芸術科学会論文誌 Vol. 11, No. 4, pp. 157 – 165

Interactive Simulation of Whirlwind Using Grid and Particles

Satoshi Nakajima[1)] **Issei Fujishiro**[2)]

1) Graduate School of Science and Technology, Keio University
2) Faculty of Science and Technology, Keio University
E-mail: {nakajima, fuji}@fj.ics.keio.ac.jp

Abstract
In this paper, we present an interactive approach to visual simulation of whirlwind using grid and particles. Higher-resolution simulation can be carried out efficiently by perturbing air flow computed on a rough grid with small-scale turbulence. The grid-based simulation first provides a primary whirlwind flow field through modeling its underlying heat convection. Then, the particle-based simulation computes the transport energy of turbulence based on a two-equation $k-\varepsilon$ model, which allows us to capture the turbulence details. The dynamism of whirlwind is conveyed effectively by the interaction with rigid and deformable bodies, such as flag, branch and paper, placed on interactively-controllable trajectories. In addition, a simplified variant of particle-based volume rendering is utilized to delineate changes in the translucency of whirlwind, which reflects the amount of sand particles stirred up from the ground.

Keywords : Whirlwind, visual simulation, $k-\varepsilon$ model, turbulence, and particle-based volume rendering.

1 Introduction

In the field of Computer Graphics (CG), simulation of various natural phenomena has been intensively studied so far. In particular, physically-based simulation is a key integral part because it can offer more realistic and powerful representations of the underlying fluid behavior.

There are many known video works that contain sand dynamics based on fluid simulation. In the CG research, however, most of the works deal only with the interaction of sand with water or solid, whereas few the interaction with wind. In this paper, therefore, we attempt to propose an interactive method for visual simulation of whirlwind. Whirlwind is a phenomenon where updraft occurs since the ground is heated due to solar insolation during daylight, and thereby stirs up sand and dust. Tornado is a large-scale phenomenon generated by cumulonimbus and reaches one hundred meters in height, whereas whirlwind is smaller and just up to ten meters in height, occurring more commonly on the playgrounds and schoolyards, and toppling and/or blowing objects placed on the ground.

In this research, we aim at interactive imaging where whirlwind behavior is represented in a physically-plausible way. To this end, we model the behavior of whirlwind efficiently through coupling Eulerian and Lagrangian solvers. First, the dynamics of surrounding air is roughly computed using a grid-based method, and then detailed motions are added onto the sand particles. Our grid-based solver adopts a heat convection model proposed in Fedkiw et al. [2], which takes into account changes in the density and temperature of the surrounding air to generate updraft. We build upon an energy transport model to compute and synthesize small-scale turbulence directly onto the particles.

To generate realistic whirlwind scenes, three primary factors should be considered, namely interactive trajectory control; interaction with objects; and ground condition effects. We make it possible for the users to use a mouse for interactive control of the whirlwind trajectory. We carry out rigid and deformable body simulations to generate dynamic scenes such that the whirlwind stirs up a piece of small branch and paper and flutters a flag. The amount of sand stirred up can be changed depending on the ground condition so as to have an immediate influence on the translucency of the whirlwind delineated by particle-based volume rendering.

The remainder of this paper is organized as follows.

After related work is surveyed in the next section, Section 3 is devoted to the description of our simulation method. Section 4 focuses on several factors needed to synthesize realistic whirlwind scenes interactively, while Section 5 describes our particle-based method for volumetric rendering of whirlwind. Section 6 illustrates several experimental results. In Section 7, we conclude this paper with several remarks on future research issues.

An initial version of this paper was presented at NICOGRAPH International 2012 [9].

2 Related Work

There are two known basic approaches to solving fluid equations: particle-based and grid-based. Particle-based solvers employ a method for computing the fluid behavior by assuming the fluid to be a set of movable particles. They are suitable for representing detailed fluid motions, while they are not suited to the analysis of aerial flow. On the contrary, grid-based solvers rely on a method that computes the fluid action by covering the surrounding space and the objects with the grid. They can analyze the aerial flow in the simulation space, and thus are suitable for wind simulation. Therefore, the proposed method has adopted the grid-based approach.

Grid-based fluid simulation has become popular in CG with the advent of semi-Lagrangian solver by Stam [15]. The semi-Lagrangian solver is the method for solving for the advection term by tracing each point of the field backward in time and obtaining the new velocity through interpolating the velocity at the old position. Based on this, Fedkiw et al. [2] use a simple model of buoyancy with temperature and density to simulate smoke. However, these methods are not able to represent fluid motions of a scale smaller than the grid size. So, if we would like to represent the detailed motions, we are forced to increase the number of grid points, and hence giving rise to a highly-expensive computational cost.

Related researchers have tackled this kind of problem in many ways. For example, the MacCormack method [14] accounts for secondary derivatives as well for evaluating the advection term, and octree spatial indices are used in [7] to simulate smoke efficiently. These methods can indeed reduce the computational cost to represent detailed motions, but do not make it possible for us to perform the simulation at an interactive frame rate.

Instead, many methods using both grid and particles have been recently studied. Narain et al. [10] use the Particle-in-Cell method for simulating crowded space behavior feasibly, where density-related constraints are newly introduced to accelerate inter-agent collision avoidance. Narain et al. [11] also adapt this approach to sand simulation. They can treat the interaction of sand with solids, whereas the interaction with wind is out of their scope. Pfaff et al. [12] propose a method for high-resolution fluid simulation using a coarse grid and particles. They integrate detailed turbulence computed with the particles into the underlying air flow solved on the low-resolution grid. However, a direct application of the method does not allow us to simulate whirlwind, which stirs up sand furiously.

A work by Liu et al. [6] for simulating tornado is most closely related to ours. They think of tornado as a flow intermixed with air and dust particles to present a Reynolds-Average Two-Fluid Model, where the air flow is simulated by Reynolds-average Navier-Stokes equations, while the motions of dust particles are modeled by non-viscous Navier-Stokes equations. Although this method enables us to generate realistic tornado scenes, interactive tornado control and interaction with deformable objects are limited.

To summarize, there is no existing work which can represent the interaction with sand and wind, and allow the users to operate on the phenomena interactively. In this paper, we attempt to interactively simulate the behavior of whirlwind with small-scale details using both grid and particles.

3 Simulation

In this section, the method for simulating whirlwind is described in detail. An overview of our simulation method is illustrated in Fig. 1. Grid-based simulation, turbulence energy model, and turbulence synthesis method are explained in subsections 3.1, 3.2, and 3.3, respectively.

3.1 Grid-based simulation

In this paper, grid-based simulation is used to analyze the air flow interacting with objects. To begin with, we need to simulate heat convection so as to represent updraft, which occurs due to solar insolation during daylight. To this end, we chose the model presented by Fedkiw et al. [2] for the grid-based simu-

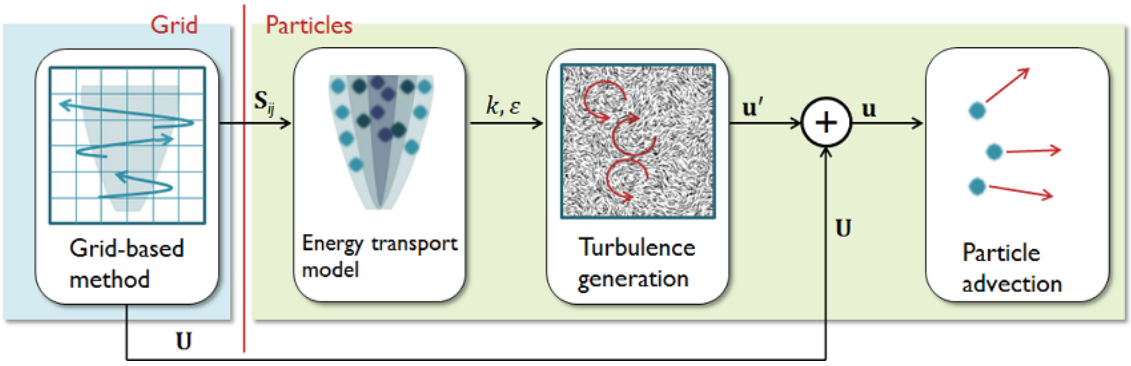


Fig. 1: Overview of whirlwind simulation using grid and particles.

lation. The underlying fluid equations are as follows:

$$\nabla \cdot \mathbf{u} = 0$$
$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{f}. \quad (1)$$

Eq. (1) is termed Navier-Stokes equations, where **u** denotes the velocity of fluid, ρ the fluid density, p the pressure, ν the viscosity coefficient, and **f** the external force.

In this paper, buoyancy is added to the external force of Eq. (1). Buoyancy is the force that is directly proportional to the density and temperature:

$$\mathbf{f_{buoy}} = -\alpha d\mathbf{z} + \beta(T - T_{amb})\mathbf{z}, \quad (2)$$

where d denotes the density of whirlwind, T the temperature, **z** the unit vector in the upward vertical direction, and α and β positive coefficients for adjusting buoyancy. The quantities d and T are advected simply along the whirlwind velocity as

$$\frac{\partial d}{\partial t} = -(\mathbf{u} \cdot \nabla)d \quad (3)$$
$$\frac{\partial T}{\partial t} = -(\mathbf{u} \cdot \nabla)T. \quad (4)$$

Since whirlwind is a phenomenon where updraft generates by heating the ground, this model is suitable for simulating whirlwind. We solve these equations using the semi-Lagrangian method [15].

3.2 Turbulence energy model

It is very inefficient to rely on a high-resolution grid for simulating small-scale turbulence. Instead, we consider herein another scheme which computes the turbulence directly on particles. It is expected to enable us to represent small-scale vortices even with the low-resolution grid. Energy computation for turbulence synthesis uses a k-ε turbulence model [5], which computes the turbulence energy and dissipation effectively.

k-ε models are most-widely used to model turbulence in computational fluid dynamics. On the basis of a flow field **U** computed by the grid-based method, the model governs the two variables: the turbulence energy k and turbulence dissipation ε. Our method adopts the simplified model proposed by Pfaff et al. [12], where k and ε are computed in Eq. (5) and Eq. (6), respectively:

$$\frac{\partial k}{\partial t} = P + \varepsilon \quad (5)$$
$$\frac{\partial \varepsilon}{\partial t} = \frac{\varepsilon}{k}(C_1 P - C_2 \varepsilon), \quad (6)$$

where C_1 and C_2 denote two independent modeling constants with standard values $C_1 = 1.44, C_2 = 1.92$ according to Launder and Sharma [5]. The production P, *i.e.*, energy transfer from the large-scale flow field **U** to small-scale turbulence is defined in terms of strain of the coarse flow S_{ij} as

$$P = 2\nu_T \sum_{ij} {S_{ij}}^2 \quad (7)$$
$$\nu_T = C_\mu \frac{k^2}{\varepsilon} \quad (8)$$
$$S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right), \quad (9)$$

where ν_T denotes turbulent viscosity, C_μ an empirical constant with standard values $C_\mu = 0.09$ according to Launder and Sharma [5], and i and j the row and column numbers of strain tensor, respectively. .

Originally, there exist advection and diffusive terms in the k-ε model. In Pfaff's model, however, solving for advection term can be omitted because it is inherently handled by the motions of the particles. Also, the turbulent particle motions cause mixing as particles cross paths, so incorporating additional diffusion becomes visually-unnecessary. As the result, we can avoid solving for this term as well. The turbulence energy and dissipation are used for turbulence synthesis, which will be explained in the next subsection.

3.3 Turbulence synthesis

In order to synthesize turbulence, the method proposed by Kim et al. [4] is used. The wavelet noise function generates noise tiles $\omega_{i=1,2,3}$ which are scalar fields computed by Kim's method [4]. Wavelet noise is guaranteed to exist only over a narrow spectral band. We can construct 3D noise textures $\mathbf{w}$ by taking the curl of the noise tiles:

$$\mathbf{w} = \left(\frac{\partial \omega_1}{\partial y} - \frac{\partial \omega_2}{\partial z}, \frac{\partial \omega_3}{\partial z} - \frac{\partial \omega_1}{\partial x}, \frac{\partial \omega_2}{\partial x} - \frac{\partial \omega_3}{\partial y} \right). \quad (10)$$

Synthesizing multiple bands of the noise texture, final turbulence function $turbulence\,(\mathbf{q})$ is formulated by:

$$turbulence\,(\mathbf{q}) = \sum_{i}^{N} \mathbf{w}\left(2^i \mathbf{q}\right) 2^{-\frac{5}{6}i}, \quad (11)$$

where $\mathbf{q}$ denotes the texture coordinates, and N the spectral band number. In our results, we use eight bands. Each particle velocity $\mathbf{u}$ is computed by integrating turbulence velocity into $\mathbf{U}$ as follows:

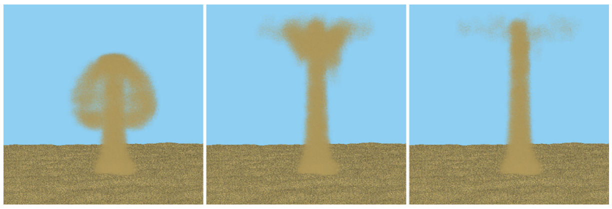
$$\mathbf{u} = \mathbf{U} + 2\left(\alpha_t k\right)^{\frac{1}{2}} turbulence\,(\mathbf{q}), \quad (12)$$

where α_t denotes a scaling parameter. Turbulence velocity is scaled by α_t and turbulence kinetic energy k. Fig. 2 shows an example which compounds the turbulent flow.

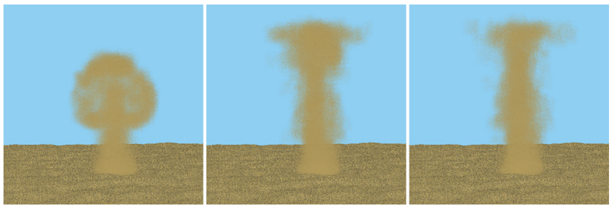
The simulation code is summarized in Algorithm 1.

Algorithm 1 Simulation using grid and particles.

Semi-Lagrangian advection of $\mathbf{U}$
Calculate strain field S_{ij}
for each particle **do**
  Sample $\mathbf{U}$, S_{ij} at position $\mathbf{x}$
  Compute turbulent viscosity $\nu_T :\leftarrow$ Eq. (8)
  Compute Production $P :\leftarrow$ Eq.(7)
  Update $k \leftarrow k + \Delta t\,(P - \varepsilon)$
  Update $\varepsilon \leftarrow \varepsilon + \Delta t \frac{\varepsilon}{k}\,(C_1 P - C_2 \varepsilon)$
  Generate turbulence $turbulence\,(q)$
  Synthesize turbulence to $\mathbf{U}$: $\mathbf{u} \leftarrow$ Eq. (12)
  Update $\mathbf{x} \leftarrow \mathbf{x} + \Delta t \mathbf{u}$
  Update $\mathbf{q} \leftarrow \mathbf{q} + \Delta t \mathbf{u}$
end for



(a) Only grid-based simulation.



(b) Grid- and particle-based simulation.

Fig. 2: The difference in whirlwind behavior when sand being injected upward. (a) The result with only a grid-based method, (b) the result integrating turbulence into (a). The shape of (a) keeps symmetry while (b) possesses detailed behavior.

4 Interaction

In this section, we describe several factors to support various interactive scenarios for generating realistic whirlwind-involved scenes.

4.1 Interactive trajectory control

To generate desirable scenes for film making, it is required to interactively control the trajectory of whirlwind. In our approach, the users are allowed to use a mouse to change the horizontal location of the whirlwind. The simulation space is given its own local coordinates, separately from the world coordinates for the ground. The position of the particles in the world coordinates is given by adding the distance dragged with the mouse to the position in the local coordinates (Fig. 3). During the interactive control of whirlwind trajectory, interaction with objects and ground condition effects, described in subsections 4.2 and 4.3, can be evaluated on the fly.

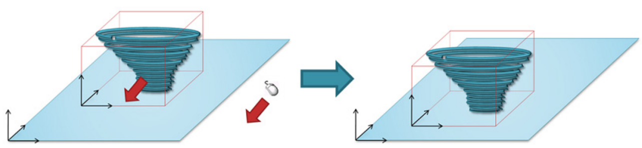
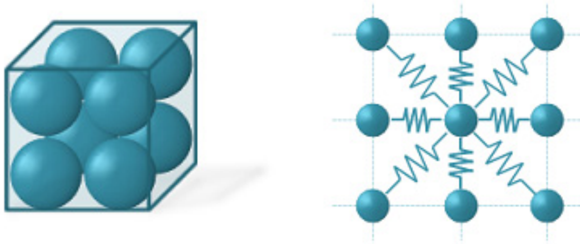


Fig. 3: Interactive control of whirlwind trajectory.

4.2 Interaction with objects

As described in Section 1, the dynamism of whirlwind can be conveyed effectively through the interaction with objects. Pieces of small branch and paper may be easily blown away while a flag may flutter sharply. In order to represent these scenes, it is necessary to perform rigid and deformable body simulations.



(a) Particle representation of rigid cube.

(b) Particle-spring representation of deformable object.

Fig. 4: Particle representations of rigid and deformable bodies.

4.2.1 Rigid body simulation

Rigid body simulation employs a particle-based method [16]. A rigid body is represented by a set of particles (Fig. 4(a)), and the force to rotate the object is computed from the external forces impressed upon each particle. Then, the angular velocity and rotational matrices are derived from the rotational force.

If particles are translated or rotated incrementally at a relatively large time-step, the shape of the rigid body may not be maintained. In order to relocate each particle properly, its present position can be obtained through left-multiplying the relative position vector from the center in its initial state by the rotational matrix of rigid body, and adding the obtained vector to the present position vector of the center.

4.2.2 Deformable body simulation

Deformable body simulation is also based on a particle-based method [3]. The method adopts the mass-spring model (Fig. 4(b)), where particles are inter-connected horizontally and vertically, and even diagonally by means of 3D network of springs. The velocity and position of each particle are computed from the length of corresponding springs expanded by the impressed force.

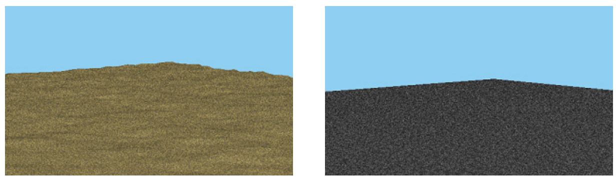
Since the Euler method is likely to become numerically-unstable, our time integral computation employs the Verlet method, which computes the position of a particle at the time $t+\Delta t$ using the position at the time t and $t-\Delta t$, as follows:

$$\mathbf{x}_i(t+\Delta t) = \mathbf{x}_i(t)+dp(\mathbf{x}_i(t)-\mathbf{x}_i(t-\Delta t))+\frac{d^2\mathbf{x}}{dt^2}\Delta t^2, \quad (13)$$

where $\mathbf{x}_i$ denotes the position of a particle, and dp a damping coefficient.

4.3 Ground condition effects

The amount of sand stirred up by whirlwind changes drastically depending on the state of ground. In this paper, two kinds of ground condition are considered: sand area and asphalt area (Fig. 5). As time passes by, the whirlwind residing in the asphalt area gains its the translucency due to the undersupply of sand particles, and finally looks invisible.



(a) Sand area.

(b) Asphalt area.

Fig. 5: Two kinds of ground condition.

5 Rendering

In order to directly visualize particle-represented whirlwind in a semi-translucent manner, we employed a method called Particle-based Volume Rendering (PBVR) [13]. The PBVR was originally proposed in the field of volume visualization, while it was recently simplified for rendering particle-represented flame by Mabuchi et al. [8].

Fig. 6 gives an overview of the simplified PBVR, which generates rendering particles randomly according to the local density of the original particles, and then projects the generated particles repeatedly onto the Z-buffer to obtain the final distribution of luminance. We can increase the number of rendering particles artificially by mapping an alpha texture onto each of the particles, and then performing alpha test to the accumulated projection.

Fig. 7 compares rendered whirlwinds without PBVR (a) and with PBVR (b). In Fig. 7(a), mapped textures can be seen individually, whereas in Fig. 7(b), the projected texture gets blurred and the whirlwind is delineated more natunally, just like smoke.

If we alter the color transfer function, we can easily imitate various phenomena. Fig. 8 shows fire whirlwind as an example.

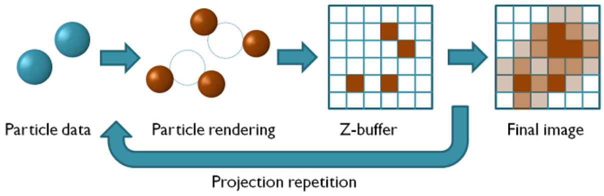
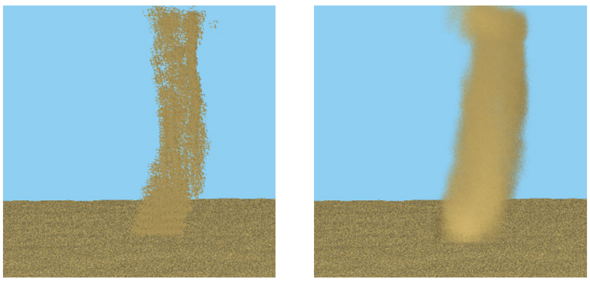


Fig. 6: An overview of simplified PBVR.



(a) Result without PBVR. (b) Result with PBVR.

Fig. 7: Comparison of rendered whirlwinds without PBVR (a) and with PBVR (b).

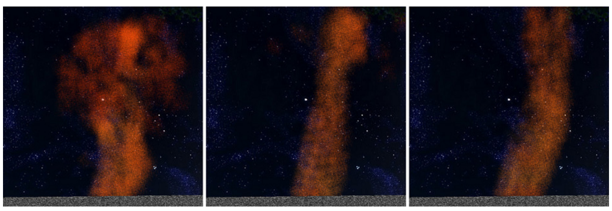


Fig. 8: Fire whirlwind rendered with a specific color transfer function.

6 Results

The code was implemented in C++ with an OpenGL API. All simulations were run on a personal computer with an Intel(R) Xeon(R) X5492 3.40GHz CPU and a 2.00GB RAM. The height field used to represent the ground is commonly discretized into $63 \times 63 \times 2$ triangles, and the resolution of synthesized images is 512×512. The grid resolution of all simulations is $32 \times 32 \times 32$. We set on the ground the initial value velocity field to form a vortex and the temperature field which gets warm closer to the vortical center. The initial values for the velocity and temperature fields need to be set manually before running the simulation. For the simplified PBVR, we generated two rendering particles at random positions near each of the original particles, and the projection was repeated 20 times to generate a single blurred frame. See the accompanying animations for more details.

Fig. 9 on the next page compares simulated whirlwinds in terms of behavior. For both cases, we used approximately 2,300 physical particles, and the frame rate was approximately 9 fps (including rendering). Fig. 9(a) shows the result of simulation only with a grid-based method. The resultant whirlwind has a monotonous behavior. On the other hand, Fig. 9(b) shows the result of simulation using grid and particles. Unlike the result in Fig. 9(a), we can observe that detailed turbulence makes the shape of the whirlwind more natural.

We can observe from Fig. 10 that the whirlwind changes its appearance by passing over a fixed object. Whirlwind is not allowed to flow into an object domain by imposing a free-slip boundary condition at the fixed object boundary. We used 50 particles to model the fixed object. The simulation ran at approximately 9 fps as for the results in Fig. 9.

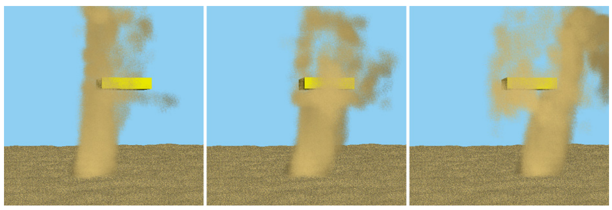
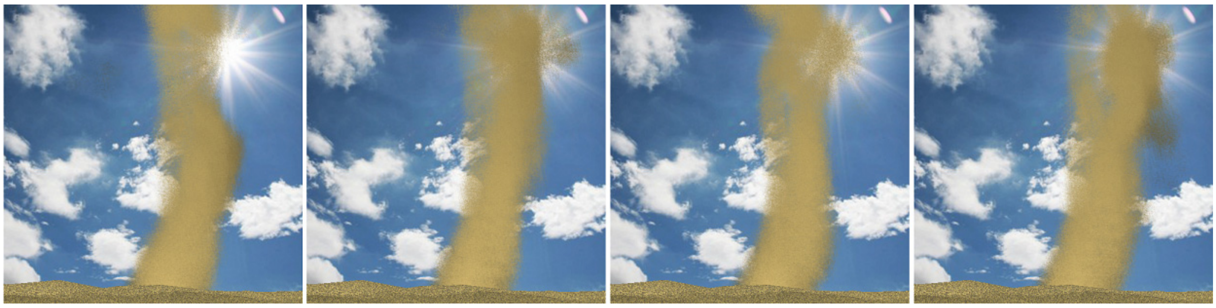


Fig. 10: Interaction with a fixed object.

Fig. 11 exemplifies how the whirlwind interacts with the objects which change their positions and/or shapes caused by the whirlwind. In Fig. 11(a), the whirlwind interacts with the small branch, and blows it away. In Fig. 11(b), the whirlwind interacts with the flag when passing through. The flag droops when the whirlwind stays off, whereas the flag starts to flutter as the whirlwind approaches.

Fig. 12 demonstrates more complex interaction with four objects, namely flag, branch, paper and brick wall. Note that the whirlwind is affected from the brick wall whereas the other objects are affected from the whirlwind. We used 40, 7, 16, 200 particles to model the flag, branch, paper and brick wall, respectively. The frame rate of the simulation was

(a) Simulation only with grid-based method.



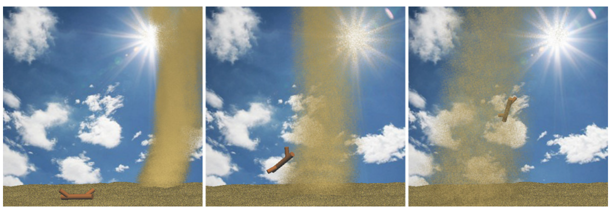
(b) Simulation using grid- and particle-based methods.

Fig. 9: Comparison of simulated whirlwind behavior.

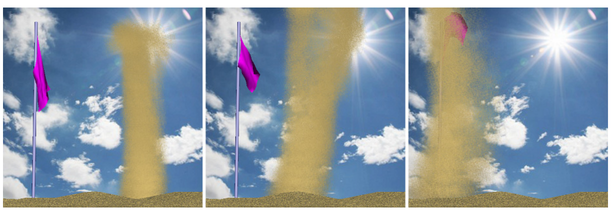
decreased a bit down to approximately 7 fps.

Finally, Fig. 13 shows how the whirlwind changes its appearance depending on the ground condition. For the sand and asphalt areas, 100 and 16 sand particles are generated every frame, respectively. The whirlwind above the sand area looks dense, whereas it gets thinner when moving to the asphalt area.

Note that the results in Figs. 10 and 12 are newly added to the early work [9].



(a) Interaction with a branch modeled with 7 particles.



(b) Interaction with a flag modeled with 40 cloth particles.

Fig. 11: Interaction of whirlwind with objects.

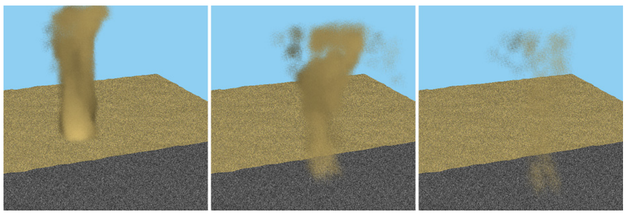


Fig. 13: The change in whirlwind appearance depending on ground condition.

7 Conclusions

We have presented a novel interactive approach to visual simulation of whirlwind. Using grid and particles, we are allowed to represent whirlwind with detailed turbulence at a small computational cost. In addition, interactive trajectory control, interaction with the objects, and ground condition effects were considered to generate dynamic whirlwind scenes. Semi-translucent appearance of whirlwind was visualized efficiently using a simplified scheme of particle-based volume rendering.

We have many issues to be addressed in our future work. To begin with, it is necessary to compare our results with the results of using tornado simulation method [6] and the real photographs. In addition, we have to consider the computational model which

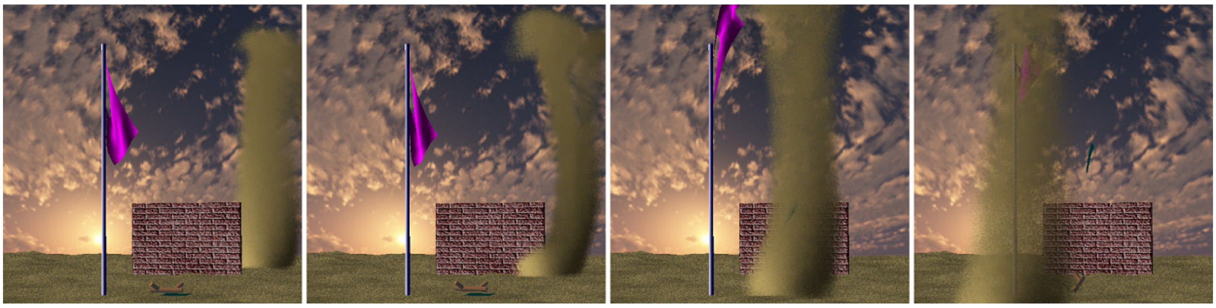


Fig. 12: Complex interaction with four objects.

adds a swirling force like Coriolis force to initiate the whirlwind more naturally. The present approach shows only one-way interaction of whirlwind with objects. To increase the accuracy of collision detection and reaction with larger-scale objects, two-way interaction will have to be computed. More generalized ground conditions in terms of geometry, such as slopes, bumps, and terrain, should be able to be specified. More sophisticated rendering should also be taken into account to give more realistic depth cues to rendered whirlwinds. Self-shadowing and light scattering are representative candidates for this. Moreover, we plan to make use of the GPU for speeding up the entire process of simulation and rendering.

Acknowledgements

This work has been partially supported by JSPS and Keio University under a Grant-in-Aid for Scientific Research (B) No. 22300037 and an Institutional Program for Young Researcher Overseas Visits, respectively.

References

[1] Cook, L. R. and DeRose, T.: "Wavelet Noise," *ACM Transactions on Graphics*, Volume 24, Issue 3, pages 803–811, July 2005.

[2] Fedkiw, R., Stam, J., and Jensen, W. H.: "Visual Simulation of Smoke," *Proceedings of ACM SIGGRAPH 2001*, pages 15–22, August 2001.

[3] Harada, T., Koshizuka, S., and Kawaguchi, Y.: "Real-time Fluid Simulation Coupled with Cloth," *Proceedings of Theory and Practice of Computer Graphics 2007*, pages 13–20, June 2007.

[4] Kim, T., Thuerey, N., James, D., and Gross, M.: "Wavelet Turbulence for Fluid Simulation," *ACM Transactions on Graphics*, Volume 27, Issue 3, Article 50, August 2008.

[5] Launder, B. E. and Sharma, D. B.: "Applications of the Energy-dissipation Model of Turbulence to the Calculation of Flow near a Spinning Disc," *Letters in Heat and Mass Transfer*, Volume 1, Issue 2, pages 131–137, November 1974.

[6] Liu, S., Wang, Z., Gong, Z., and Peng, Q.: "Real Time Simulation of a Tornado," *The Visual Computer*, Volume 23, Issue 8, pages 559–567, July 2007.

[7] Losasso, F., Gibou, F., and Fedkiw, R.: "Simulating Water and Smoke with an Octree Data Structure," *ACM Transactions on Graphics*, Volume 23, Issue 3, pages 457–462, August 2004.

[8] Mabuchi, S., Ohno, Y., and Fujishiro, I.: "SPH-Based Method for Interactive Flame Simulation — Implementation on CPU and GPU," *Proceedings of ASIAGRAPH 2011*, pages 41–46, October 2011.

[9] Nakajima, S. and Fujishiro, I.: "Visual Simulation of Whirlwind Using a Grid and Particles," *Proceedings of NICOGRAPH international 2012*, pp. 27–34, July 2012.

[10] Narain, R., Golas, A., Curtis, S., and Lin, M. C.: "Aggregate Dynamics for Dense Crowd Simulation," *ACM Transactions on Graphics*, Volume 28, Issue 5, Article 122, December 2009.

[11] Narain, R., Golas, A., and Lin, M. C.: "Free-Flowing Granular Materials with Two-Way Solid Coupling," *ACM Transactions on Graphics*, Volume 29, Issue 6, Article 173, December 2010.

[12] Pfaff, T., Thuerey, N., Cohen, J., Tariq, S., and Gross, M.: "Scalable Fluid Simulation Using Anisotropic Turbulence Particles," *ACM Transactions on Graphics*, Volume 29, Issue 6, Article 174, December 2010.

[13] Sakamoto, N., Nonaka, J., Koyamada, K., and Tanaka, S.: "Particle-based Volume Rendering," *Proceedings of Asia-Pacific Symposium on Visualization 2007*, pages 129–132, February 2007.

[14] Selle, A., Fedkiw, R., Kim, B., Liu, Y., and Rossignac, J.: "An Unconditionally Stable MacCormack Method," *Journal of Scientific Computing*, Volume 35, Issue 2-3, pages 350–371, June 2008.

[15] Stam, J.,: "Stable Fluid," *Proceedings of ACM SIGGRAPH 1999*, pages 121–128, August 1999.

[16] Tanaka, M., Sakai, M., and Koshizuka, S.: "Rigid Body Simulation Using a Particle Method," *ACM SIGGRAPH 2006 Research Posters*, No.132, August 2006.

Satoshi Nakajima

Satoshi Nakajjima is a master course student at the Graduate School of Science and Technology, Keio University. He received his Bachelor of Engineering from Department of Information and Computer Science, Keio University in 2011. His research interest includes natural phenomena simulation.

Issei Fujishiro

Issei Fujishiro is currently a professor in Department of Information and Computer Science, Faculty of Science and Technology, Keio University. He received his Master of Engineering in information sciences and electronics from University of Tsukuba in 1985 and his Doctor of Science in information sciences from The University of Tokyo in 1988. His major research interests include volume graphics, visualization lifecycle management, and multi-modal information display. He is currently serving on the editorial board for Computers and Graphics, and as a vice president for this society. He is a member of IEEE Computer Society, ACM, Eurographics Association, IPSJ, IIEEJ, JSCES, and VRSJ.