A Fast Rendering Method of Clouds Using Shadow-View Slices

Ryo Miyazaki* Yoshinori Dobashi** Tomoyuki Nishita*
*University of Tokyo **Hokkaido University
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-0033, Japan Kita-ku, Kita 13, Nishi 8 Sapporo, Japan
{ryomiya, nis}@nis-lab.is.s.u-tokyo.ac.jp doba@nis-ei.eng.hokudai.ac.jp

ABSTRACT
Clouds make an important contribution when composing images for various applications such as virtual outdoor scenes. An efficient rendering method is proposed for the rendering of clouds suitable for graphics hardware. Most of the calculations for rendering clouds are processed by texture operations in the graphics hardware. The method creates images of clouds processing photo realism by taking into account the single scattering of light, shadows on the ground, and shafts of light through the clouds. It is necessary to consider the light attenuation in the sun direction and in the view direction to render the volume data of the cloud density. These attenuations are calculated in a single step, using a sectional technique called "shadow-view slice".

KEY WORDS
Clouds, Natural Phenomena, Atmospheric Effects, Volume Rendering, Graphics Hardware.

1. Introduction

Clouds play an important role when constructing images for flight simulators, computer games, or outdoor scenes. Their colors and shapes change both in time and on the relative positions of the sun and the observer. The density distribution of clouds must be defined in three-dimensional space in order to create realistic images and animations, that for example, are obtained in fluid simulation [20].

This paper proposes a new method of cloud rendering that can rapidly create realistic images, preferably in real-time. The method is suitable for animation of cumulus-like clouds, that are deep and extend over the whole sky in a landscape. The method calculates cloud color, taking into account the single scattering of the light, the shadows on the ground and the shafts of light through clouds by making most of graphics hardware capabilities. The method is well suited to graphics hardware, especially the pixel shader, since most of the process of rendering is in the form of a texture operation in the graphics hardware. The pixel shader is a programmable shader, which processes pixel values.

It is necessary to consider the light attenuations in the sun direction (from the sun to cloud) as well as the view direction (from cloud to the viewpoint) for cloud rendering. The method calculates these attenuations by using surface slices facing mid-way between the sun direction and the view direction. These slices are termed "shadow-view slices" (SVSs). Slices of the volume data of the cloud density are used to calculate the accumulated attenuation ratio in the sun direction. Additionally, the density of atmospheric particles depends on their height above the ground. Thus the atmospheric density and the accumulated attenuation ratio are stored as a single texture in a pre-process, termed the "atmosphere texture". The atmosphere texture is mapped onto the SVS. The intensity of light scattered in the SVS is then calculated. The final image is created by blending a number of SVSs in order to compute the total intensity of light scattered towards the viewpoint.

In the following sections, Section 2 discusses previous work related to cloud rendering. Section 3 describes overview of the proposed method. Section 4 explains this cloud rendering method using the shadow-view slices. In Section 5 the proposed method is applied to creating images of various scenes of clouds. Finally, Section 6 discusses the conclusion.

2. Previous Work

In order to display photo-realistic images of clouds, it is desirable to a physical model, taking into account scattering of light due to particles. Many such methods have therefore been developed [4,7,8,13,14,19,24,26]. There are also a number of methods for generating shafts of light [11,12,17,18,22,23,25]. Most of these use a ray-tracing algorithm or a scan-line algorithm. However these methods require a few up to tens of minutes.

Dobashi et al. developed a fast method for rendering realistic clouds with shafts of light using graphics hardware [5]. In this method, clouds are rendered by using a splatting method. To render shafts of light, they used multi spherical shells. Their method calculates cloud color by using metaballs. As rendering of clouds and shafts of light are different processes, shafts of light are rendered only below the base of clouds. But the proposed method calculates the intensity of scattered light due to cloud and atmospheric particles simultaneously in a single process. Dobashi's method computes light scattering point by point, whereas the proposed method achieves this surface by surface. Thus the proposed method is more suited for hardware-acceleration since the main process of the rendering method is treated as a texture calculation. Dobashi et al. proposed a rendering method for atmospheric scattering effects at an interactive rate using graphics hardware [6]. In their method, the simulation space is divided into sampling planes. This method requires two passes for computing attenuations in the sun and the view directions.

A variety of other hardware-accelerated methods for volume rendering have been developed [1,3,16,21,27]. Many of them use hardware texture mapping functions to render volumes defined in voxels. However, when these methods are applied directly to the rendering of clouds and atmosphere, a huge volume data is required. The proposed method uses volume texture for clouds, but uses 2D texture for atmosphere, since the density of atmosphere depends on the height above the ground. The intensity of light scattered is efficiently computed by using both the volume texture for clouds and the 2D texture for the atmosphere.

Harris and Lastra developed real-time cloud rendering method [9]. However this method assumes that cloud is static. Therefore, Harris et al. developed a fast rendering method for dynamic clouds. This method computes both the motion and the images of clouds on the GPU using 3D textures [10]. The rendering algorithm of this method requires two passes, and the method does not consider the light scattered due to the atmospheric particles.

It is necessary to consider the light attenuation in both the sun direction and the view direction in order to render the volume data of the cloud density. Most of volume rendering methods calculate these attenuations using separate processes, computing the attenuations in two passes. The first pass computes the attenuation in the sun direction, whilst the second pass is for the attenuation in the view direction. In this case, temporary volume data are required for the accumulated attenuation ratio in the sun direction. The proposed method calculates these attenuations by using the slices facing a direction mid-way between the sun direction and the view directions. This approach does not need the temporary volume data for the accumulated attenuation ratio in the sun direction and can render images faster than the two pass algorithms. Zhang et al. and Kniss have proposed rendering methods for volume data by using same slice for these attenuations [15,28,29]. Zhang's method uses a splatting algorithm and Kniss uses a 3D texture renderer. But these methods did not treat clouds scene. For rendering clouds, atmospheric effects must be considered. The proposed method calculates not only intensity of cloud including Mie scattering effect but also corrects sampling interval by using graphics hardware.

3. Overview of the Proposed Method

Clouds consist of particles, such as water droplets, whose density distribution is non-uniform. The sizes of the particles are relatively large, so that the particles have strong forward scattering, known as Mie scattering. The contribution of Rayleigh scattering is trivial in clouds. Multiple scattering is important for scattering in the clouds but their computational cost is very high. The proposed method approximates the effect of the multiple scatterings as a constant term. For shafts of light, atmospheric particles are also taken into account. In the earth's atmosphere, the density of particles decreases exponentially with their height above the ground. In this paper, Mie scattering due to the atmospheric particles is also considered.

As described previously, it is necessary to consider the light attenuations in both the sun direction (from the sun to cloud) and the view direction (from cloud to the viewpoint). These attenuations are calculated by using the shadow-view slice (SVS) facing mid-way between the sun and the view directions. As shown in Fig. 1, the simulation space is divided by a number of SVSs. For calculating the intensity of light scattering on a SVS, the density of cloud and atmospheric particles are required, together with the light attenuation ratio from the sun to the SVS. This attenuation ratio is denoted as the "accumulated attenuation ratio in the sun direction". This is obtained by multiplying the attenuation ratio due to the clouds particles with that due to the atmospheric particles.

The density of a slice of the volume data is projected onto a surface for a rendering target called a "shadow surface". The accumulated attenuation ratio due to the cloud in the SVS is then calculated and stored. The shadow surface is set vertical to the sun direction. Shadow surface is also used to render shadows on the ground of the final image. On the other hand, the atmospheric density can be calculated analytically, depending on the height above the ground. And the accumulated attenuation ratio in the sun direction due to the atmospheric particles can be computed numerically by the atmospheric density. In this case, these are stored as a single texture called "atmosphere texture" in a pre-process, which is then mapped onto the SVS as shown in Fig.1. The intensity of light scattered on a SVS is calculated on the screen by combining the cloud density of the volume texture, the accumulated attenuation ratio due to the cloud of the shadow surface, together with the density of atmospheric particle and the accumulated attenuation ratio due to the atmospheric particles of the atmosphere texture. The final image is created by blending a number of SVSs on the viewing ray to compute the total intensity of light scattered towards the viewpoint on the screen.

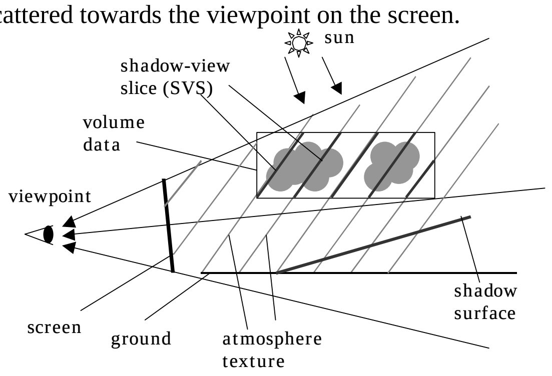


Fig 1: Overview of the proposed method

When sampling with parallel planes, the sampling interval and the phase angle between the sun direction and the direction on the viewing ray are different on each viewing ray. The proposed method utilizes the texture operations for correcting the difference of sampling interval and calculating Mie scattering on the viewing ray in each view direction. The sampling intervals and the values of Mie scattering are stored as textures referred to as "sampling interval texture" and "Mie scattering texture", respectively. These textures are combined with the volume texture of the cloud density, the atmosphere texture and the shadow surface when calculating the scattered light on the SVS. Eventually the scattered light on a SVS is calculated by five textures.

4. Rendering of Cloud Scenes

4.1. Intensity Calculation of Clouds

As shown in Fig. 2, denoting a point on the viewing ray $\mathbf{P}$, the following equation gives the intensity, I_v, of the light reaching the viewpoint.

$$I_v = I_{back} g_{view}(T) + I_{sun}\int_0^T ((\beta_{cl}(\alpha) + k_{multi})\rho_{cl}(t) + \beta_{atm}(\alpha)\rho_{atm}(h)) g_{view}(t) g_{sun}(s)dt, \quad (1)$$

where I_{back} is the intensity of the background on the viewing direction. g_{view} is the attenuation ratio from $\mathbf{P}$ toward the viewpoint. g_{sun} is the attenuation ratio from the top of space toward $\mathbf{P}$. T is the distance between the background and the viewpoint. I_{sun} is the intensity of the sunlight. β_{cl} and β_{atm} are phase functions of Mie scattering due to cloud and atmosphere respectively. k_{multi} is a constant coefficient for approximating the multi scattering in cloud. α is the phase angle between the sun direction and the direction on the viewing ray. t is the distance between the viewpoint and point $\mathbf{P}$ on the viewing ray. ρ_{cl} is the cloud density at point $\mathbf{P}$. ρ_{atm} is the atmospheric density at point $\mathbf{P}$. h is the height of point $\mathbf{P}$ from the ground. s is the distance between point $\mathbf{P}$ and the top of atmosphere. s depends on h and the position of the sun. For the phase function β, we use the Heney-Greenstein function described by Cornett [2]. g_{view}, g_{sun}, and β are given by the following equations.

$$g_{view}(t) = \exp(-\tau(t)),\ g_{sun}(s) = \exp(-\tau(s)), \quad (2)$$

$$\beta(\alpha) = \frac{3(1-g^2)}{2(2+g^2)}\frac{(1+\cos^2\alpha)}{(1+g^2-2g\cos\alpha)^{3/2}}, \quad (3)$$

where τ is the optical length, and g is the coefficient that depends on the state of particles and the wave length of light. A numerical integration method is often used to calculate Eq. 1 since it is difficult to obtain an analytical solution. I_v is calculated numerically by the integral of Eq. 1, using n SVSs as follows.

$$I_v = I_{back} g_{view}(T) + \sum_{i=1}^{n} \Delta I_v(t_i), \quad (4)$$

$$\begin{aligned}\Delta I_v(t_i) &= \Delta t(\mathbf{p}) I_{sun} \times \\ &((\beta_{cl}(\alpha(\mathbf{p})) + k_{multi})\rho_{cl}(t_j) + \beta_{atm}(\alpha(\mathbf{p}))\rho_{atm}(h_i)) \times \\ &g_{view}(t_i) g_{sun}(s_i),\end{aligned} \quad (5)$$

$$\begin{aligned}g_{view}(t_i) &= g_{view\_atm}(t_i) g_{view\_cl}(t_i) \\ &= \prod_{j=1}^{i} \Delta g_{view\_atm}(t_j)\Delta g_{view\_cl}(t_j) \\ &= \prod_{j=1}^{i} \exp((-k_{atm}\rho_{atm}(h_j) - k_{cl}\rho_{cl}(\mathbf{P}_j))\Delta t(\mathbf{p})),\end{aligned} \quad (6)$$

$$\begin{aligned}g_{sun}(s_i) &= g_{sun\_atm}(s_i) g_{sun\_cl}(s_i) \\ &= g_{sun\_atm}(s_i)\prod_{j=1}^{i} \Delta g_{sun\_cl}(s_j) \\ &= g_{sun\_atm}(s_i)\prod_{j=1}^{i} \exp(-k_{cl}\rho_{cl}(\mathbf{P}_j)\Delta s),\end{aligned} \quad (7)$$

g_{view_atm} and g_{view_cl} are the attenuation ratios in the view direction due to atmosphere and cloud, respectively, g_{sun_atm} and g_{sun_cl} are the attenuation ratios in the sun direction due to atmosphere and cloud, respectively. k_{atm} and k_{cl} are coefficients of attenuation of atmosphere and cloud, respectively. ρ_{atm} and g_{sun_atm} are obtained from the atmosphere texture. The atmosphere texture is recreated when the height of the sun changes. Δs is the sampling interval in the sun direction. Δs is uniform in the all pixels in the shadow surface. Δt is the sampling interval in the view direction. Δt is different in each pixel $\mathbf{p}$ on the screen. The phase angle, α, depends on $\mathbf{p}$ on the screen and the sun position. Δt, β_{cl} and β_{atm}, depending on α, are stored as "sampling interval texture" and "Mie scattering texture", respectively (see 4.4).

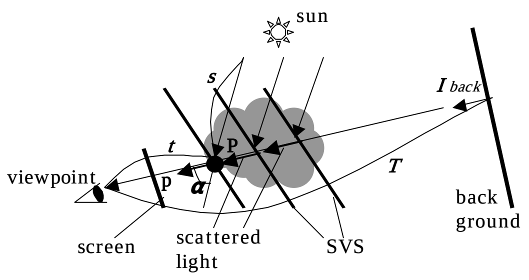


Fig. 2: Intensity calculation of clouds

Here, we get equations from Eqs. 5, 6, and 7.

$$\Delta I_{SVS} = \Delta t(\mathbf{p})\ ((\beta_{cl}(\mathbf{p}) + k_{multi})\rho_{cl}(\mathbf{P}_i) + \beta_{atm}(\mathbf{p})\rho_{atm}(h_i)) g_{sun}(s_i), \quad (8)$$

$$\Delta g_{view}(t_i) = \exp((-k_{atm}\rho_{atm}(\mathbf{P}_i) - k_{cl}\rho_{cl}(h_i))\Delta t(\mathbf{p})), \quad (9)$$

$$\Delta g_{sun\_cl}(s_i) = \exp(-k_{cl}\rho_{cl}(\mathbf{P}_i)\Delta s), \quad (10)$$

where, ΔI_{SVS} is a part of ΔI_v that satisfies $\Delta I_v(t_i) = I_{sun}\,\Delta I_{SVS}(t_i) g_{view}(t_i)$. ΔI_{SVS} and Δg_{view} are calculated by a SVS on the screen. First, in order to calculate ΔI_{SVS} and Δg_{view}, Δg_{sun_cl} is calculated by Eq.10 in the shadow surface. ρ_{cl} is obtained from the volume texture. ρ_{cl} in the SVS is projected onto the shadow surface, and $g_{sun\_cl}(s_i)$ is obtained by multiplying $\Delta g_{sun\_cl}(s_i)$ by $g_{sun\_cl}(s_{i-1})$ on each pixel in the shadow surface. The shadow surface $g_{sun\_cl}(s_0)$ has been initialized to 1.0. In the next stage, ΔI_{SVS} and Δg_{view} are calculated on the screen by Eqs. 8 and 9 using five components: the volume texture (ρ_{cl}), the shadow surface (g_{sun_cl}), the atmosphere texture (ρ_{atm} and g_{sun_atm}), the sampling interval texture (Δt), and the Mie scattering texture (β_{cl} and β_{atm}). The calculation of Eqs. 8, 9, and 10 are a main part of rendering. These can be processed by texture operations using the pixel shader. The color of final image is calculated by using ΔI_{SVS} and Δg_{view} of each SVS on the screen (see 4.3). Fig. 3 shows flow chart of the proposed method.

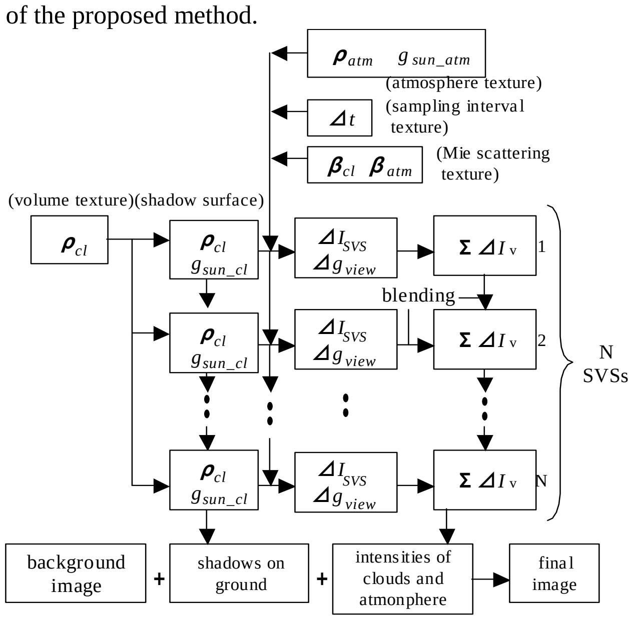


Fig. 3: Flow chart of the proposed method

4.2. Shadow-View Slice

The proposed method calculates the accumulated attenuation ratio due to the cloud in the sun direction on the shadow surface and the intensity of light scattered toward the viewpoint on the screen by the SVS, respectively. Temporary volume data is not required for the accumulated attenuation ratio in the sun direction. SVS is not necessarily vertical to the sun direction or the view direction. In the method, the order of projecting the slice onto the screen changes, depending on the positions of the sun and the viewpoint. That is, the order is changed depending on whether the angle between the view direction and the sun direction is obtuse or acute (see Fig. 4). In Fig.4, $\mathbf{V}_e$ is the vector in the view direction, $\mathbf{V}_s$ is the vector in the sun direction. In case of Fig.4 (a) ($\mathbf{V}_e \cdot \mathbf{V}_s \geq 0$), back-to-front traversal is employed, and in the case of Fig.4 (b) ($\mathbf{V}_e \cdot \mathbf{V}_s < 0$), front-to-back traversal is used. In the both cases, the attenuation ratio in the sun direction is calculated in an ascendant order, depending on the distance from the sun. The shadow surface is set vertical to the sun direction. The direction normal to SVS is defined to be $\theta_e : \theta_s = 1 : a$, where θ_e, θ_s are the angles between the normal to the SVS and $\mathbf{V}_e, \mathbf{V}_s$ respectively, a is a parameter to define the direction normal to the SVS. a is assigned values of $1.0 \sim 2.0$ to reduce the difference of the sampling interval of each of the pixels of the screen. The sampling interval in the sun direction is the same in each pixel of the shadow surface because of the parallel projection. However, the sampling interval in the view direction is different in each pixel on the screen, due to the perspective projection. Thus, the greater a is, the less difference of the sampling interval on each screen pixel is.

4.3. Blending method for calculation of scattered light

The color of the image is calculated using the following blending methods.

(a) back-to-front traversal (Fig. 4 (a))

$$S_i = C_i + \alpha_i S_{i-1}$$

(b) front-to-back traversal (Fig. 4 (b))

$$S_i = S_{i-1} + \alpha_i^* C_i, \qquad \alpha_i^* = \alpha_{i-1}\alpha_{i-1}^*$$

where C_i is color in i th SVS (i.e. $I_{sun}\,\Delta I_{SVS}$), α_i, attenuation ratio in the view direction in i th SVS (i.e. Δg_{view}), α_i^*, accumulated attenuation in i th SVS, and S_i, blended color from first SVS to i th SVS.

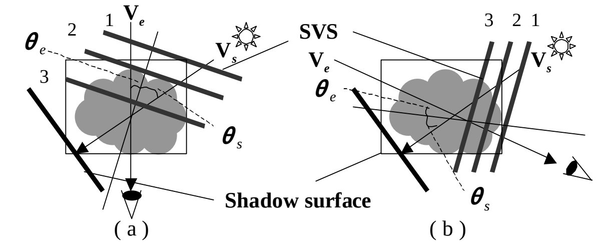


Fig. 4: Shadow-view slice (SVS) (a) $\mathbf{V}_e \cdot \mathbf{V}_s \geq 0$, (b) $\mathbf{V}_e \cdot \mathbf{V}_s < 0$, the numbers denote the traversal order of slices.

4.4. Correcting Sampling Interval and Mie Scattering Calculation

For the perspective projection onto the screen, the sampling interval and the phase angle for Mie scattering are different for each ray to the viewpoint (see Fig.5). These values are used for calculating the intensity of light scattered in Eq.5. Thus the sampling intervals and values of the phase function of Mie scattering are stored as "sampling interval texture" and "Mie scattering texture", respectively. The sizes of these textures are same as that of the screen. The scattered light to the viewpoint is calculated using these textures. These textures must be updated whenever the sun position or the viewpoint changes. Calculating the sampling intervals and the values of Mie scattering are also fast when using the pixel shader.

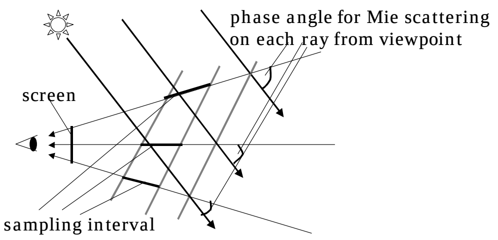


Fig. 5: Sampling intervals and phase angles of each ray to viewpoint

5. Results

Fig. 6 shows images generated by the proposed method. Fig. 6 (a) shows the example of cumulonimbus development. Fig. 6 (b) shows examples of the cumulus development process in the daytime. Shadows on the ground move with clouds. Fig. 6 (c) shows cumulus development in the evening. The shafts of light pass through the clouds. In Figs. (a), (b) and (c), the numbers of SVSs are 261, 353 and 366, respectively. These images are generated in 1.3 fps, 1.0 fps and 0.9 fps, respectively. The proposed method is several times faster than Dobashi's method [5]. In order to create volume data for these images, we use cumulus dynamics simulation methods [20]. The number of voxels are 128x128x128 for Fig.6 (a) and 256x256x64 for Figs.6 (b) and (c).

These images are rendered by a desktop PC (Pentium 4 2.7GHz, ATI FireGL X1 AGP Pro). We implemented the proposed method by the DirectX 9.0 and the pixel shader 2.0. Floating point textures are used for all 2D textures.

6. Conclusion

In this paper, an efficient rendering method for clouds is proposed. The space is divided into sampling planes called "shadow-view slices" (SVSs). The light attenuation in the sun direction and in the view direction is calculated in a single step using the SVS technique. Most of the calculations for rendering clouds are processed by texture operations in the graphics hardware. The method creates images of clouds processing photo realism by taking into account the single scattering of light, shadows on the ground, and shafts of light through the clouds.

References

[1] U. Behrens and R. Ratering, Adding Shadows to a Texture-based Volume Renderer, *1998 Proc. Symposium on Volume Visualization*, 1998, 39-46.

[2] W. M. Cornetto and J. G. Shanks, Physical Reasonable Analytic Expression for the Single-scattering Phase Function, *Applied Optics*, *31*(16), 1992, 3152-3160.

[3] T. J. Cullip, U. Neumann Accelerating Volume Reconstruction with 3D Texture Hardware, *Technical Report TR93-027*, University of North Carolina, Chapel Hill 1993.

[4] Y. Dobashi, T. Nishita, H. Yamashita, T. Okita, Using Metaballs to Modeling and Animate Clouds from Satellite Images, *The Visual Computer*, *15*(9), 1998, 471-482.

[5] Y. Dobashi, K. Kaneda, H. Yamashita, T. Okita, T. Nishita, A Simple, Efficient Method for Realistic Animation of Clouds, *Proc. SIGGRAPH2000*, 2000, 19-28.

[6] Y. Dobashi, T. Nishita, T. Yamamoto, Interactive Rendering of Atmospheric Scattering Effects Using Graphics Hardware, *Proc. Graphics Hardware 2002*, 2002, 99-108.

[7] D. S. Ebert, R. E. Parent, Rendering and Animation of Gaseous Phenomena by Combining Fast Volume and Scanline A-Buffer Techniques, *Computer Graphics*, *24*(4), 1990, 357-366.

[8] R. Fedkiw, J. Stam, H. W. Jensen, Visual Simulation of Smoke, *Proc. SIGGRAPH 2001*, 2002, 15–22.

[9] M. J. Harris, A. Lastra, Real-Time Cloud Rendering, *Computer Graphics Forum* (*Proc. EUROGRAPHICS 2001*), *20*(3), 2001, 76-84.

[10] M. J. Harris, W. V. Baxter III, T. Scheuermann, A. Lastra, Simulation of Cloud Dynamics on Graphics Hardware, *Proc Graphics Hardware 2003*, 92-101.

[11] M. Inakage, Volume Tracing of Atmospheric Environments, *The Visual Computer*, *7*(2-3), 1991, 104-113.

[12] H. W. Jansen, P. H. Christensen, Efficient Simulation of Light Transport in Scenes with Participating Media using Photon Maps, *Proc. SIGGRAPH'98*, 1998, 311-320.

[13] J. T. Kajiya, B. P. V. Herzen, Ray Tracing Volume Densities, *Computer Graphics*, *18*(3), 1984, 165-174.

[14] K. Kaneda, T. Okamoto, E. Nakamae, T. Nishita, Photorealistic Image Synthesis for Outdoor Scenery under Various Atmospheric Conditions, *The Visual Computer*, *7*(5&6), 1991, 247-258.

[15] J. Kniss, G. Kindlmann, C. Hansen, "Multi-Dimensional Transfer Function for Interactive Volume Rendering," *TVCG 2002*, *8*(3), 2002, 277-285.

[16] E. Lamar, B. Hamann, K. I. Joy, "Multiresolution Techniques for Interactive Texture-Based Volume Visualization," *Proc. Visualization'99*, 1999, 355–362.

[17] N. Max, "Light Diffusion through Clouds and Haze," *Graphics and Image Processing*, *13*(3), 1986, 280-292.

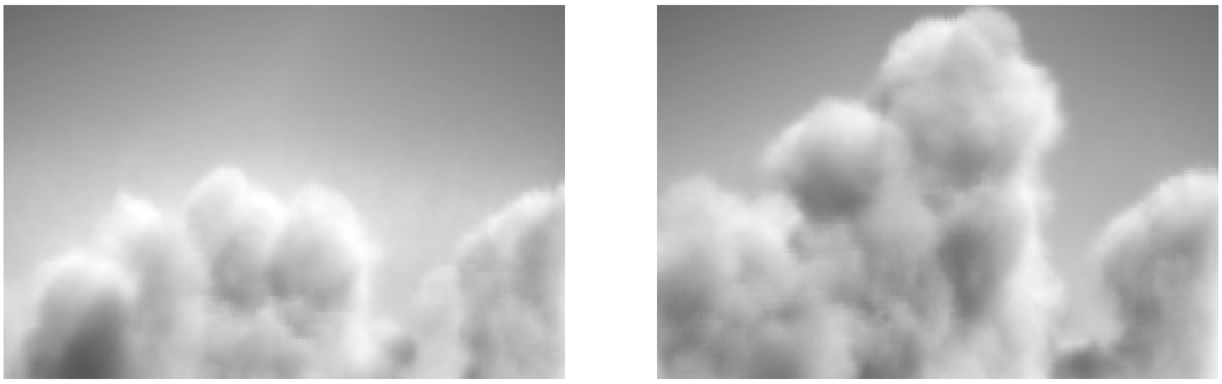
[18] N. Max, "Atmospheric Illumination and Shadows," *Computer Graphics*, *20*(4), 1986, 117-124.

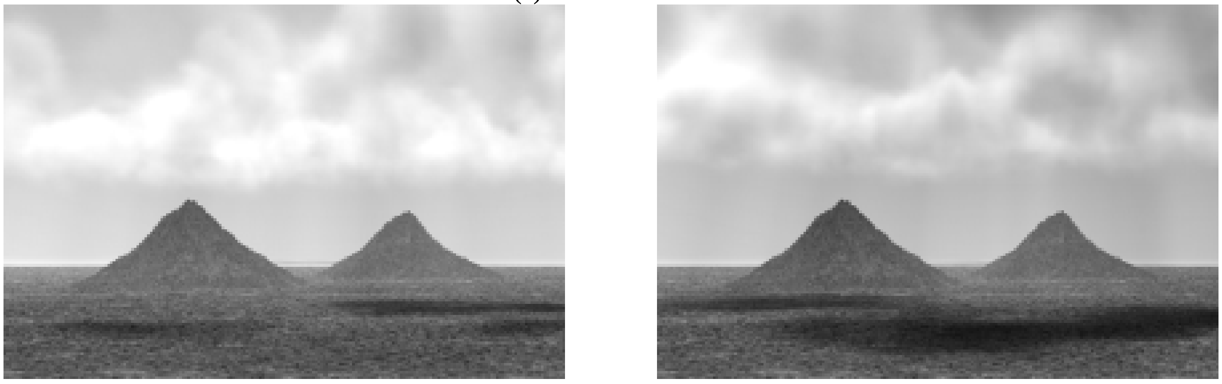
[19] N. Max, "Efficient Light Propagation for Multiple Anisotropic Volume Scattering," *Proc. the Fifth Eurographics Workshop on Rendering*, 1994, 87-104.

[20] R. Miyazaki, Y. Dobashi, and T. Nishita, Simulation of cumuliform clouds based on computational fluid dynamics, *Proc. EUROGRAPHICS 2002 Short Presentations*, 2002, 405-410.

[21] M. Nulkar, K. Mueller, Splatting with Sadows, *Proc. Volume Graphics 2001*, 2001, 35–49.

[22] T. Nishita, Y. Miyawaki, E. Nakamae, A Shading Model for Atmospheric Scattering Considering Distribution of Light Sources, *Computer Graphics*, *21*(4), 1987, 303-310.
[23] T. Nishita, E. Nakamae, Method of Displaying Optical Effects within Water using Accumulation Buffer, *Proc. SIGGRAPH'94*, 1994, 373-379.
[24] T. Nishita, Y. Dobashi, E. Nakamae, Display of Clouds Taking into Account Multiple Anisotropic Scattering and Sky Light, *Proc. SIGGRAPH'96*, 1996, 379-386.
[25] H. E. Rushmeier, K. E. Torrance, The Zonal Method for Calculating Light Intensities in The Presence of a Participating Medium, *Computer Graphics*, *21*(4), 1987, 293-302.
[26] G. Sakas, M. Gerth, Sampling and Anti-Aliasing of Discrete 3-D Volume Density Textures, *Graphics Forum* (*Proc. EUROGRAPHICS'91*), *10*, 1991, 87-102.
[27] R. Westermann, T. Ertl, Efficiently Using Graphics Hardware in Volume Rendering Applications, *Proc. SIGGRAPH'98*, 1998, 169–177.
[28] C. Zhang, R. Crawfis, Volumetric Shadows Using Splatting, *Proc. Visualization 2002*, 2002, 85-92.
[29] C. Zhang, R. Crawfis, Shadows and Soft Shadows with Participating Media Using Splatting, *IEEE, Transactions on Visualization and Computer Graphics*, *9*(2), 2003, 139-149.


(a) Cumulonimbus


(b) Cumulus in daytime

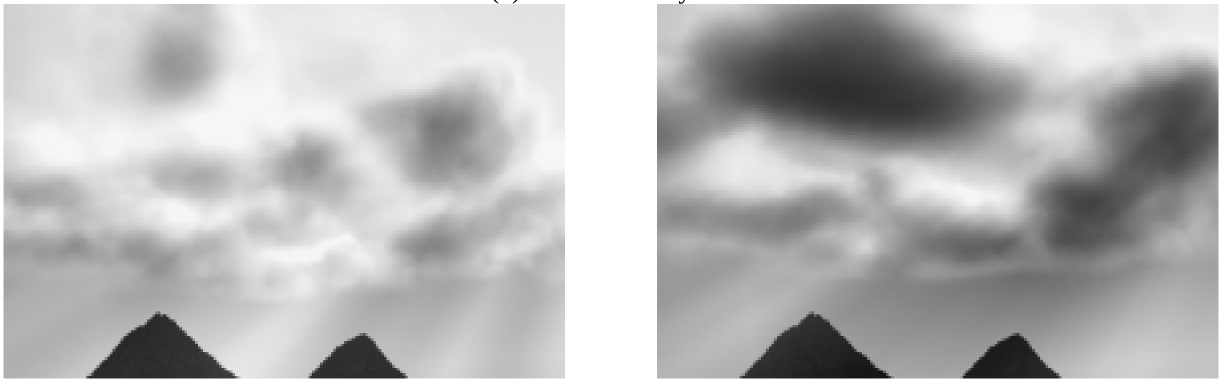

(c) Cumulus in evening

Fig.6: Result images