Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Physically based modeling and animation of tornado*

LIU Shi-guang[†1,2], WANG Zhang-ye[†‡1], GONG Zheng[1], CHEN Fei-fei[1], PENG Qun-sheng[1,2]

([1]*State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, China*)

([2]*Department of Mathematics, Zhejiang University, Hangzhou 310027, China*)

[†]E-mail: lsg@cad.zju.edu.cn; zywang@cad.zju.edu.cn

Received Apr. 7, 2006; revision accepted Apr. 19, 2006

Abstract: Realistic modeling and rendering of dynamic tornado scene is recognized as a challenging task for researchers of computer graphics. In this paper a new physically based method for simulating and animating tornado scene is presented. We first propose a Two-Fluid model based on the physical theory of tornado, then we simulate the flow of tornado and its interaction with surrounding objects such as debris, etc. Taking the scattering and absorption of light by the participating media into account, the illumination effects of the tornado scene can be generated realistically. With the support of graphics hardware, various kinds of dynamic tornado scenes can be rendered at interactive rates.

Key words: Natural phenomenon, Tornado scene, Physically based simulation, Two-Fluid model
doi:10.1631/jzus.2006.A1099 **Document code:** A **CLC number:** TP39

INTRODUCTION

Tornado is one of the most terrible natural phenomena in the world. It causes incredible amounts of damage and destroys significant numbers of facilities every year. No wonder it receives great attention as a major research topic in many fields. Realistic simulation of tornado is also a great challenge for computer graphics researchers. It provides useful information for natural disaster prevention and can be integrated into entertainment, computer games, etc.

Most tornadoes are caused by an especially intense weather system known as a supercell. A supercell is formed when warm, moist air comes in contact with heavier, cooler, drier air. The resulting instabilities inspire powerful vortex motions. Within the fiercest tornadoes, wind speeds can exceed three hundred miles per hour. Air rushes in to fill the low-pressure void left by the tornado, and thus creates additional fierce, potentially damaging winds. Its shape is usually like a huge funnel twisted below the convective cloud. Every tornado has four primary stages in its duration or life. These stages are referred to as the organization, mature, shrinking and decaying stages (Flora, 1953).

Most tornado researches are focused on laboratory observation, mathematical modeling and numerical simulation. None of these methods can generate the realistic visual tornado images. Our intention is to combine the advantage of numerical simulation with computer graphics techniques and demonstrate the tornado flow movement and its interaction with surrounding objects such as debris, etc. based on physical model. The main contributions of this paper include:

(1) A physically based model for simulating the flow movement and color of tornado;

(2) An efficient approach to calculate the light scattering and absorption of the participating media making the tornado scene more realistic;

(3) Hardware acceleration enabling the rendering of tornado scene at interactive rate.

The rest of this paper is organized as follows. In

‡ Corresponding author

* Project supported by the National Basic Research Program (973) of China (No. 2002CB312101) and the National Natural Science Foundation of China (No. 60475013)

Section 2 we give a brief survey of related work. In Section 3 we propose the Two-Fluid model and give its implementation. Section 4 describes how the illumination effects of the tornado scene are rendered. We give the rendering results in Section 5. Conclusions are drawn at last.

RELATED WORK

Extensive studies on this phenomenon have been conducted in the field of physics, meteorology, etc. Lewellen (1993) reviewed previous work and proposed the tornado vortex theory. It provided a new basis for later researches. Lewellen and Lewellen (1997) used LES (large-Eddy Simulation) method to study the tornado dynamics. Nolan and Farell (1999) explored the structure and dynamics of axisymmetric tornado-like vortices based on a numerical model of incompressible flow. It is well known that strong tornadoes can transport substantial quantities of debris at high velocity and hence increasing their damage potential. It is therefore important to study the interaction between tornado and surrounding debris. Xia *et al*.(2003) investigated whether debris loading could significantly affect the fluid-dynamic structure of a tornado itself and gave some numerical simulation results. However, all the above works are based on mathematical modeling and numerical simulation, and focused mainly on the velocity, pressure and temperature fields without simulating a realistic dynamic tornado scene to show its movement, shape and color, thus making the simulation less visually interesting.

In order to promote understanding of a tornado's velocity field, many visualization methods on a tornado's velocity field were proposed. Xue and Crawfis (2004) developed the Texture Splats algorithm for direct flow volume rendering of vector fields. Several anisotropic textured splats were exploited to implement flow volume rendering. Weiskopf *et al*.(2005) presented an interactive technique for the dense texture-based visualization of unsteady 3D flow, taking into account issues of computational efficiency and visual perception. High efficiency is achieved by a novel 3D GPU-based texture advection mechanism. But, all the above methods rely on real data, which are very difficult to acquire.

On the other hand, realistic visual simulations of tornado are very much in demand in many domains such as digital entertainment, movie, science research, etc. But in the field of computer graphics, little research has been done. Ding (2004) proposed an approach for tornado simulation. He generated the tornado flow movement by solving Navier-Stoke equation. A particle system was then introduced to define the tornado volume and a volume rendering algorithm was used to visualize the flow movement based on the particle density. As large number of particles are introduced, the rendering speed is not fast. Besides, he did not take into account the light scattering effect of participating media. The illumination effect of the tornado scene was hence less realistic.

It was found that to realistically simulate the tornado scene, we must integrate computer graphics techniques with physically based simulation model. In addition to the tornado's velocity field and flow movement, its interaction with surrounding objects must also be considered. The illumination effects and the participating media play important roles in demonstrating a tornado scene.

PHYSICAL MODEL OF TORNADO

In the field of numerical simulation of tornado, most works are based on the Navier-Stokes equation. They mainly concern one kind of fluid, such as water, smoke, fire, etc. (Foster and Fedkiw, 2001; Fedkiw *et al*., 2001; Nguyen *et al*., 2003). However, natural tornado cannot be treated as the movement of one fluid. Here, we propose a Two-Fluid model to simulate this phenomenon.

Two-Fluid model

Tornado itself is actually the clotted vapor due to the sudden decrease of air pressure. We treat it as the primary fluid. The debris turning inside is treated as the second fluid. Both can be considered as incompressible fluid. When the debris is small enough, the second fluid can be considered as pressureless. Given the large ratio of the density of the debris such as sand, dirt, etc. to that of clotted vapor, this condition is satisfied. Suppose the second fluid is composed of spherical particles, our Two-Fluid model can be expressed as follows (Marble, 1970):

$$\nabla\cdot\boldsymbol{u}=0,\ \nabla\cdot\boldsymbol{u}_{\mathrm{d}}=0,$$
$$\partial\boldsymbol{u}/\partial t=-(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}-\nabla p-\boldsymbol{F}_{\mathrm{d}}, \quad (1)$$
$$\partial\boldsymbol{u}_{\mathrm{d}}/\partial t=-(\boldsymbol{u}_{\mathrm{d}}\cdot\nabla)\boldsymbol{u}_{\mathrm{d}}-\nabla p_{\mathrm{d}}+(\boldsymbol{F}_{\mathrm{d}}+m\mathrm{g}),$$

where $\boldsymbol{u}$ is the velocity field of the primary fluid, $\boldsymbol{u}_{\mathrm{d}}$ is the velocity field of the second fluid, both are three dimensional vectors. $\nabla\cdot$ and ∇ are divergence operator and gradient operator, respectively. $\partial/\partial t$ denotes a differential operator. p and p_{d} are the pressure field of the primary fluid and second fluid, respectively. g is acceleration of gravity. Suppose m and d are the mass and diameter of each spherical particle in the second fluid. $\boldsymbol{F}_{\mathrm{d}}$ is the interaction force between the two fluids which is a function of Reynolds number Re. We use it to simulate the interaction between tornado and its surrounding debris. $\boldsymbol{F}_{\mathrm{d}}$ can be expressed as:

$$\boldsymbol{F}_{\mathrm{d}}=\rho_{\mathrm{d}}\frac{\boldsymbol{u}_{\mathrm{d}}-\boldsymbol{u}}{\tau_{\mathrm{v}}}, \quad (2)$$

where

$$\tau_{\mathrm{v}}\approx\frac{m}{3\pi d\mu_{\mathrm{d}}}\left(1+\frac{Re}{60}+\frac{Re/4}{1+\sqrt{Re}}\right)^{-1}, \quad (3)$$

$$Re=\rho_{\mathrm{d}}d|\Delta\boldsymbol{u}|/\mu_{\mathrm{d}}. \quad (4)$$

In Eq.(2), ρ_{d} and μ_{d} are the density and viscosity coefficients of the second fluid, respectively. ρ_{d} is equal to 2500 kilogram per cubic meter.

Boundary conditions

The boundary conditions are essential in tornado simulation to achieve the rotation and uplifting movement and can affect the tornado's shape. We use four kinds of boundary conditions:

(1) No-slip condition, that is, the fluid is at rest there and can be given as:

$$\varphi_{\mathrm{n}}(x,y,z)=0,\ \varphi_{\mathrm{t}}(x,y,z)=0,$$

where $\varphi_{\mathrm{n}}(x,y,z)$ is the component of velocity orthogonal to the boundary in the exterior normal direction at position (x,y,z), $\varphi_{\mathrm{t}}(x,y,z)$ is the component of velocity in the tangential direction of 3D boundary.

(2) Free-slip condition, that is, no fluid penetrates the boundary and can be given as:

$$\varphi_{\mathrm{n}}(x,y,z)=0,\ \partial\varphi_{\mathrm{t}}(x,y,z)/\partial n=0,$$

where $\partial\varphi_{\mathrm{t}}(x,y,z)/\partial n$ is the first derivative of $\varphi_{\mathrm{t}}(x,y,z)$.

(3) Inflow condition, that is, the value of the velocity component is given explicitly and the flow can go in and out of the boundary and can be given as:

$$\varphi_{\mathrm{n}}(x,y,z)=\varphi_{\mathrm{n}}^{0},\ \varphi_{\mathrm{t}}(x,y,z)=\varphi_{\mathrm{t}}^{0},$$

where φ_{n}^{0}, φ_{t}^{0} are given values changing with time.

(4) Outflow condition, that is, the velocity does not change in the direction normal to the boundary and can be given as:

$$\partial\varphi_{\mathrm{n}}(x,y,z)/\partial n=0,\ \partial\varphi_{\mathrm{t}}(x,y,z)/\partial n=0,$$

where $\partial\varphi_{\mathrm{n}}(x,y,z)/\partial n$ is the first derivative of $\varphi_{\mathrm{n}}(x,y,z)$. Our boundary conditions are similar to those adopted by Trapp and Fiedler (1993).

Solution of the Two-Fluid model

We discritize the space into three dimensional cells and calculate the model at the center of each cell. A vertical section is shown in Fig.1, where the gray part and the white part are the boundary strip and the interior domain, respectively. We express a velocity vector $\boldsymbol{u}$ as (u,v,w). For a particular cell in 3D, pressure p is located at the cell center, while u, v and w are at the center of the right, back and top planes of that cell, respectively.

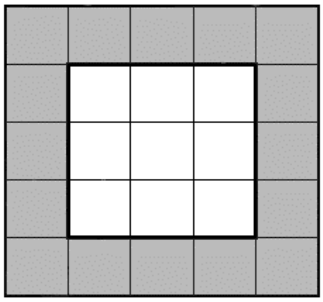


Fig.1 Boundary strip and interior domain

We adopt the Semi-Lagrangian method to solve the Two-Fluid model equation. Solving this equation involves computing three terms to update the velocity at each time interval: advection, diffusion and force application (Stam, 1999). The result generates a new velocity field with divergence-free velocity and is also a Poisson-pressure function. By using the

Helmholtz-Hodge Decomposition Theorem to define a project operator, we can solve the Poisson equation efficiently. Thus, the simulation algorithm can be expressed as:

$$S(\boldsymbol{u}) = P \circ F \circ D \circ A(\boldsymbol{u}), \tag{5}$$

where $\boldsymbol{u}$ is the velocity field of the primary fluid, P is project operator, D is a diffusion operator, F is the interaction force, and A is advection operator. The detailed information on Semi-Lagrangian method can be found in reference. Every step is implemented on GPU. Because the frame buffer is limited to two dimensions, we decompose the 3D cells into a stack of 2D slabs. A slab operation consists of one or more fragments in the frame buffer, usually with an active fragment program followed by a texture update. Each fragment processing is driven by rendering geometric primitives. We take the computation of the advection term for example. The advection operator can be expressed as follows:

$$A(x, t+\delta t)=A(x-\boldsymbol{u}(x, t)\delta t, t) \tag{6}$$

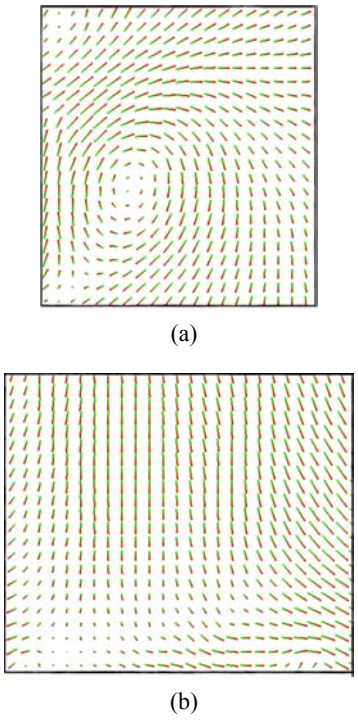
where x is an arbitrary 3D position, t is an arbitrary time, δt is the time interval, $\boldsymbol{u}(x,t)$ is the velocity of position x at time t. To compensate for the characteristics smoothing of the small-scale details due to numerical dissipation of Semi-Lagrangian method, we introduce the confinement force $f_{\text{vorticity}}$.

$$f_{\text{vorticity}}=\varepsilon \cdot h \cdot (\boldsymbol{N}\times\omega),$$

where $\omega=\nabla\times\boldsymbol{u}$ is the curl of the velocity field,

$$N=\eta/|\eta|, (\eta=\nabla|\omega|), \tag{7}$$

ε is used to control the amount of small-scale details added back into the whole velocity field, and it is positive. h is the spatial step which guarantees that even if the mesh is redefined the physically accurate solution can still be obtained. This technique was invented by Steinhoff and Underhil (1994), and has been used successfully as an engineering model for very complex flow fields. In 2001, Fedkiw *et al.*(2001) first applied it to the field of computer graphics, and used it to simulate smoke scene. Fig.2 shows a horizontal section and a vertical section of the velocity field modeled by the above method. In Fig.2, a velocity vector is expressed as a line segment. We can see that a tornado's horizontal velocity field is rotating and that its vertical velocity field is uplifting. The results accord with fact.



(a)

(b)

Fig.2 Horizontal section (a) and vertical section (b) of the velocity field

Implementation on GPU

As mentioned before, we perform all the numerical computation for the Two-Fluid model in the programmable, floating point fragment unit of a graphics processor. State fields, such as $\boldsymbol{u}$ and p are stored in textures. We conduct computations using fragment programs written in Cg shading language (Mark *et al*., 2003). The fragment programs implement the steps described in the previous section using texture operations to read data from the grids.

Previous methods for 3D simulation on GPUs use 3D textures or a stack of 2D textures to represent the cells. To apply a simulation operation to the cells, for example, to compute the interaction force, the volume must be updated slice by slice. At each slice, the operation is applied, and the texture copy or a context switch associates with rendering to texture. We instead represent our cells using what is called a

flat 3D texture. A flat 3D texture represents actually a 3D volume (Dong *et al*., 2004), as shown in Fig.3. Flat 3D textures can be updated in a single rendering pass. This means that a 3D simulation can be implemented in the same number of passes as that required by an equivalent 2D simulation. So, it provides a quick inexpensive way to preview the results of a 3D simulation (Harris *et al*., 2003).

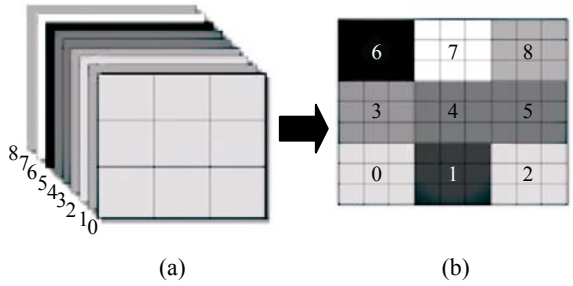


Fig.3 3D texture (a) and its corresponding flat 3D texture (b)

SHADING OF THE TORNADO SCENE

Most previous works on the simulation of tornado did not take light scattering and absorption of the participating media into account, and adopted gray color as the texture. Thus, the simulated results lacked reality. In fact, the shading effect of the tornado scene due to scattering and absorption of light by the participating media is obvious. We simulate the shading effect based on Multiple Mie Scattering Theory (van de Hulst, 1982).

Light scattering illumination

Scattering models simulate the emission and absorption of light by a medium as well as scattering through the medium. Single scattering models can simulate scattering of light by the medium in a single direction. This direction usually leads to the point of view. Multiple scattering models are more physically accurate, and account for scattering in all directions or a sample of directions, and therefore are much more complicated and expensive to evaluate.

In a multiple scattering simulation that samples N directions on the sphere, each additional order of scattering should multiply the number of simulated paths by N. Fortunately, as demonstrated by Nishita *et al*.(1996), the contribution of most of these paths is insignificant. Nishita *et al*.(1996) found that scattering illumination is dominated by the first and second orders, and therefore they simulated up to the fourth order. They reduced the directions sampled when estimating scattering to sub-spaces of high contribution, which are composed mostly of directions near the direction of forward scattering and those directed at the viewer. We approximate multiple scattering only in the light direction (multiple forward scattering) and anisotropic single scattering in the eye direction (Harris and Lastra, 2001).

Our rendering method is a two-pass algorithm. The first pass computes the amount of incident light at each position P from direction ω. This light consists of all direct light in direction l that is not absorbed by the participating media plus light scattered to P from other particles. This multiple scattering model is written as:

$$\begin{aligned} I(P,\omega) &= I_0(\omega)\mathrm{e}^{-\int_0^{D_P}\tau(t)\mathrm{d}t} + \int_0^{D_P} g(s,\omega)\mathrm{e}^{-\int_0^{D_P}\tau(t)\mathrm{d}t}\mathrm{d}s, \\ g(x,\omega) &= \int_{4\pi} r(x,\omega,\omega')I(x,\omega')\mathrm{d}\omega', \end{aligned} \tag{8}$$

where $I_0(\omega)$ is the intensity of light in direction ω outside the tornado scene, $\tau(t)$ is the extinction coefficient at depth t, D_P is the depth of P along the light direction, $g(x,\omega)$ represents the light coming from other directions ω' and scattering into direction ω by participating media at the point x, $r(x,\omega,\omega')$ is the bi-directional scattering distribution function determining the percentage of light incident on x from direction ω' then scattering in direction ω. It can be expressed as:

$$r(x,\omega,\omega')=a(x)\cdot\tau(x)\cdot p(\omega,\omega'), \tag{9}$$

here $a(x)$ is the albedo of the medium at x, and $p(\omega,\omega')$ is the phase function which will be discussed in Section 4.3. In our simplification of multiple forward scattering, $\omega=l$ and $\omega'=-l$. That means we approximate the integration of Eq.(8) over only a small solid angle γ around the forward direction. Because this solid angle is small, and as r and l are assumed to be constant over γ, Eq.(8) is reduced to:

$$g(x,l)=r(x,l,-l)\cdot l(x,-l)\cdot\gamma/4\pi. \quad (10)$$

We split the light path from 0 to D_P into a number of segments S_j, for j from 1 to N. By approximating the integrals with Riemann Sums, we have

$$I_P = I_0 \cdot \prod_{j=1}^{N} \mathrm{e}^{-\tau_j} + \sum_{j=1}^{N} g_k \prod_{k=j+1}^{N} \mathrm{e}^{-\tau_k}, \quad (11)$$

where I_0 is the intensity of light incident on the edge of the tornado, g_k is the discrete form of $g(x,\omega)$ and $g_k=a_k\cdot\tau_k\cdot p(l,-l)\cdot I_k\cdot\gamma/4\pi$. In order to transform Eq.(9) into an algorithm that can be implemented in graphics hardware, we cast it as a recurrence relation:

$$I_k = \begin{cases} g_{k-1} + T_{k-1}\cdot I_{k-1}, & 2 \le k \le N, \\ I_0, & k = 1, \end{cases} \quad (12)$$

where I_k is the intensity emitted from position P_k, $T_{k-1} = \mathrm{e}^{-\tau_{k-1}}$ is the transparency coefficient.

Eye scattering

In addition to simulating multiple forward scattering, we also implement single scattering toward the viewer. The recurrence for this is slightly different:

$$E_k=S_k+T_k\cdot E_{k-1},\ 1\le k\le N,$$

which indicates that the light E_k, exiting from any position, is equal to the light incident on it $T_k\cdot E_{k-1}$ plus the light that it scatters, S_k.

$$S_k=a_k\cdot\tau_k\cdot p(\omega,-l)\cdot I_k/4\pi,$$

where ω is the view direction and T_k is as above.

Phase function

The phase function $p(\omega,\omega')$ is a very important factor for the shading of the tornado scene. Here we use the Henyey-Greenstein function as the phase function, for its simplicity and good control of results (Nishita *et al*., 1993). It can be expressed as:

$$F(\theta,g) = \frac{3(1-g^2)}{2(2+g^2)}\cdot\frac{(1+\cos^2\theta)}{(1+g^2-2g\cos\theta)^{3/2}}, \quad (13)$$

where θ is the scattering angle, g is asymmetry factor given by

$$g = \frac{5}{9}u - \left(\frac{4}{3} - \frac{25}{81}u^2\right)x^{-1/3} + x^{1/3},$$
$$x = \frac{5}{9}u + \frac{125}{729}u^3 + \left(\frac{64}{27} - \frac{325}{243}u^2 + \frac{1250}{2187}u^4\right)^{1/2}, \quad (14)$$

here, u is determined by the size of the scattering particle and the wavelength of the incident light. Its value varies from 0.7 to 0.85.

Fig.4 shows a part of the illumination texture we obtained with the above method. We sampled 128 vertical sections of the whole space. Sixteen samples' illumination textures are selected and we put them together in Fig.4. We can see that the illumination texture of each sample is different due to its relative position to the light source and the viewpoint. We get the final illumination texture of a tornado by blending all the sample sections' illumination texture.

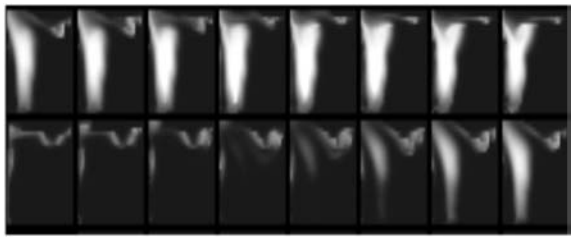


Fig.4 Illumination texture

RESULTS

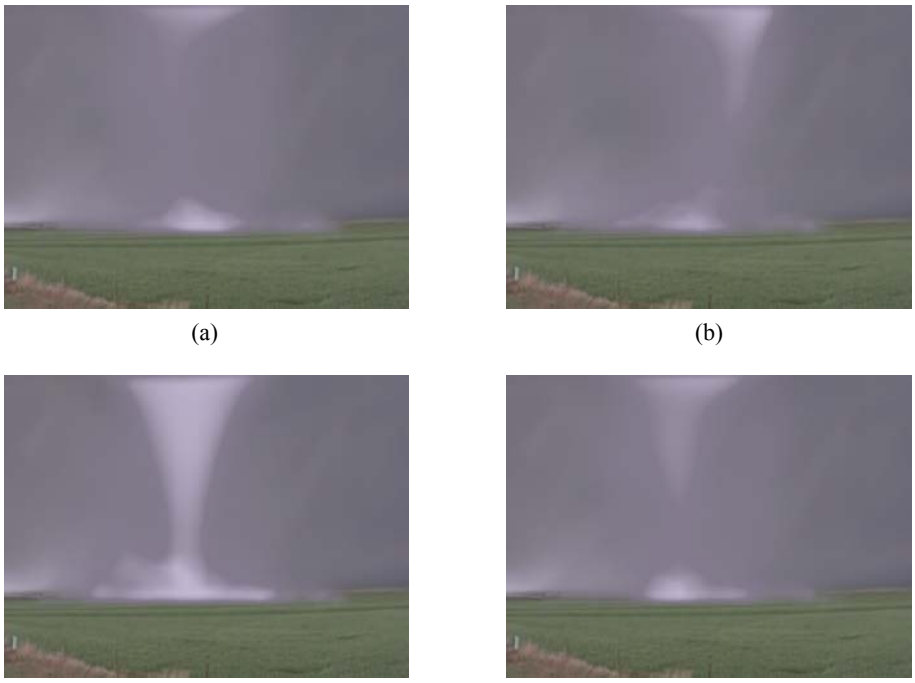
With the proposed methods, we successfully generated various kinds of dynamic tornado scenes realistically on a PC with 2.8 GHZ, Pentium IV processor, 1 GB memory and an nVidia GeForce FX 6800GT graphics card at State Key Laboratory of CAD & CG, Zhejiang University. All images are sized 640 by 480 pixels. The rendering rate is about 15 frames per second. Fig.5 shows a moving tornado scene at a distance. From Figs.5a and 5b, we can see that the tornado color and shape change when the tornado moves from left to right. Some debris on the ground is also rolled up by the tornado. Fig.6 shows a near tornado scene. We can see the tornado's details

(a) (b)

Fig.5 Moving tornado scene at a distance

(a) (b)

Fig.6 A near tornado scene



(a) (b)

(c) (d)

Fig.7 Different stages of a tornado during its life

more clearly from Figs.6a to 6b. Its color is very dark as its components are mainly dust, sand, etc. Fig.7 (see page 1105) shows the different stages of a tornado during its life. From Figs.7a to 7b, we can see the forming stage of the tornado scene. The tornado is launching from the cloud above, and the debris on the ground is rolled up more and more. Fig.7c shows the mature stage of the tornado scene. The decaying stage is shown in Fig.7d.

CONCLUSION AND FUTURE WORK

We have presented a physically based simulation method of various tornado scenes in this paper. We first propose a Two-Fluid model to simulate the tornado flow movement and its interaction with surrounding debris. Then, we develop an efficient method to calculate the light scattering and absorption in the participating media, thus making the whole scene more realistic. By using techniques of graphics hardware, interactive rendering rates are achieved.

Future work includes simulation of other natural catastrophic phenomena such as sand storm, debris flow, etc.

References

Ding, X., 2004. Physically Based Simulation of Tornadoes. Master Thesis. Waterloo University.

Dong, Z., Chen, W., Bao, H., Zhang, H.X., Peng, Q.S., 2004. Real-time Voxelization for Complex Polygonal Models. Proceedings of Pacific Graphics 2004, p.73-78.

Fedkiw, R., Stam, J., Jensen, H.W., 2001. Visual Simulation of Smoke. Proceedings of SIGGRAPH'01, p.15-22.

Flora, S., 1953. Tornadoes of the United States. University of Oklahoma Press, Norman, Oklahoma, USA.

Foster, N., Fedkiw, R., 2001. Practical Animation of Liquids. Proceedings of SIGGRAPH'01, p.23-30.

Harris, M.J., Lastra A., 2001. Real-time Cloud Rendering. Proceedings of Eurographics 2001, p.76-84.

Harris, M.J., Baxter, W.V., Scheuermann, T., Lastra, A., 2003. Simulation of Cloud Dynamics on Graphics Hardware. Proceedings of the ACM SIGGRPAH/EUROGRAPHICS Conference on Graphics Hardware, p.92-101.

Lewellen, W.S., 1993. Tornado Vortex Theory. Proceedings of the Tornado: its Structure, Dynamics, Prediction, and Hazards (Geophysical Monograph 79), p.19-39.

Lewellen, W.S., Lewellen, D.C., 1997. Large-Eddy simulation of a tornado's interaction with the surface. *Journal of the Atmospheric Sciences*, **54**(5):581-605. [doi:10.1175/1520-0469(1997)054<0581:LESOAT>2.0.CO;2]

Marble, F.E., 1970. Dynamics of dusty gases. *Annual Review of Fluid Mechanics*, **2**(1):397-446. [doi:10.1146/annurev.fl.02.010170.002145]

Mark, W.R., Glanvlle, R.S., Akeley, K., Kilgard, M.J., 2003. Cg: A System for Programming Graphics Hardware in a C-like Language. Computer Graphics (Proceedings of SIGGRAPH'03), ACM Press.

Nguyen, D., Enright, D., Fedkiw, R., 2003. Simulation and Animation of Fire and Other Natural Phenomena in the Visual Effects Industry. Western States Section, Combustion Institute, Fall Meeting, UCLA.

Nishita, T., Sirai, T., Tadamura, K., Nakamae, E., 1993. Display of the Earth Taking into Account Atmospheric Scattering. Proceedings of SIGGRAPH'93, p.175-182.

Nishita, T., Dobashi, Y., Nakamae, E., 1996. Display of Clouds Taking into Account Multiple Anisotropic Scattering and Sky Light. Proceedings of SIGGRAPH'96, p.379-386.

Nolan, D.S., Farrell, B.F., 1999. The structure and dynamics of tornao-like vortices. *Journal of the Atmospheric Sciences*, **56**(16):2908-2935. [doi:10.1175/1520-0469(1999)056<2908:TSADOT>2.0.CO;2]

Stam, J., 1999. Stable Fluids. Proceedings of SIGGRAPH'99, p.121-128.

Steinhoff, J., Underhil, D., 1994. Modification of the euler equation for "vorticity confinement": application to the computation of interacting vortex rings. *Physics of Fluids*, **6**(8):2738-2744. [doi:10.1063/1.868164]

Trapp, R.J., Fiedler, B.H., 1993. Numerical Simulation of Tornado-like Vortices in Asymmetric Flow. *In*: Church, C.R. (Ed.), The Tornado: Its Structure, Dynamics, Prediction, and Hazards. AGU Geophysical Monograph 79, Washington, p.49-54.

Weiskopf, D., Schafhitzel, T., Ertl, T., 2005. Real-time Advection and Volumetric Illumination for the Visualization of 3D Unsteady Flow. Proceedings of Eurovis (EG/IEEE TCVG Symp. Vis.), p.13-20.

Xia, J., Lewellen, W.S., Lewellen, D.C., 2003. Influence of mach number on tornado tornado flow dynamics. *Journal of the Atmospheric Sciences*, **60**(22):2820-2825. [doi:10.1175/1520-0469(2003)060<2820:IOMNOT>2.0.CO;2]

Xue, D., Crawfis, R., 2004. Fast Dynamic Flow Volume Rendering Using Textured Splats on Modern Graphics Hardware. Proceedings of SPIE IS&T Electronic Imaging. *SPIE*, **5295**:133-140. [doi:10.1117/12.539248]

van de Hulst, H.C., 1982. Light Scattering by Small Particles. Dover Publication Inc., New York.