moving Obstacles Using Navier-Stokes Equations," Graphical Models and Image Processing, vol. 57, pp. 107-116, March 1995.

[Fer03],R. Fernando and M. J. Kilgard, The Cg Tutorial: The Definitive Guide to Programmable Real-Time Graphics: Addison-Wesley Longman Publishing Co., Inc., 2003.

[Fos01], N. Foster and R. Fedkiw, "Practical animation of liquids," Proc. of SIGGRAPH 2001, ACM Press/ ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, pp. 23-25, 2001.

[Gre00], A. Gregory, M. C. Lin, S. Gottschalk and R. Taylor, "Fast and accurate collision detection for haptic interaction using a three degree-of-freedom force-feedback device," Computational Geometry, vol. 15, pp. 69-89, February 2000.

[Har04],M. J. Harris, "Fast Fluid Dynamics Simulation on the GPU," in GPU Gems - Triks for Real-Time Graphics, R. Fernando, Ed.: Addison Wesley, 2004, pp. 637-665.

[Hig02], A. Higgins and S. Koucky, "Mission impossible?," Machine Design, vol. 74, p. 28, Dec 12, 2002.

[Lev05],S. Lefebvre, S. Hornus and N. Fabrice, "Octree Textures on the GPU," in GPU Gems 2 - Programming Techniques for High-Performance Graphics and General-Purpose Computation, M. Pharr, Ed.: Addison Wesley, 2005, pp. 595-613.

[Liu04],Y. Liu, X. Liu and E. Wu, "Real-time 3D Fluid Simulation on GPU with Complex

Obstacles," in Computer Graphics and Applications, 2004. PG 2004. Proceedings. 12th Pacific Conference on, 2004, pp. 247-256.

[Mei07],X. Mei, P. Decaudin and B.-G. Hu, "Fast Hydraulic Erosion Simulation and Visualization on GPU," in Computer Graphics and Applications, 2007. PG '07. 15th Pacific Conference on, 2007, pp. 47-56.

[Nei05],B. Neidhold, M. Wacker and O. Deussen, "Interactive physically based fluid and erosion simulation," in Proceedings of Eurographics Workshop on Natural Phenomena, 2005, pp. 25-32.

[Ros00], L. Rosenblum and M. Macedonia, "Virtual Orthopedic Surgery Training," IEEE Computer Graphics and Applications, pp. 6-9, 2000.

[Sta99], J. Stam, "Stable Fluids," Proc. of the 26th annual conference on computer graphics and interactive techniques, pp. 121-128, 1999.

[Sto00], R. J. Stone, "Haptic feedback: A potted hystory, from telepresence to virtual reality," Lecturer notes in computer science, vol. 2058/2001, pp. 1-7, 2000.

[Tak03], T. Takahashi, H. Fujii, A. Kunimatsu, K. Hiwada, T. Saito, K. Tanaka and H. Ueki, "Realistic Animation of Fluid with Splash and Foam," Computer Graphics Forum, vol. 22, pp. 391-400, September 2003.

[Uts97], T. Utsumi, T. Kunugi and T. Aoki, "Stability and accuracy of the Cubic Interpolated Propagation scheme," Computer Physics Communications, vol. 101, pp. 9-20, April 1997.

[Uts04], T. Utsumi, T. Yabe, J. Koga, T. Aoki and M. Sekine, "Accurate basis set by the CIP method for the solutions of the Schrodinger equation," Computer Physics Communications, vol. 157, pp. 121-138, February 2004.

[Li03], W.Li, Z.Fan, X.Wei and A.Kaufman, "GPU-Based flow simulation with complex boundaries," Technical Report 031105, Computer Science Department, SUNY at Stony Brook, 2003.

[Was01], T. M. Wasfy and A. K. Noor, "Visualization of CFD results in immersive virtual environments," Advances in Engineering Software, vol. 32, pp. 717-730, September 2001.

[Wil03],R. William, R. Mark, S. Glanville, K. Akeley and M. J. Kilgard, "Cg: A system for programming graphics hardware in a C-like language," in SIGGRAPH, 2003.

[Uts91a], T. Yabe and T. Aoki, "A universal solver for hyperbolic equations by cubic-polynomial interpolation I. One-dimensional solver," Computer Physics Communications, vol. 66, pp. 219-232, 1991.

[Yab91c], T. Yabe, T. Aoki, G. Sakaguchi, P.-Y. Wang and T. Ishikawa, "The compact CIP (cubic-interpolated pseudo-particle) method as a general hyperbolic solver," Computers & Fluids, vol. 19, pp. 421-431, 1991.

[Yab91b], T. Yabe, T. Ishikawa, P. Y. Wang, T. Aoki, Y. Kadota and F. Ikeda, "A universal solver for hyperbolic equations by cubic-polynomial interpolation II. Two- and three-dimensional solvers," Computer Physics Communications, vol. 66, pp. 233-242, 1991.

[Yab04], T. Yabe, H. Mizoe, K. Takizawa, H. Moriki, H.-N. Im and Y. Ogata, "Higher-order schemes with CIP method and adaptive Soroban grid towards mesh-free scheme," Journal of Computational Physics, vol. 194, pp. 57-77, February 2004.

[Yab02], T. Yabe, Y. Ogata, K. Takizawa, T. Kawai, A. Segawa and K. Sakurai, "The next generation CIP as a conservative semi-Lagrangian solver for solid, liquid and gas," Journal of Computational and Applied Mathematics, vol. 149, pp. 267-277, December 2002.

[Yok02], K. Yokoi and F. Xiao, "Mechanism of structure formation in circular hydraulic jumps: numerical studies of strongly deformed free-surface shallow flows," Physica D: Nonlinear Phenomena, vol. 161, pp. 202-219, January 2002