A New Trajectory Integration Scheme in the Semi-Lagrangian Framework

Suwan Park[1] and Nakhoon Baek[2,1]
[1] *IS Lab, TSonNet Co. Ltd., Daejon 305-343, Korea*
[2] *Kyungpook National University, Daegu 702-701, Korea*
ipsuwan@gmail.com, oceancru@gmai.com

Abstract

In the typical semi-Lagrangian framework for Navier-Stokes equations, the advection source positions for each grid center points are repeatedly estimated to get interpolated physical quantities. Most research efforts are concentrated on the numerical accuracy, especially for the interpolation of target physical quantities. In this paper, we adopted an improved trajectory integration scheme during the advection calculation process. Our method traces the grid center points using their previous velocities at the previous time step, while other previous methods use their current velocities. Our experimental results show remarkable improvements, and we achieved real-time processing capability even with straight-forward serial implementations.

Keywords: *Advection, Fluid Simulation, semi-Lagrangian*

1. Introduction

In the field of computational fluid dynamics and its related areas, there have been various research results to simulate fluid motions in various circumstances. Though they require intensive theoretical foundations and complex mathematical models, real-time computer graphics simulations are now available, mainly due to enhanced computing power and simplified computation models [1, 2, 3, 4, 5].

Most of the fluid simulations in the computer graphics field are based on the Navier-Stokes equations. Foster and Metaxas simulated three-dimensional fluid motions using Navier-Stokes equations with the finite difference method [2]. Since they used an explicit solver, their system may become unstable for relatively large time steps, and thus unsuitable for interactive or real-time fluid simulations. Stam represented a semi-Lagrangian method, which is much more stable even for relatively large time steps [3]. He introduced the pressure projection scheme for supporting incompressible fluids and also semi-Lagrangian methods for calculating advection terms. These techniques have become popular for fluid simulation, including those for smoke [5, 6] and also for more challenging free surface flows [7, 8].

Most semi-Lagrange methods use grid structures and simulate the propagation of physical quantities, mainly at the grid center points. In the typical implementations including Stam's work [3], they first estimate the position of the advection sources for each grid center point, and then interpolate the physical quantities at that source positions from the grid point values

[1] Corresponding author: Nakhoon Baek, oceancru@gmail.com

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology (Grant 2011-0014886).

at the previous time step. To achieve sufficient numerical accuracy, they need to calculate the exact positions of the advection sources and more enhanced interpolations at those locations [9]. In the Stam's framework [3], for example, they retrace the advection positions based on the current velocity of the target grid point, and then apply linear interpolations to estimate their corresponding physical quantities. Due to these numerical approximations, they usually show rapid dissipation compared to real world fluids.

Fedkiw introduced cubic interpolated propagation (CIP) schemes to reduce these dissipations, and also a vorticity confinement term to model the small-scale vortices [6]. Later, various numerical techniques including MCIP (monotonic CIP) [10], USCIP (unsplit semi-Lagrangian CIP) [11], BFECC [12, 13], and stable MacCormack method [14], are introduced to reduce numerical dissipations. Most of them are focused on the numerical accuracy of the interpolation process and/or physical quantity estimation process, still using the same framework to find the advection source locations. Since their backward tracing scheme uses current velocity at the grid point to retrace their advection source positions, the grid points with near-zero velocities fail to get suitable physical quantities. With rapid flows, the estimated positions are actually easy to be directed to unexpected positions.

A set of hybrid approaches based on both of Lagrangian and Euler schemes are adopted to overcome these limitations. Particles with physical quantities are now moving around the grid structures and each grid point uses these particle values. Particle level set method [7], vertex particle method [15], and derivative particles [16] are good examples. In spite of their numerical accuracy, they are difficult to be widely used for real-time applications due to complex calculations for particle motions and their interactions with grid points.

In this paper, we present a new trajectory integration method, which updates physical quantities at the grid center points through retracing their corresponding advection sources in a way of forward calculation, to finally show less numerical dissipation. We adopted the intuitive way of estimating the new positions of the advection sources with their previous velocities, rather than current velocities. Each grid center point is assumed to be moved from an advection source position with its previous velocity, and its physical quantities are updated with respect to those of that position. It is actually an enhanced way of calculating the advection positions in Stam's framework [17], and thus it also has similar characteristics of being stable for large time steps and easy to implement even with more accurate and visually suitable results.

We will start from basic fluid dynamics equations, and our advection term calculation scheme is presented in Section 2. Section 3 shows experimental results. Conclusions and future work are followed in Section 4.

2. Our Method

In this paper, we aimed at simulating the dynamic behavior of fluids. Storing physical quantities such as velocities at the grid center positions, the overall simulation process is based on the Stam's fluid simulation framework [17]. Vorticity confinement forces are additionally applied to simulate small-scale vortices, using Fedkiw's method [6].

The Navier-Stokes equation can be expressed as follows:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} - \frac{1}{\rho}\nabla p + \mathbf{f} \,,$$

where $\mathbf{u}$ is the velocity, ρ is the density of the gas, ∇p is the pressure gradient, and $\mathbf{f}$ is the sum of external forces.

When simulating the fluid motions using the Navier-Stokes equation in a relatively sparse grid, it is easy to miss some small-scale vortices. We can simulate these small-scale vortices with vorticity confinement forces expressed as the followings [6]:

$$\mathbf{f}_c = \varepsilon h(\mathbf{N} \times \omega),$$

where ε is the vorticity confinement constant, h is the length of spatial discretization, $\mathbf{N}$ is the normalized vector field, and the vorticity ω is the rotation axes of the spinning flows in the velocity field.

To evaluate the advection terms with the fluid velocity $\mathbf{u}(t)$ at time t, we assume that fluid flows are streamlined. Precisely, we use the method of characteristics for calculating advection terms in the semi-Lagrangian framework [3]. It is based on the fact that the scalar field preserves the same value along the streamlines along the characteristic path $P(\mathbf{x}_i; \tau)$ as the time τ varies from t_i to $t_{i+1} = t_i + \Delta t$. Thus, letting $\mathbf{x}_i = P(\mathbf{x}_{i-1}; t_i)$, the scalar function f shows the following equalities along the streamlines:

$$f(\mathbf{x}_i, t_i) = f(P(\mathbf{x}_{i-1}, t_i), t_i),$$

Most of the previous methods estimate the departure position of $\mathbf{x}_{i-1}$ using the current velocity $\mathbf{u}(\mathbf{x}_i; t_i)$, as follows:

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \mathbf{u}(\mathbf{x}_i, t_i) \cdot \Delta t.$$

Thus,

$$\mathbf{x}_{i-1} = \mathbf{x}_i - \mathbf{u}(\mathbf{x}_i, t_i) \cdot \Delta t,$$

and the physical quantities can be evaluated as:

$$f(\mathbf{x}_i, t_i) = f(\mathbf{x}_{i-1} + \mathbf{u}(\mathbf{x}_i, t_i) \cdot \Delta t, t_i).$$

This kind of semi-Lagrangian approach enables fast and stable fluid simulations, while it also shows relatively rapid dissipations of physical quantities. To avoid these dissipations, we focused on the numerical errors due to estimating the departure position with the current velocity $\mathbf{u}(\mathbf{x}_i; t_i)$, and modified the above equations to get more reliable results.

In this paper, we calculate the current location $\mathbf{x}_i$ from the previous location $\mathbf{x}_{i-1}$ as follows:

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \mathbf{u}(\mathbf{x}_{i-1}, t_{i-1}) \cdot \Delta t,$$

with the previous velocity $\mathbf{u}(\mathbf{x}_{i-1}; t_{i-1})$ at the departure position $\mathbf{x}_{i-1}$, rather than the current velocity at the target position $\mathbf{x}_i$. A physical quantity f can be evaluated as:

$$f(\mathbf{x}_i, t_i) = f(\mathbf{x}_{i-1} + \mathbf{u}(\mathbf{x}_{i-1}, t_{i-1}) \cdot \Delta t, t_i).$$

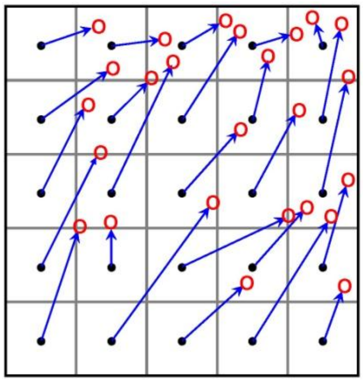


Figure 1. Grid Center Points (black dots) at time t_{i-1} moves to the Current Positions (red circles) at time t_i

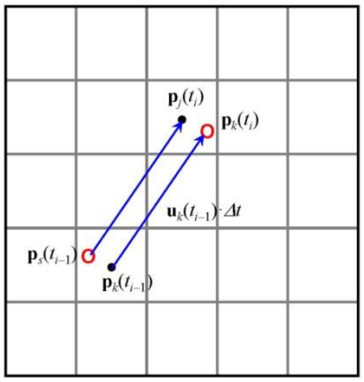


Figure 2. Get the Advection Departure Position $\mathbf{p}_s(t_{i-1})$: grid center point $\mathbf{p}_k(t_{i-1})$ moved to $\mathbf{p}_k(t_i)$, and $\mathbf{p}_s(t_{i-1})$ moved to another grid center point $\mathbf{p}_j(t_i)$

Now, as shown in Figure 1, a grid center point $\mathbf{p}_k(t_{i-1})$ is moved to its new position $\mathbf{p}_k(t_i)$ as follows:

$$\mathbf{p}_k(t_i) = \mathbf{p}_k(t_{i-1} + \Delta t) \approx \mathbf{p}_k(t_{i-1}) + \mathbf{u}_k(t_{i-1}) \cdot \Delta t,,$$

while $\mathbf{u}_k(t_{i-1})$ is its previous velocity at the departure grid point $\mathbf{p}_k(t_{i-1})$. The new position $\mathbf{p}_k(t_i)$ is hard to be another grid center point at time $t_i = t_{i-1}+\Delta$t. Thus, as shown in Figure 2, we calculate the difference vector $\mathbf{p}_j(t_i) - \mathbf{p}_k(t_i)$ with respect to another grid center point $\mathbf{p}_j(t_i)$ to whose grid cell $\mathbf{p}_k(t_i)$ belongs, and estimate the advection departure position $\mathbf{p}_s(t_{i-1})$ as follows:

$$\mathbf{p}_s(t_{i-1}) = \mathbf{p}_j(t_i) - \mathbf{u}_k(t_{i-1}) \cdot \Delta t.,$$

Now, the advection process is performed from the departure point $\mathbf{p}_s(t_{i-1})$ to the grid center point $\mathbf{p}_j(t_i)$.

In the case of sparse grids generally used in Eulerian frameworks, a certain grid cell may be targeted from a set of advection source points or there may be no appropriate source position. Both are abnormal cases. In the former case of duplicated source points, according to our experimental results, there is no remarkable visual difference depending on the choice of source positions. In contrast, we get discontinuities in the case of no available source positions. We choose the best position from a candidate region and apply the same scheme.

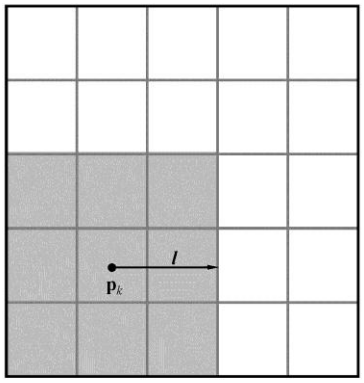


Figure 3. Applying the Search Region with the Average Moving Distance *l*

```
procedure advection($t_i$) begin
    time $t_i = t_i + \Delta t$
    // step 1. forward-tracing
    foreach grid point $\mathbf{p}_k$ begin
        $\mathbf{p}_k(t_i) = \mathbf{p}_k(t_{i-1}) + \mathbf{u}_k(t_{i-1}) \cdot \Delta t$
        get the most nearest grid point, $\mathbf{p}_j(t_i)$
        $\mathbf{p}_s(t_i) = \mathbf{p}_j(t_i) - \mathbf{u}_k(t_i) \cdot \Delta t$
        store $\mathbf{p}_s(t_i)$ as the advection departure position for $\mathbf{p}_j(t_i)$
    end
    // step 2. search region for no-source cases
    calculate $l = (\Delta t / N) \sum |\mathbf{u}_k|$
    foreach grid point begin
        if there is no advection source point then
            set the search region
            get the best available advection position
        endif
    end
    // step 3. perform the advection
    foreach grid point begin
        perform advection simulation for time $t_i$
    end
end procedure
```

Figure 4. Overall Process for each Time Step

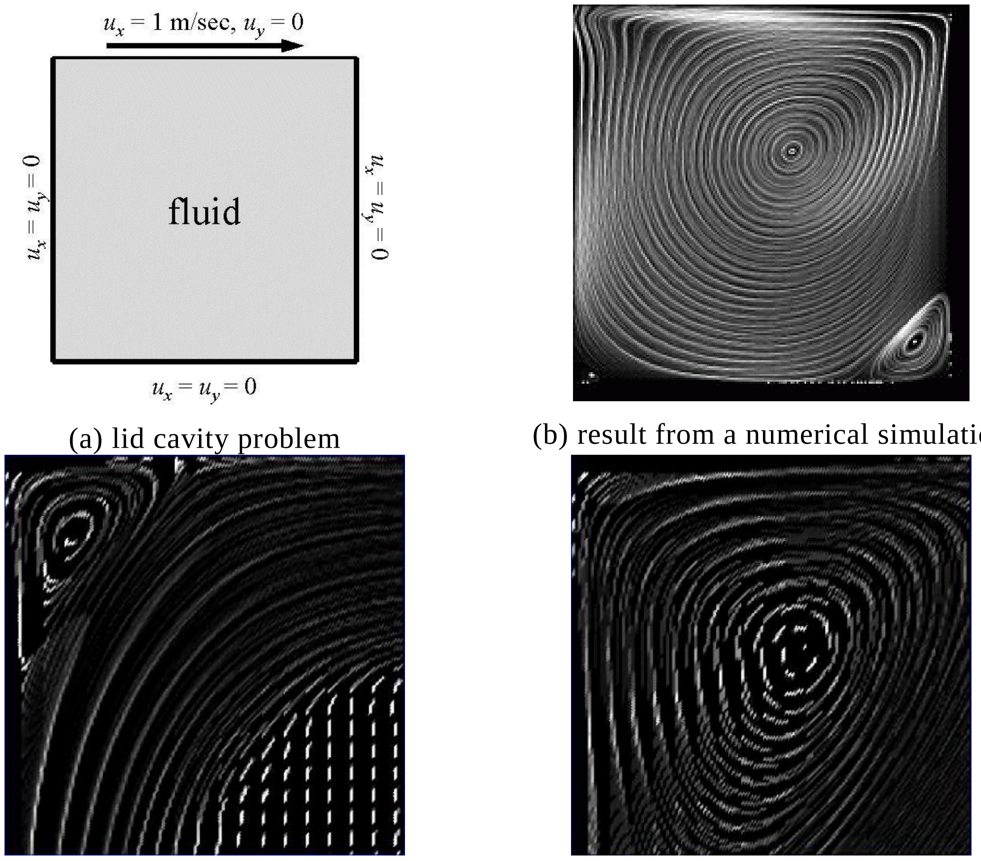
The candidate region is established as a square area whose edge length is calculated from the average moving distance over the whole grid, as shown in Figure 3. The average moving distance l can be calculated as follows:

$$l = \frac{\Delta t}{N} \sum_k |\mathbf{u}_k| ,$$

where $\mathbf{u}_k$ is the velocity for each grid point and N is the total number of grids. This moving distance l is actually clamped to a user-predefined range, to avoid extremely small or large search regions. The overall process is summarized in the pseudo code presented in Figure 4.

3. Implementation Results

Our prototype system is implemented with the Visual C++ programming language with OpenGL and GLUT libraries, using a desktop PC with Intel Core2 Quad Q9550 (2.83GHz), 3G byte RAM and NVIDIA GeForce GTX 260 with 896M byte video RAM.



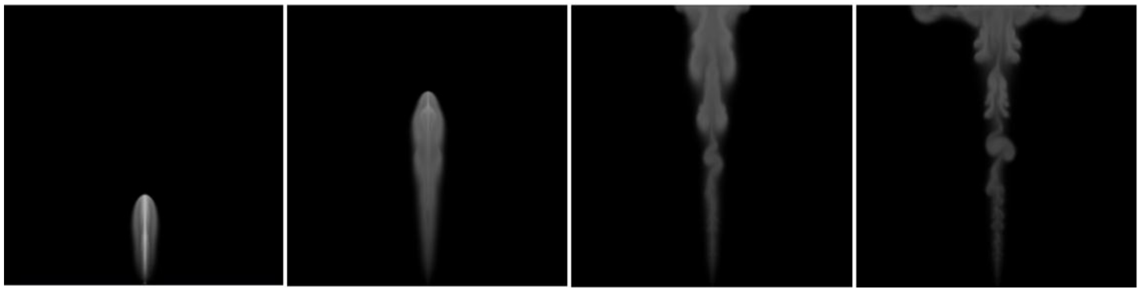
(a) lid cavity problem (b) result from a numerical simulation

(c) Stam's result (d) our result

Figure 5. Simulation Results for the Lid Driven Cavity Problem (with the same 128×128 grids and the same time step Δt = 0:01)

The lid-driven cavity problem has long been used as a test case for new solution methods in fluid simulations [18]. The problem geometry is simple and two-dimensional, and its Dirichlet boundary conditions are also simple. We used a discretized version of 128×128 grids with time step Δt = 0.01, as shown in Figure 5.

Figure 5(b) shows the streamlines from precise numerical simulations, while streamline plots from Stam's and our method are presented in Figures 5 (c) and 5 (d), respectively. Although both of our method and Stam's method are approximation schemes, our result shows similar patterns to the precise numerical simulations, while Stam's failed.


(a) our method

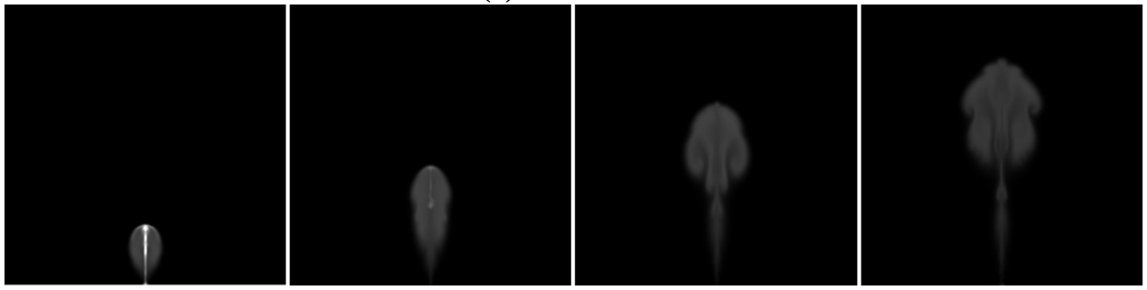

(b) Stam's method

Figure 6. Upward Swirling Smoke Simulation (256×256 grids, Δt = 0.01)

Table 1. Execution Times

(unit: msec)

	grid size			
	64×64	128×128	256×256	512×512
our method	4.7	18.9	103.0	375.9
Stam's method	4.2	17.1	92.4	337.2
ratio	1.119	1.105	1.115	1.115

Figure 6 shows simulation results for upward swirling smokes on the 256×256 grid. The images are shown from left to right, as time goes on. With the same time steps, our results in Figure 6 (a) shows less dissipation in comparison with Stam's in Figure 6 (b). The propagation speed in our simulation is faster since our method also shows less dissipation in the velocity terms.

Table 1 shows total execution times with respect to the various grid resolutions. We used CPU-based serialized implementations for this comparison. Our system only needs less than 12% of total execution time in comparison with Stam's vanilla version.

Additional times are mostly used for region searches in the no-source positions case, and would be almost ignorable when implemented in any GPU-based parallelized ones. Notice that Stam's vanilla version is one the most fast ones among this kind of CPU-based implementations. Conclusively, we achieved remarkable quality improvements with 12% extra execution time, in comparison with one of the fastest methods.

Table 2: Statistics Data for Abnormal Cases

	# of abnormal cases			ratio
	count	average	std. dev.	
after 2,000 frames	526,509	263.25	30.67	0.016
after 5,000 frames	1,456,931	291.39	30.56	0.018

In our forward-tracing method, there may be abnormal cases of no previous propagation source location to a certain grid point. We got statistical results shown in Table 2, which shows only less than 2.0% of grid points belong to these abnormal cases. This ratio will be more reduced as we use smaller time steps.

4. Conclusion

In this paper, we improved the advection source position calculation process in the semi-Lagrangian framework, to finally get the more realistic motions for fluids. Instead of traditional backward-tracing methods, we adopted more intuitive forward-tracing method to get the source position at the previous time step, which will propagate to the current grid point, and then use linear interpolation to get the final physical quantities. Although there were many improvements especially on the interpolation schemes in the semi-Lagrangian scheme, we accomplished more realistic fluid motions through improving the tracing methods, rather than directly improving the numerical accuracy.

Our implementation shows less dissipation in the simulated physical quantities such as velocities, and displays more dynamic behavior of fluid motions. Since our method is based on the Stam's framework, it would be much stable for larger time steps. GPU-based parallelized implementations will be available in near future, as the very next step.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology (Grant 2011-0014886).

References

[1] E. Wu, Y. Liu and X. Liu, “An improved study of real-time fluid simulation on GPU”, *Computer Animation and Virtual Worlds*, vol. 15, no. 3, **(2004)**, pp. 139–146.

[2] N. Foster and D. Metaxas, “Modeling the motion of a hot, turbulent gas”, In SIGGRAPH '99, **(1997)**, pp. 181–188.

[3] J. Stam, “Stable fluids”, In SIGGRAPH '99, **(1999)**, pp. 121–128.

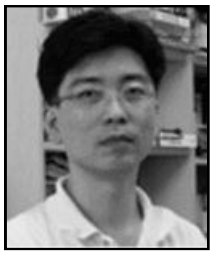
[4] N. Heo and H. -S. Ko, “Detail-preserving fully Eulerian interface tracking framework”, ACM Trans. on Graphics, SIGGGRAPH Asia '10, vol. 29, no. 6, article 176, **(2010)**.

[5] T. Pfaff, N. Thuerey, J. Cohen, S. Tariq and M. Gross, “Scalable fluid simulation using anisotropic turbulence particles”, ACM Trans. on Graphics, SIGGGRAPH Asia '10, vol. 29, no. 6, article 174, **(2009)**.
[6] R. Fedkiw, J. Stam and H. W. Jensen, “Visual simulation of smoke”, In SIGGRAPH '01, **(2001)**, pp. 15–22.
[7] D. Enright, S. Marschner and R. Fedkiw, “Animation and rendering of complex water surfaces”, In SIGGRAPH '02, **(2002)**, pp. 736–744.
[8] N. Foster and R. Fedkiw, “Practical animation of liquids”, In SIGGRAPH '01, **(2001)**, pp. 23–30.
[9] P. Bartello and S. J. Thomas, “The cost-effectiveness of semi-Lagrangian advection,” Monthly Weather Review, 124:2883–2897, 1996.
[10] O. -Y. Song, H. Shin and H. -S. Ko, “Stable but non-dissipative water”, ACM Transactions on Graphics, vol. 24, no. 1, **(2005)**, pp. 81–97.
[11] D. Kim, O. -Y. Song and H. -S. Ko, “A semi-Lagrangian CIP fluid solver without dimensional splitting”, In EUROGRAPHICS '08, **(2008)**, pp. 467–475.
[12] B. -M. Kim, Y. Liu, I. Llamas and J. Rossignac, “Flowfixer: Using BFECC for fluid simulation”, In Eurographics Workshop on Natural Phenomena, **(2005)**.
[13] B. -M. Kim, Y. Liu, I. Llamas and J. Rossignac, “Advections with significantly reduced dissipation and diffusion”, IEEE Transactions on Visualization and Computer Graphics, vol. 13, no. 1, **(2007)**, pp. 135–144.
[14] A. Selle, R. Fedkiw, B. -M. Kim, Y. Liu and J. Rossignac, “An unconditionally stable Maccormack method”, Journal of Scientific Computing, vol. 35, no. 2-3, **(2008)**, pp. 350–371.
[15] A. Selle, N. Rasmussen and R. Fedkiw, “A vortex particle method for smoke, water and explosions”, In SIGGRAPH '05, **(2005)**, pp. 910–914.
[16] O. -Y. Song, D. Kim and H. -S. Ko, “Derivative particles for simulating detailed movements of fluids”, IEEE Transactions on Visualization and Computer Graphics, vol. 13, no. 4, **(2007)**, pp. 711–719.
[17] J. Stam, “Real-time fluid dynamics for games”, In Proc. Game Developer Conference '03, **(2003)**.
[18] http://www.cfd-online.com/Wiki/Lid-driven_cavity_problem.

Authors

Suwan Park

Suwan Park is currently a researcher at TSonNet CO. Ltd. He received his B.A., M.S., and Ph.D. degrees in Computer Engineering from Kyungpook National University. His research interests include computer graphics applications and computer virus engines.



Nakhoon Baek

Nakhoon Baek is currently an associate professor in the School of Computer Science and Engineering at Kyungpook National University, Korea. He received his B.A., M.S., and Ph.D. degrees in Computer Science from Korea Advanced Institute of Science and Technology (KAIST) in 1990, 1992, and 1997, respectively. His research interests include graphics standards, graphics algorithms and real-time rendering. He is now also the Chief Engineer of Mobile Graphics Inc., Korea.