The Visual Computer

Using metaballs to modeling and animate clouds from satellite images

Yoshinori Dobashi[1],
Tomoyuki Nishita[2],
Hideo Yamashita[3],
Tsuyoshi Okita[1]

[1] Hiroshima City University, Faculty of Information Sciences, 3-4-1 Ozukahigashi, Asaminami-ku, Hiroshima, 731-3194, Japan
E-mail: {doba, okita}@im.hiroshima-cu.ac.jp
[2] The University of Tokyo, Department of Information Science, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan
E-mail: nis@is.s.u-tokyo.ac.jp
[3] Hiroshima University, Faculty of Engineering, 1-4-1 Kagamiyama, Higashi-Hiroshima, 739-8527, Japan
E-mail: yama@eml.hiroshima-u.ac.jp

We propose modeling that uses metaballs to create realistic clouds from satellite images. The method is intended for space flight simulators, visualization of the weather information, and simulation of surveys of the earth. The density distribution inside the clouds is defined by a set of metaballs. The parameters (center positions, radii, and density values) are automatically determined so that the synthesized image is similar to the satellite image. We also propose an animation method for clouds generated by a sequence of satellite images taken at a given interval. We give several examples of clouds generated from satellite images of typhoons passing through Japan.

Key words: Clouds – Image-based modeling – Satellite image – Metaballs

Correspondence to: Y. Dobashi

1 Introduction

The simulation of natural phenomena, such as water, smokes, fire, and clouds, has often been attempted in computer graphics (Klassen [12]; Kaneda et al. [11]; Stam and Fiume [23]; Max [15]; Stam and Fiume [24]; Nishita et al. [19]; Foster and Metaxas [5]). Clouds play an important role in the images of outdoor scenes, the earth viewed from outer space, and the visualization of weather information. This paper proposes a method for modeling large-scale clouds viewed from space, such as those in a typhoon. Many methods have been developed for modeling clouds, and they can be classified into two groups.

Methods in the first group create clouds by procedural modeling. Voss [26] and Musgrave [16] create realistic clouds with fractals. Gardner [6] has produced realistic images of clouds by using Fourier synthesis. However, this does not create a true three-dimensional geometric model. Ebert and Parent [3] have used solid texture to generate clouds in three-dimensional space. Ebert [2] has also developed a method combining metaballs and a noise function to model clouds. Sakas [20] has modeled clouds by using spectral synthesis, and Nishita et al. [19] create clouds by generating metaballs using the idea of fractals. Using these methods to create clouds that the user pictures in the mind is, however, very difficult since many parameters have to be specified by trial and error. Stam and Fiume [22] have developed a simple method for modeling clouds. In their method, a user specifies density values at several points in three-dimensional space. Then the density distribution of the clouds is obtained by interpolating the specified density values. Although this method can create realistic clouds, it is impractical for creating large-scale clouds viewed from space. Methods for modeling a set of clusters of clouds have also been developed. Ebert et al. [4] create realistic images of typhoons by the procedural approach. Nishita and Nakamae [19] have modeled clouds to generate realistic images of the earth viewed from space. In both these methods, however, clouds are simply modeled by applying two-dimensional fractals. The color and shape of clouds change depending on both the viewpoint and the position of the sun. These methods cannot simulate such effects.

Methods classified in the second group model clouds by simulating the physical process of the clouds forming (Kajiya and Herzen [9]; Stam and Fiume [23, 24]). In these methods, however, there is of-

The Visual Computer (1999) 15:471–482

ten a high computational cost for simulating clouds over the earth.

To address this problem, we propose a new method for modeling three-dimensional clouds based on a satellite image. In recent years, many methods of image-based rendering and/or modeling have been developed in computer graphics (Terzopoulos and Witkin [25]; Debevec et al. [1]; Gortler et al. [7]; Levoy and Hanrahan [14]; Seitz and Dyer [21]; Horry et al. [8]). The proposed method makes use of the image-based approach to generate clouds. Lee et al. [13] also use satellite images to generate clouds in order to visualize weather information. However, they use polygons to model them, resulting in poor-looking clouds. In the proposed method, the density distribution of clouds is represented by using metaballs. Parameters of metaballs (center positions, effective radii, and density values) are determined automatically so that a synthesized image of the clouds coincides with the satellite image. With the satellite image, the proposed method makes use of the fact that the color of clouds can be calculated by integrating the scattered light due to particles in them. In this method, clouds with a shape and color similar to real clouds can be automatically generated. Moreover, this paper also proposes an animation method for clouds generated by a sequence of satellite images. Note that our method does not reconstruct the exact three-dimensional shape of clouds in the satellite image. The goal of our method is to generate realistic clouds as easily as possible, so that they can be used for TV films and so on. In order to demonstrate the usefulness of the method, we apply it to animations of the visualization of weather information and the earth viewed from space.

In the following sections, the basic idea of modeling clouds is described in Sect. 2. Next, in Sect. 3, the details of the modeling method are discussed. In Sect. 4, the method for animating the generated clouds is proposed. In Sect. 5, several examples are given. Finally, in Sect. 6, conclusions and future work are discussed.

2 The basic idea of modeling clouds from a satellite image

Various representations can be used for modeling clouds. Examples are: voxels (Kajiya and Herzen [9]), procedural functions (Ebert et al. [4]; Ebert [1]), and metaballs (or blobs) (Stam and Fiume [24]; Nishita et al. [19]). With metaballs, clouds can easily be animated by moving their centers. Therefore, we use metaballs to model them as shown in Fig. 1.

A density value at a point P in the cloud is given by the following equation.

$$\rho(P) \approx \sum_{j=1}^{N} q_j f(r_j, R_j) \tag{1}$$

where N is the number of metaballs, q_j the density at the center of a metaball j, f the field function, and R_j the radius of the metaball. Further, r_j is the distance between the point P and the center position of a metaball, C_j; that is, $r_j = |P - C_j|$. We use the field function proposed by (Wyvill and Trotman [27]). That is to say,

$$f(r_j, R_j) = -\frac{4}{9}\left(\frac{r_j}{R_j}\right)^6 + \frac{17}{9}\left(\frac{r_j}{R_j}\right)^4 - \frac{22}{9}\left(\frac{r_j}{R_j}\right)^2 + 1,$$

when $r_j \leq R_j$, otherwise 0.

Determining parameters of the metaballs is equivalent to solving an inverse problem of determining the density distribution inside the clouds so that an image of synthesized clouds is similar to a satellite image. The problem is, however, very complicated and hard to solve. Therefore, we assume that the multiple scattering can be neglected. Furthermore, for modeling clouds, the attenuation of light due to cloud particles is approximated by a constant. Despite these assumptions, there is no unique solution to the problem. Therefore, the parameters of the metaballs are heuristically determined as follows. First, each pixel of the satellite image is classified as part of either a cloud region or a background region. To do this, the satellite image is converted to monotone. Then pixels with intensities higher than a specified threshold are identified as clouds. Next, one metaball is added at the pixel with the maximum intensity in the cloud region. After that, its radius and density at the center are optimized and the approximated image is calculated from the clouds modeled by metaballs. Then a new metaball is added if the error between the satellite image and the approximated image is greater than a specified threshold. These processes are repeated until the error is less than the threshold.

The overview of the proposed method is as follows.

1. Classify the pixels in the satellite image into either cloud regions or background regions.
2. Generate a metaball at the pixel with the maximum intensity in the cloud regions of the satellite image.
3. Optimize the radius and the density value at the center of the metaball.
4. Calculate the approximated image using the clouds modeled by metaballs.
5. If the error between the satellite image and the approximated image is less than a specified threshold, stop. Otherwise, go to step 6 after calculating the image of difference between the satellite image and the approximated image.
6. Go to step 3 after generating a metaball at the pixel with the maximum intensity in the cloud regions of the image of difference.

In the following sections, details of the method are discussed.

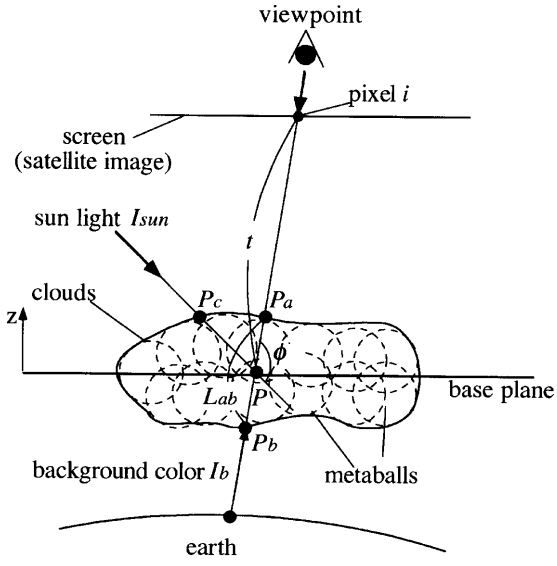


Fig. 1. Geometry of clouds

3 Modeling clouds with metaballs

In this section, the method for determining the parameters is proposed after we describe the method for calculating the error between the satellite image and the image synthesized by using clouds modeled by metaballs.

3.1 Calculating the error between the satellite image and synthesized image

The parameters of the metaballs are determined so that the mean square error between the satellite image and the synthesized image is minimized; that is:

$$Err = \frac{1}{M}\sum_{i=1}^{M}(I_i^{(\text{true})} - I_i)^2 \rightarrow \min\,, \tag{2}$$

where M is the number of pixels in the satellite image, and $I_i^{(\text{true})}$ and I_i are the intensity of the satellite image and the intensity of the synthesized image at pixel i, respectively. In the following, the calculation method of Err is described.

As shown in Fig. 1, let the density be $\rho(t)$ at a point P at distance t from the viewpoint P_v. Then the intensity I_i is calculated by the following equation (Kaneda et al. [10]).

$$I_i(\lambda) = I_b(\lambda)\exp(-\tau(L_{ab},\lambda)) + I_{\text{sun}}(\lambda)\beta(\phi,\lambda) \times \int_{P_a}^{P_b}\rho(t)\exp(-\tau(t,\lambda)-\tau(s,\lambda))\mathrm{d}t\,, \tag{3}$$

where I_b is the background color; λ, the wave length; L_{ab}, the distance from P_a to P_b; $\beta(\phi,\lambda)$, the phase function of the cloud particles; ϕ, the phase angle between the viewing ray and the sun direction; I_{sun}, the intensity of the sunlight outside the atmosphere; and $\tau(L_{ab},\lambda)$ is the optical length from P_a to P_b. The optical length is obtained by integrating the density of the cloud particles from P_a to P_b. Similarly, $\tau(t,\lambda)$ is the optical length from P_a to P; and $\tau(s,\lambda)$, from P_c to P (see Fig. 1). Although s is a function of t, we denote it as s instead of $s(t)$ for simplification. By putting Eq. 1 into Eq. 3, the following equation is obtained.

$$I_i(\lambda) = I_b(\lambda)\exp(-\tau(L_{ab},\tau)) + I_{\text{sun}}(\lambda)\beta(\phi,\lambda) \times \int_{P_a}^{P_b}\left\{\sum_{j=1}^{N} q_j f_j(r_j, R_j) \times \exp(-\tau(t,\lambda)-\tau(s,\lambda))\right\}\mathrm{d}t\,, \tag{4}$$

where

$$\tau(t, \lambda) = c \int_0^t \sum_{j=1}^{N} q_j f_j(r_j, R_j) \mathrm{d}t \,, \tag{5}$$

and c is the extinction coefficient. Since it is difficult to determine the parameters of metaballs so that the intensity of the satellite image coincides with the intensity calculated by Eq. 4, it is approximated as follows. First, the wavelength is neglected since scattering of light due to cloud particles obeys Mie scattering, and the spectrum of scattering is not much influenced. Therefore, the satellite image is converted to a monotone one and the phase function $\beta(\phi, \lambda)$ depends only on ϕ. Second, the attenuation terms $\exp(-\tau(L_{ab}, \lambda))$ and $\exp(-\tau(t, \lambda) - \tau(s, \lambda))$ in Eq. 4 are approximated as constants κ_1 and κ_2. Note that these approximations are only for modeling clouds. Then, the following equation is obtained.

$$\begin{aligned} I_i &= I_b \kappa_1 + I_{\mathrm{sun}} \beta(\phi) \kappa_2 \int_{P_a}^{P_b} \sum_{j=1}^{N} q_j f(r_j, R_j) \mathrm{d}t \\ &= I_b \kappa_1 + I_{\mathrm{sun}} \beta(\phi) \kappa_2 \sum_{j=1}^{N} q_j F(r_j, R_j) \,, \\ &\qquad (i = 1, \ldots, M) \end{aligned} \tag{6}$$

where

$$F(r_j, R_j) = \int_{P_j^{\mathrm{in}}}^{P_j^{\mathrm{out}}} f(r_j, R_j) \mathrm{d}t \,, \tag{7}$$

and P_j^{in} and P_j^{out} the intersection point of the viewing ray with the metaball j. F is the integrated function of the field function f and can be calculated analytically (Nishita and Nakamae [17]). To obtain the background intensity I_b a satellite image without clouds is required. We obtain this image by assembling the background regions of a set of satellite images by hand. The threshold to decide whether pixels belong in cloud regions or background regions is experimentally determined. Then, the intensity I_i of the synthesized image can be calculated by Eq. 6. This can be calculated incrementally every time a new metaball is added. Let us assume that $I_i^{(\mathrm{old})}$ is the intensity before adding the metaball k and that $I_i^{(\mathrm{new})}$ is the intensity after adding it. Then,

$$I_i^{(\mathrm{new})} = I_i^{(\mathrm{old})} + I_{\mathrm{sun}} \beta(\phi) \kappa_2 q_k F(r_k, R_k) \,. \tag{8}$$

The second term in Eq. 8 is evaluated at the pixels through which the viewing rays intersect the metaball k. Therefore, I_i can be calculated with a small computation cost, resulting in faster optimization processes for the parameters.

Using I_i obtained by this method, the error *Err* between the satellite image and the synthesized image is calculated. Metaballs are added repeatedly until the error is less than a specified tolerance.

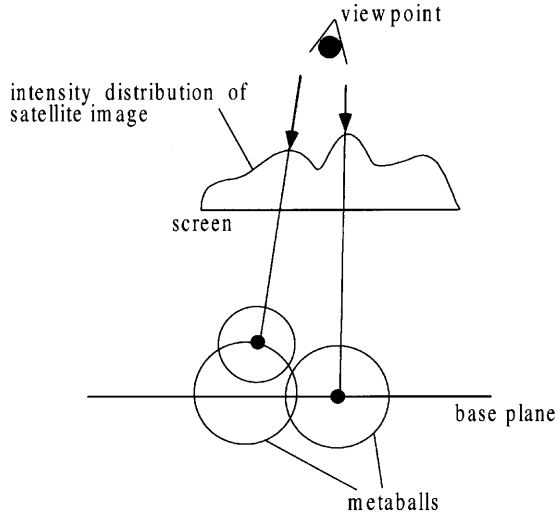


Fig. 2. Determination of center of metaballs

3.2 Determining the center of a metaball

When the number of metaballs becomes large, the computation for generating images takes a long time. Therefore, the number of metaballs should be as small as possible. Depending on the position of the center of each metaball, the number of metaballs changes for representing the clouds within the tolerance. The integrated function F of the field function f has a peak value at $r_j = 0$, and it decreases gradually as r_j increases, finally becoming zero at $r_j = R_j$. The number of metaballs tends to be small when the center position is chosen where the function F fits the intensity distribution of the satellite

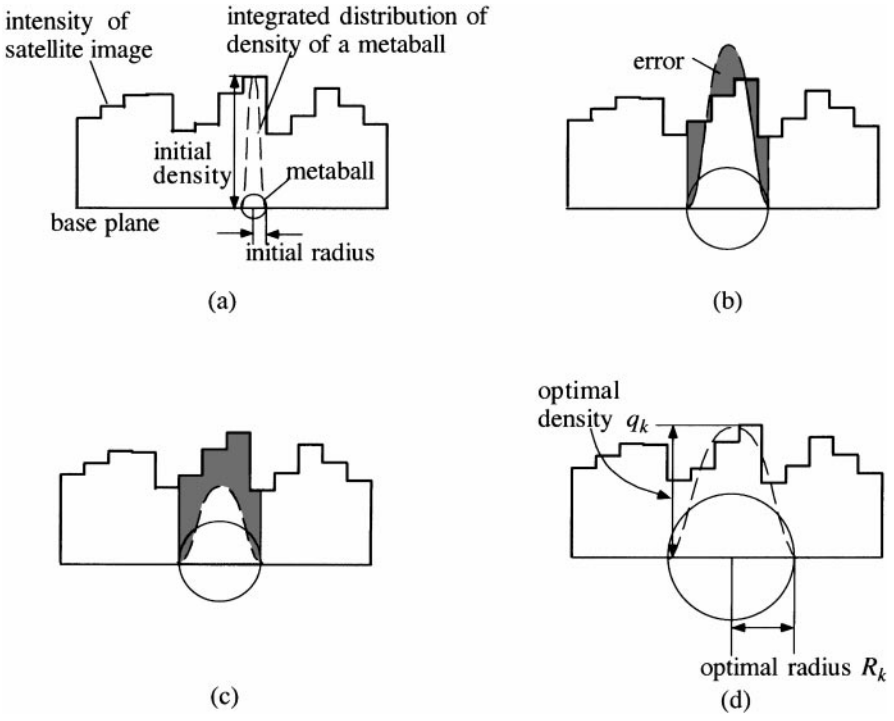


Fig. 3. Optimization steps for the density and radius

image. The reason for this is that the intensity distribution of cloud images is smooth and the distribution around the pixel with the maximum intensity is considered similar to the distribution of the function F. Therefore, the center position is determined on the viewing ray through the pixel with the maximum intensity. In order to determine the center position in three-dimensional space, a base plane is virtually generated (see Fig. 2). In our system, the base plane is set parallel to the image plane. Although generation works well in most cases, the user can also specify it manually. The center point of the metaball is set to the intersection point of the viewing ray with the base plane. After the metaballs have been placed at the intersection point, the intersection point of the viewing ray with the metaballs is calculated, and the center is set to the nearest point as shown in Fig. 2. After one metaball is generated, the image of the difference is calculated. Then it is used for determining the center of the next metaball in order to avoid the same pixel being chosen for the center point.

3.3 Optimization of the radius and density at the center

The basic idea for calculating radius R_k and density q_k at the center of the metaball k is to search for the optimal values by changing them little by little. Figure 3 shows the process for determining the radius and density. In Fig. 3, the base plane is parallel to the image plane, and the dotted curve indicates the function $F(r_k, R_k)$. For this search, the upper limit R^* for the radius is specified by the user.

The radius R_k and the density q_k are determined to minimize the difference of intensity between the synthesized image and the satellite image at pixels inside the radius of the metaball. In Fig. 3, the shaded areas indicate the difference. First, an initial radius and density are set to δ and $(I_i^{(\mathrm{true})} - \kappa_1 I_b)/(I_{\mathrm{sun}}\beta(\phi)\kappa_2 F(0, \delta))$, respectively (Fig. 3a). In other words, the radius and the density are initialized so that the intensity due to the metaball at the

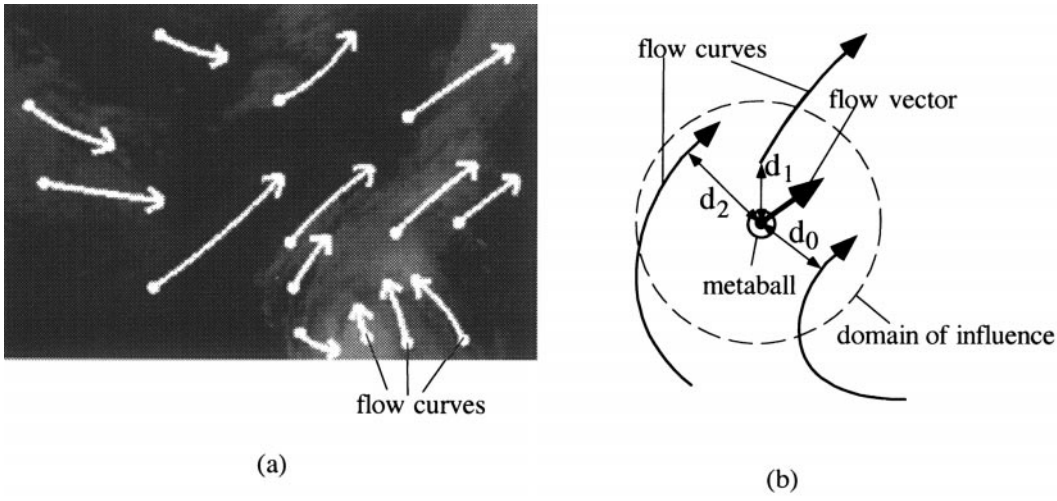


Fig. 4. Calculation of flow vectors

pixel is equal to that of the satellite image. Next, the density is fixed, and the optimal radius that minimizes the error is calculated by incrementing the radius by d (Fig. 3b). Then, after making the density smaller by w, the optimal radius is searched for again (Fig. 3c). By making the density smaller, the radius can be larger, hence one metaball can cover a larger area. This results in reducing the number of metaballs. This process is repeated until the optimal radius and density are found (Fig. 3d). In this paper, d is set to the size of one pixel and w is specified by the user. Note that a metaball is deleted if the density becomes zero during the search.

4 Animation of clouds

In this section, an method for animating clouds is proposed. First, clouds are modeled from satellite images taken at a certain interval. Then they are animated by interpolating the parameters of the metaballs.

In traditional morphing techniques, meshes are used for correspondence between images. Recently, more sophisticated methods have been developed: the user specifies the features of images by using points or line segments, and intermediate images are generated from the correspondence between these feature points or line segments. However, this method is not applicable to the animation of clouds. Clouds in one image move according to the flow of the atmosphere, resulting in the clouds in the next image. Therefore, their movement cannot be expressed just by interpolating the feature points. To create realistic cloud motion, we assume each metaball can be classified as one of the following three cases.

Case 1. Clouds move from one place to another.

Case 2. Clouds disappear while moving.

Case 3. Clouds appear while moving.

Although the flow analysis of the atmosphere is required for obtaining the exact movement of clouds taking these cases into account, this computation is too expensive. Thus, in the proposed method, the user inputs the flow as Bézier curves by observing a sequence of satellite images. Figure 4a shows an example of flow curves specified by the user. With the aid of such flowcurves, metaballs are classified as one of the three cases, and clouds are animated by changing their center positions, densities, and radii.

4.1 Classification of metaballs using flow curves

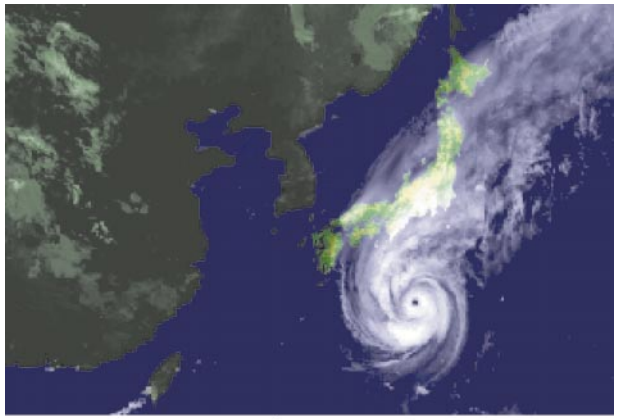
Let us denote the metaballs generated from satellite image A (recorded at time t_A) as $M_{A,i}$ ($i = 1, \ldots, n_A$), and the metaballs generated from satellite image B (recorded at t_B) as $M_{B,j}$ ($j=1, \ldots, n_B$). n_A and n_B are numbers of metaballs $M_{A,i}$ and $M_{B,j}$, respectively. First, to get correspondence between $M_{A,i}$ and $M_{B,j}$, we move metaballs $M_{A,i}$ using the flow curves. The method for this is described in the next section. Then, $M_{A,i}$ is associated with $M_{B,j}$ if $M_{B,j}$ is the closest, the distance between $M_{A,i}$ and $M_{B,j}$ is less than a specified threshold, and $M_{B,j}$ has not yet been associated with any other metaballs. After this process, each metaball is classified as one of the three cases. Metaball $M_{A,i}$ is classified as case 1 if there the corresponding metaball $M_{B,j}$ exists; otherwise it is classified as case 2. The metaball $M_{B,j}$ is classified as case 3 if none of $M_{A,i}$ corresponds to it. The density q_i and the radius R_i of each metaball at time t ($t_A < t < t_B$) are calculated as follows. The densities, radii, and center positions of case 1 metaballs are linearly interpolated by using the corresponding metaballs. For case 2, the densities are linearly faded out, and for case 3, they are faded in. For both these cases, the center positions are determined by a method described in the next section.

4.2 Moving metaballs with flow curves

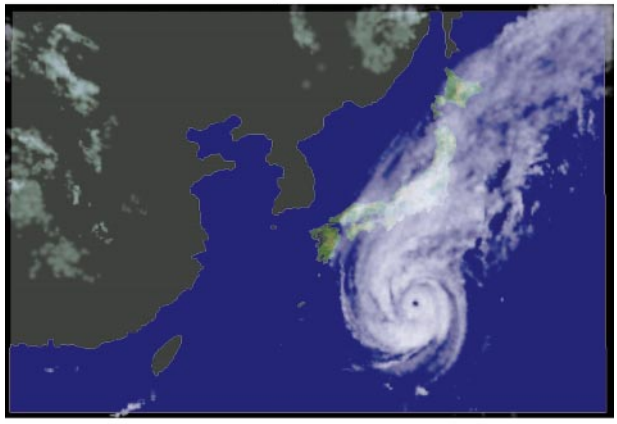
Each metaball moves toward a flow vector $\boldsymbol{v}$ calculated at its center with the use of the flow curves (Fig. 4b). First, the minimum distance d_k and closest point P_k from the center point to each Bézier curve are calculated. If the distance d_k is less than a specified distance D, the Bézier curve affects the movement of the metaball. Then tangent vector $\boldsymbol{v}_k$ at the closest point is calculated. The flow vector $\boldsymbol{v}$ at the point is defined as the weighted sum of the tangent vectors; that is, $\boldsymbol{v} = \sum_{k=1}^{n} w_k \boldsymbol{v}_k$. n is the number of Bézier curves inside the domain of influence. The weight w_k is calculated as follows. First, the contribution ratio a_k is calculated by:

$$a_k = \begin{cases} 1.0 - \left(\frac{d_k}{D}\right)^2 & (d_k \le D) \\ 0.0 & (d_k > D) \end{cases} . \tag{9}$$

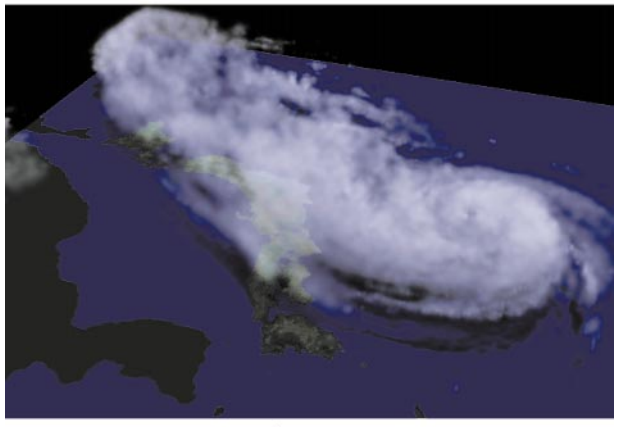
Then the weight w_k is calculated by normalizing a_k; that is,



a)



b)



c)

Fig. 5. Clouds generated from a satellite image. **a** A satellite image. **b** Clouds modeled by metaballs. **c** Clouds viewed from different viewpoint

$$w_k = a_k / \sum_{j=1}^{n} a_j . \tag{10}$$

The amount of movement in time step Δt is $\Delta t \boldsymbol{v}$.

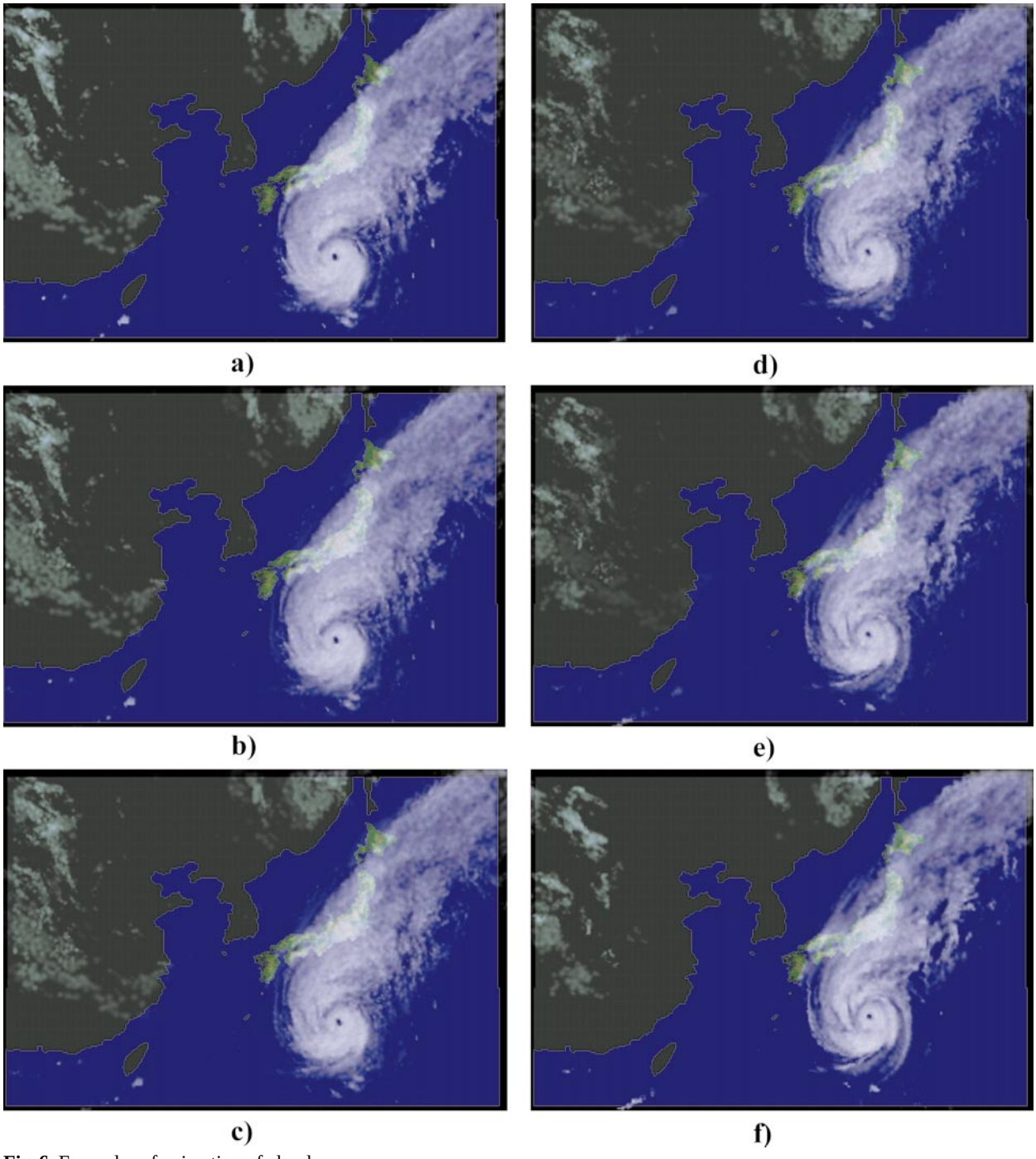


Fig. 6. Examples of animation of clouds

5 Examples

Figure 5 shows examples of clouds using satellite images viewed from space. Figure 5a is an infrared image of a typhoon taken from the meteorological satellite "Himawari". Figure 5b shows clouds modeled by metaballs. By comparing Fig. 5a with Fig. 5b, it can be seen that the proposed method

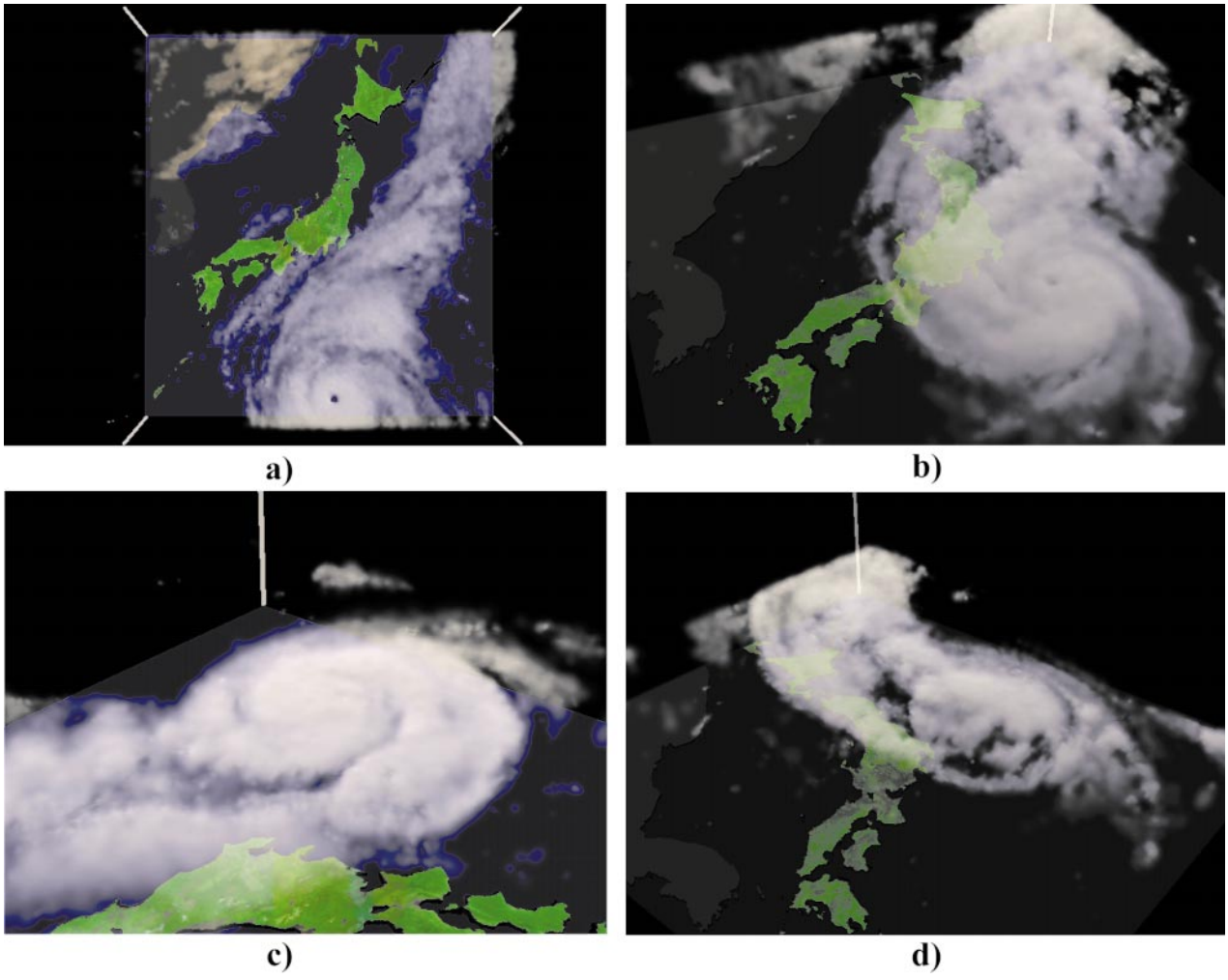


Fig. 7. Examples of the visualization of weather information

can generate clouds similar to those in satellite images. Figure 5c shows the same clouds from a different viewpoint. Since clouds are modeled in three-dimensional space, they are seen from different viewpoints, and shadows are cast on Japan. To generate Fig. 5b and c, the color of the clouds is calculated by taking into account the single scattering of light by using Eq. 4. The infrared image used here is available on the internet (Several sites provide satellite images, for example, http://www.jwa.go.jp/gms.html). This image was recorded at 21:00 on 16 September 1995.

In this example, the number of metaballs is 9534 and the calculation time for generating them is 15 s on a Silicon Graphics Indigo 2 Maximum Impact (195 MHz MIPS R10000). The calculation time for generating Fig. 5b is 9.5 min on the same machine.

Figure 6 shows an example of clouds animated by the proposed method as it is described in Sect. 4. Figure 6a and f were rendered by using clouds generated from satellite images and Fig. 6b–e, by interpolating them. The satellite image corresponding to Fig. 6a was recorded 3 h before the satellite image of Fig. 6f, which is the same as Fig. 5b. As shown in this example, the proposed method can realize the smooth and natural transition of clouds.

Next, Figs. 7 and 8 show images from two animations of the proposed method, entitled *Weather Report* and *The Earth II*. For both these animations, 13 satellite images were used to generate clouds and 900 images were produced by using the method de-

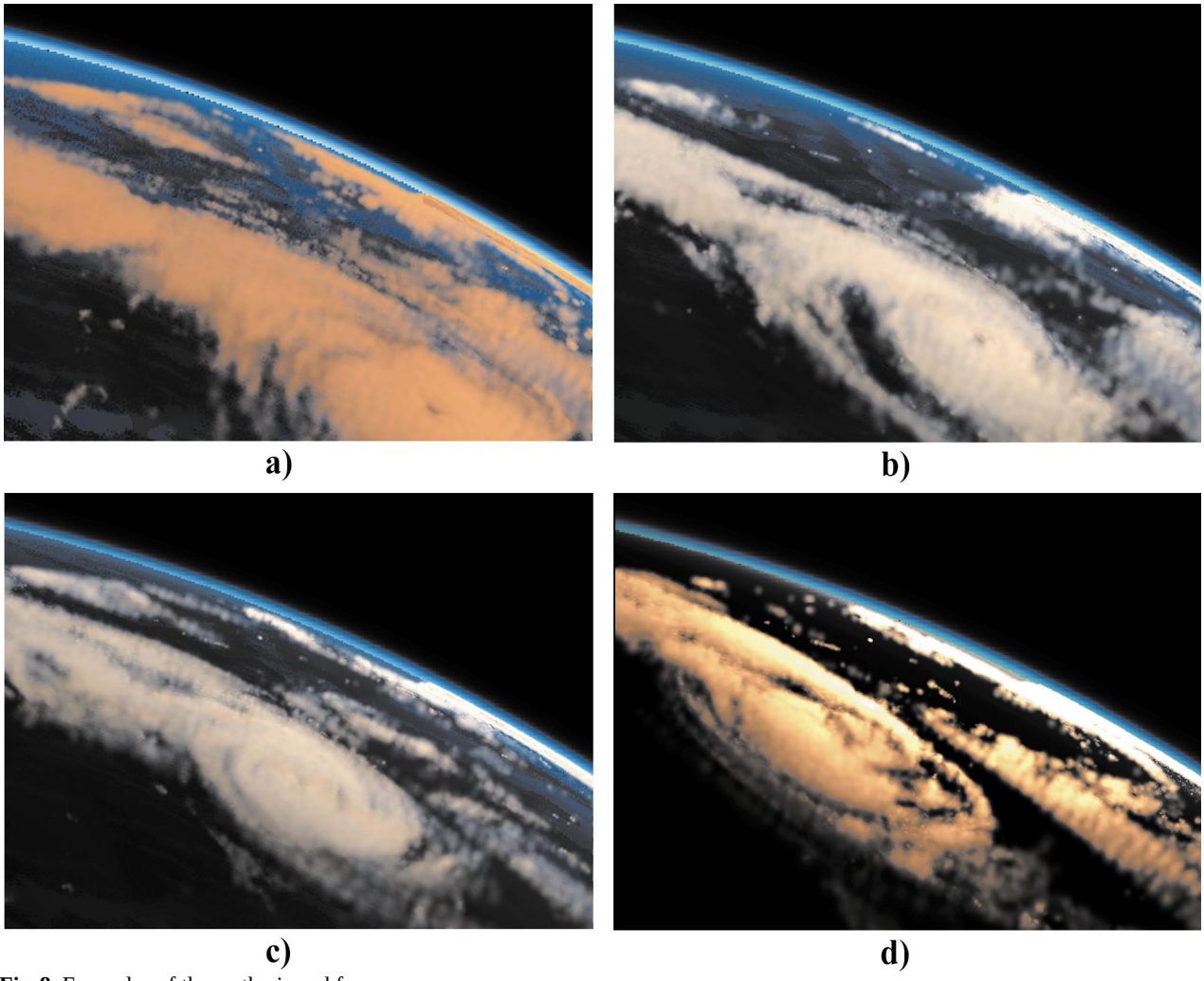


Fig. 8. Examples of the earth viewed from space

scribed in Sect. 4. Figure 7 is for visualizing weather information. In this animation, the movement of a typhoon passing across Japan is visualized. Figure 7 shows that the proposed method can generate realistic clouds even if the viewpoint is varied.

Figure 8 shows examples of the animation of the earth viewed from space. In this animation, the position of the sun changes. The color of the earth is calculated by taking into account the effect of the scattering of both sunlight and sky light due to the atmosphere with the method developed by Nishita and Nakamae [17]. In Fig. 8a, the sun is behind the viewer, turning the color of the clouds red. In Figs. 8b and c, the sun is above the clouds. Finally, in Fig. 8d, the sun is hidden behind the earth. These examples show that optical effects can be simulated when 3D clouds are generated. These effects cannot be simulated just by mapping the satellite image as textures.

6 Conclusions

A method that uses metaballs for modeling clouds in three-dimensional space from a single satellite image has been proposed. By making use of a satellite image of real clouds, the proposed method can generate clouds with authentic shapes and colors. We have also proposed an animation method for clouds modeled by metaballs. With the proposed method, three-dimensional clouds are generated, and they can be viewed from arbitrary directions. The color of the clouds can be calculated by simulating the optical

effects. Furthermore, the animated clouds are generated from a series of satellite images, and shapes of clouds at each frame are obtained by interpolating them.

There are several future projects:

1. In order to increase realism by adding various plausible fine features, some kind of turbulence model, such as the one used by Stam and Fiume [22] or Ebert [1] should be included.
2. To model clouds more accurately, methods for the estimation of the height of clouds from satellite images (Lee et al. [13]) should be incorporated.
3. Extending the proposed method to generate clouds from photographs taken from the ground will be useful for modeling clouds such as cumulonimbi.

Acknowledgements. The authors thank the reviewers for their constructive criticism and positive comments.

References

1. Debevec PE, Taylor CJ, Malik J (1996) Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. Proceedings of SIGGRAPH '96, New Orleans, LA 11–20
2. Ebert DS (1997) Volumetric modeling with implicit functions: a cloud is born. Visual Proceedings of SIGGRAPH '97 Los Angeles, California 147
3. Ebert DS, Parent RE (1990) Rendering and animation of gaseous phenomena by combining fast volume and scanline A-buffer techniques. Comput Graph 24:357–366
4. Ebert DS, Musgrave FK, Peachey P, Perlin K, Worley S (1994) Texturing and modeling: a procedural approach. Academic Press, Cambridge MA 267–294
5. Foster N, Metaxas D (1997) Modeling the motion of a hot, turbulent Gas. Proceedings of SIGGRAPH '97, Los Angeles, California 181–188
6. Gardner GY (1985) Visual simulation of clouds. Comput Graph 19: 297–303
7. Gortler SJ, Grzeszczuk R, Szeliski R, Cohen MF (1996) The lumigraph. Proceedings of SIGGRAPH '96, New Orleans, LA 43–54
8. Horry Y, Anjyo K, Arai K (1997) Tour into picture: using a spidery mesh interface to make animation from a single image. Proceedings of SIGGRAPH '97, Los Angeles, California 225–232
9. Kajiya JT, Herzen BPV (1984) Ray tracing volume densities. Comput Graph 18:165–174
10. Kaneda K, Okamoto T, Nakamae E, Nishita T (1991a) Photorealistic image synthesis for outdoor scenery under various atmospheric conditions. Visual Comput 7:247–258
11. Kaneda K, Yuan G, Nakamae E, Nishita T (1991b) Photorealistic visual simulation of water surfaces taking into account radiative transfer. Proceedings of the 2th CG&CAD'91, Hangzhou, China 25–30
12. Klassen RV (1987) Modeling the effect of the atmosphere on light. ACM Trans Graph 6:215–237
13. Lee CA, Kesselman C, Schwab S (1996) Near-real-time satellite image processing: metacomputing in CC++. IEEE Comput Graph Appl 16:79–84
14. Levoy M, Hanrahan P (1996) Light field rendering. Proceedings of SIGGRAPH '96, New Orleans, LA 31–42
15. Max N (1994) Efficient light propagation for multiple anisotropic volume scattering. Proceedings of the 5th Eurographics Workshop on Rendering, Darmstadt, Germany 87–104
16. Musgrave FK (1990) A note on ray tracing mirages. IEEE Comput Graph Appl 10:10–12
17. Nishita T, Nakamae E (1994) A method for displaying metaballs by using Bézier clipping. Comput Graph Forum 13:271–280
18. Nishita T, Takao S, Katsumi T, Nakamae E (1993) Display of the earth taking into account atmospheric scattering. Proceedings of SIGGRAPH '93, Anaheim, California 175–182
19. Nishita T, Dobashi Y, Nakamae E (1996) Display of clouds taking into account multiple anisotropic scattering and sky light. Proceedings of SIGGRAPH '96, New Orleans, LA 379–386
20. Sakas G (1993) Modeling and animating turbulent gaseous phenomena using spectral synthesis. Visual Comput 9:200–212
21. Seitz SM, Dyer CR (1996) View morphing. Proceedings of SIGGRAPH '96, New Orleans, LA 21–30
22. Stam J, Fiume E (1991) A multiple-scale stochastic modelling primitive. Proceedings of Graphics Interface'91, Calgary, Alberta 24–31
23. Stam J, Fiume E (1993) Turbulent wind fields for gaseous phenomena. Proceedings of SIGGRAPH '93, Anaheim, California 369–376
24. Stam J, Fiume E (1995) Depicting fire and other gaseous phenomena using diffusion processes. Proceedings of SIGGRAPH '95, Los Angeles, California 129–136
25. Terzopoulos D, Witkin A (1988) Physically-based models with rigid and deformable components. IEEE Comput Graph Appl 8:41–51
26. Voss R (1983) Fourier synthesis of Gaussian fractals: $1/f$ noises, landscapes, and flakes. In SIGGRAPH '83: Tutorial on State of the Art Image Synthesis, vol 10, ACM SIGGRAPH
27. Wyvill G, Trotman A (1990) Ray-tracing soft objects. Proceedings of CG International '90, Singpore 439–475

YOSHINORI DOBASHI has been a Research Assistant at the Department of Information Science, Hiroshima City University, Hiroshima, Japan since 1997. He received his BE, ME, and PhD in Electrical Engineering from Hiroshima University, Hiroshima, Japan in 1992, 1994, and 1997, respectively. His research interests include global illumination volume rendering, and animation. He is a member of Information Processing Society of Japan.

TSUYOSHI OKITA has been a Professor at the Department of Information Science, Hiroshima City University, Hiroshima, Japan since 1995. He taught at Yamaguchi University from 1978 to 1995. His research interests center on the system identification. He received his ME from Tohoku University, Japan, in 1964. He is a member of the Institute of Electrical Engineers of Japan and the Society of Instrument and Control Engineers.

TOMOYUKI NISHITA has been a Professor in the Department of Information Science at University of Tokyo, Japan since 1988. He taught at Fukuyama University from 1979 to 1998. His research interests center in computer graphics, including lighting models, hidden-surface removal, and antialiasing. He received his BE, ME, and PhD in Engineering in 1971, 1973, and 1985, respectively, from Hiroshima University. He is one of the pioneers of the radiosity method.

HIDEO YAMASHITA has been a Professor at the Department of Electrical Engineering, Hiroshima University, Hiroshima, Japan since 1992. He received his BE and ME in Electrical Engineering from Hiroshima University, in 1964 and 1968, respectively, and a PhD in Electrical Engineering from Waseda University Tokyo, Japan, in 1977. His interests include electric and magnetic fields analysis of electric machinery by finite element analysis and visualization of 3D magnetic fields by computer graphics. He is a member of the Institute of Electrical Engineers of Japan, the IEEE, and the ACM.