

# ANIMATION OF CLOUDS USING CELLULAR AUTOMATON

YOSHINORI DOBASHI, TOMOYUKI NISHITA<sup>†</sup>, TSUYOSHI OKITA

Hiroshima City University  
3-4-1 Ozukahigashi, Asaminami-ku,  
Hiroshima, 731-3194, Japan

<sup>†</sup>The University of Tokyo  
7-3-1 Hongo, Bunkyo-ku,  
Tokyo, 113-0033, Japan

## ABSTRACT

Recently, computer graphics have been used to simulate natural phenomena, such as clouds, fire, and ocean waves. This paper focuses on the evolution of clouds and proposes a simulation method for dynamic clouds. The method makes use of the cellular automaton for calculating the density distribution of clouds which varies over time. By using the cellular automaton, the distribution can be obtained with only a small amount of computation since the dynamics of clouds are expressed by several simple transition rules. The proposed method is applied to animations of outdoor scenes to demonstrate its usefulness.

**KEYWORDS:** clouds, animation, cellular automaton, natural phenomena, simulation.

## 1. INTRODUCTION

Clouds play an important role when generating images of outdoor scenes, flight simulators and so on. Displaying clouds in three-dimensional space is important since their shapes and colors are different depending on the viewpoint and the sun position. There have therefore been a lot of methods created to do this [1] [2] [3] [4] [5] [6] [7], and using these, photo-realistic images can be generated. The major purpose of these methods, however, is to generate still images. Meanwhile, fascinating animations of clouds with their shapes and colors changing are often used in movies, commercial films, and so on. Most of these animations are created by recording them in advance and replaying them quickly. It is useful to create realistic animations by using computer graphics.

This paper proposes a method for making the animation of clouds using the cellular automaton (CA). Thalmann also used CA for surface modeling such as water surface [8]. Its application to clouds, however, has not been demonstrated. The proposed method utilizes a numerical model developed by Nagel et al. for simulating the growth of clouds. In their model, the simulation can be done with a small amount of computation since the

dynamics are expressed by several simple rules. However, since their intention is to investigate the dynamics of clouds and the algorithm is not tuned for the visualization of clouds, the model has, for our purpose, the following disadvantages. (1)Clouds do not disappear. (2)Only binary information at a certain point is obtained, that is, there are clouds or there are no clouds. Therefore, we propose an improved model for creating realistic animations. Clouds are extinguished by introducing new rules. Continuous distribution is calculated by averaging the binary distribution obtained by the simulation.

In the following, Section 2 discusses the previous work. Next, the basic idea of the proposed method is described in Section 3. In Section 4, the numerical model developed by Nagel et al. is explained. In Section 5, the improved model and the control method are proposed. Next, in Section 6, the calculation method for obtaining the continuous distribution is proposed. Then, In Section 7, examples of the proposed method are demonstrated. Finally, in Section 8, conclusions of this paper are discussed.

## 2. PREVIOUS WORK

Kajiya et al. proposed a method for animating clouds by the numerical simulation of the fluid dynamics of the atmosphere [9]. In this method, however, solving complicated non-linear equations is difficult, requiring special knowledge to set proper boundary conditions. Gardner has produced the animation by using texture-mapped ellipsoids [2]. However, the scattering effects can not be simulated to calculate the color of clouds since the method does not create a true three-dimensional geometric model. Stam and Fuime have developed a method using the diffusion process [10] [11]. The method, however, is mainly used for displaying smoke and fire, and its application to cloud animation has not been demonstrated. Ebert et al. have developed a method combining metaballs and a noise function [12]. Using the method, an animation of cloud formation is created. It seems, however, that the shape of the clouds is determined in advance and their visible parts are

gradually increased. Moreover, no way to include time as a parameter is described. Neyret has proposed a method for forming clouds by qualitative simulation [13]. The method, however, focuses on simulating the growth of a single cluster of clouds and hence is not suitable for animation. Kikuchi et al. proposed a method utilizing the particle system to make an animation of cumulonimbus [14]. Cumulonimbus are formed by a dynamic simulation of the particles which takes into account several force, such as gravity. This method is also intended to simulate cloud growth and is not suitable to the animation of clouds with complicated shape transitions.

Compared to the previous methods, the proposed method has the following advantages.

- a) Formation and evolution of clouds can be simulated by a small amount of computation.
- b) Evolutionary Dynamics are expressed as simple transition rules by using CA.
- c) Density distribution in three dimensional space is obtained.

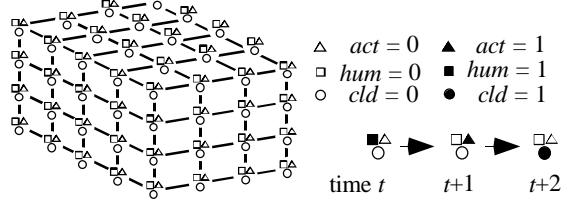
### 3. BASIC IDEA

The proposed method is based on the method developed by Nagel et al. to simulate the growth of clouds using CA [15]. In this method, the simulation space is divided into three dimensional grids as shown in Fig. 1. At each grid point, three state variables,  $hum$ ,  $cld$  and  $act$  are assigned which represent vapor, clouds, and the phase transition from vapor to clouds. The state of each variable is either 0 or 1. The growth of clouds is simulated by using simple transition rules. Since the state is either 0 or 1, the rules can be expressed by the Boolean language. Therefore, the simulation requires only a small amount of computation. The method has following disadvantages, however.

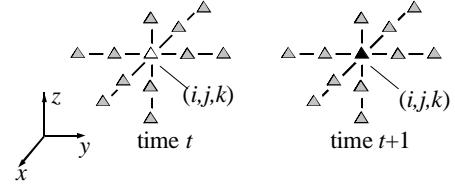
- Cloud extinction never occur since  $cld$ , after it has become 1, remains 1 forever. This means that it is impossible to create a dramatic animation including cloud formation and extinction.
- The simulation output is a binary distribution. What we can obtain is no more than *there are clouds* ( $cld = 1$ ) or, *there are not-clouds* ( $cld = 0$ ) at each grid point. Therefore, realistic images can not be generated since the density distribution in the real world is a continuous distribution between 0 and 1.

To create realistic animation of clouds, the proposed method addresses these problems as follows.

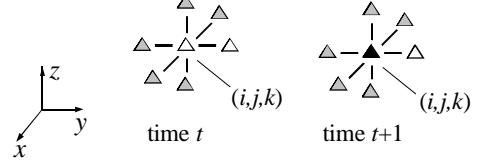
First, a new state variable,  $ext$ , and its transition rules are introduced to realize cloud extinction. In the real world, however, formation and extinction occur



(A) 3D GRIDS AND STATE TRANSITIONS



(B) STATE TRANSITION FOR  $act$  USED FOR GROWTH OF CLOUDS.



(C) STATE TRANSITION FOR  $act$  USED FOR SIMULATING WIND.

FIGURE 1: SIMULATION OF DYNAMIC CLOUDS USING CA.

repeatedly. To simulate this,  $hum$ ,  $act$  and  $ext$  are supplied at every frame using random numbers that obey a user-specified probability distribution. Animators can control the motion of the clouds in their design of the probability function. This achieves simulation of the complicated motion of clouds.

There are two possible approaches to the second problem. One is to extend the simulation process to treat continuous values. The other is to calculate the continuous distribution based on the binary distribution obtained by the simulation. In the first approach, we have to develop the new transition rules to take into account continuous values. However, the computation time is increased since the transition rules no longer have simple Boolean expressions. Therefore, in this paper, we choose the second approach. Continuous distribution is calculated in the post process of the simulation. Using the proposed method, the continuous distribution can be obtained without increasing the simulation time.

### 4. CLOUD GROWTH SIMULATION USING CELLULAR AUTOMATON

The simulation method developed by Nagel et al.[15] is explained in this section. As mentioned before, the simulation space is represented by 3D grids, and three state variables, *hum*, *act*, and *cld*, are assigned at each grid point (see Fig. 1(a)). The state of each variable is either 0 or 1. Their grid status (*i,j,k*) at time *t+1* are calculated by the status at time *t* using the following transition rules.

$$\begin{aligned} \text{act}(i,j,k,t+1) = & \text{.NOT.} \text{act}(i,j,k,t) \text{.AND.} \text{hum}(i,j,k,t) \\ & \text{.AND.} \text{fact}(\bullet) \end{aligned} \quad (1)$$

$$\begin{aligned} \text{cld}(i,j,k,t+1) = & \text{cld}(i,j,k,t) \text{.OR.} \text{act}(i,j,k,t) \\ & (2) \end{aligned}$$

$$\text{hum}(i,j,k,t+1) = \text{hum}(i,j,k,t) \text{.AND.} \text{.NOT.} \text{act}(i,j,k,t) \quad (3)$$

where, *fact*( $\bullet$ ) is a Boolean function and its value is calculated by the status of *act* around the grid. Fig. 1(a) shows the above transition rules. *act* becomes 1 at time *t+1* if both of the *hum* and *fact*( $\bullet$ ) are 1 at time *t*. Then *cld* becomes 1 at time *t+2*. In [15], the following function is used for *fact*( $\bullet$ ) by taking into account the fact that clouds grow upward and horizontally.

$$\begin{aligned} \text{fact}(\bullet) = & \text{act}(i+1,j,k,t) \text{.OR.} \text{act}(i-1,j,k,t) \\ & \text{.OR.} \text{act}(i,j+1,k,t) \text{.OR.} \text{act}(i,j-1,k,t) \\ & \text{.OR.} \text{act}(i,j,k+1,t) \text{.OR.} \text{act}(i,j,k-1,t) \\ & \text{.OR.} \text{act}(i+2,j,k,t) \text{.OR.} \text{act}(i-2,j,k,t) \\ & \text{.OR.} \text{act}(i,j+2,k,t) \text{.OR.} \text{act}(i,j-2,k,t) \\ & \text{.OR.} \text{act}(i,j,k-2,t) \end{aligned} \quad (4)$$

That is, as shown in Fig. 1(b), *fact*( $\bullet$ ) returns to 1 if the state of *act* of one of the shaded grids around the grid (*i, j, k*) is 1. By changing the rule of *fact*( $\bullet$ ), it is possible to simulate various effects. For example, we found that the following rule is suitable for clouds advected by winds (see Fig. 1(c)).

$$\begin{aligned} \text{fact}(\bullet) = & \text{act}(i+1,j,k,t) \text{.OR.} \text{.OR.} \text{act}(i,j,k+1,t) \\ & \text{.OR.} \text{act}(i-1,j,k,t) \text{.OR.} \text{act}(i,j-1,k,t) \\ & \text{.OR.} \text{act}(i,j,k-1,t)) \end{aligned} \quad (5)$$

In this case, the wind blows toward the plus direction of *y* axis.

Beginning from initial random status, cloud growth is simulated by updating the state of each variable using Eq. 1 through 3. The initialization is as follows. First, *hum* is initialized by using uniform random numbers of probability *p<sub>hum</sub>*. That is, *hum* is set to 1 if a random number between 0 and 1 is less than *p<sub>hum</sub>*, otherwise *hum* is set to 0. Similarly, *act* is set to either 0 or 1 by using the probability *p<sub>act</sub>*, but it cannot be set to 1 when *hum* is 0. *cld* is set to zero.

In this method, *act* propagates with changing vapor (*hum*=1) into clouds (*cld*=1). For further detail, see [15].

## 5. EXTENSION TO DYNAMIC CLOUD SIMULATION

The method described in the previous section has the advantage of low computational cost since the Boolean operations are used for the simulation. However, there are serious drawbacks to applying it to cloud animation. 1) *cld* remains 1 forever after *cld* has become 1 (see Eq. 2). 2) The state transition converges and stops after sufficient simulation. The complicated movement of clouds can not, therefore, be simulated. In the following, we propose methods for solving the problems.

### 5.1 CLOUD EXTINCTION

To achieve cloud extinction, a new variable *ext* is introduced. *ext* is just used to make clouds disappear and it has no physical meaning. Its transition rule is given as follows.

$$\begin{aligned} \text{ext}(i,j,k,t+1) = & \text{.NOT.} \text{ext}(i,j,k,t) \text{.AND.} \text{cld}(i,j,k,t) \\ & \text{.AND.} \text{fext}(\bullet) \end{aligned} \quad (6)$$

*fext*( $\bullet$ ) is a Boolean function similar to *fact* and its value is calculated by the state of *ext* around the grid. In this paper, the same shape is used for *fext* and *fact*. To make clouds disappear using the *ext* variable, the transition rule for *cld* is modified. That is,

$$\begin{aligned} \text{cld}(i,j,k,t+1) = & \text{.NOT.} \text{ext}(i,j,k,t) \text{.AND.} \\ & (\text{cld}(i,j,k,t) \text{.OR.} \text{act}(i,j,k,t)) \end{aligned} \quad (7)$$

Using the above rule, *cld* becomes 0 after *ext* is 1. However, making *cld* 0 just after *ext* has become 1 results in very unnatural animation since the formation and extinction of clouds are repeated frequently in a short period. To avoid this, the extinction time *T<sub>ext</sub>* is introduced. That is, *cld* becomes 0 after *T<sub>ext</sub>* steps when *ext* becomes 1. This achieves the natural shape transition of clouds.

### 5.2 CONTROL OF CLOUD MOTION

*hum*, *act*, and *ext* are supplied to simulate complicated cloud motion. That is, at every frame, the variables are checked to see whether their status is 0. The variables whose status are 0 are changed to 1 by random numbers that obey a specified probability function. Thus, clouds are gradually generated again in the regions where they have disappeared. Afterward, they may disappear due to the effect of the *ext* variable. These are repeated

again and again resulting in the complicated motion of clouds. Animator can control the motion by specifying the different probabilities at each grid point at different times. For example, making the probability high for *hum* results in a large amount of clouds. Since it is impractical to specify probability at each grid separately, the user specifies probabilities at several representative points and then the system calculates probabilities at each grid by interpolating them.

## 6. CALCULATION OF CONTINUOUS DENSITY DISTRIBUTION

The method for calculating the continuous density distribution is described in this section. The proposed method uses metaballs [16] to obtain continuous distribution. Fig. 2(a) shows the idea of metaballs. In the figure, the vertical axis indicates the density and the horizontal axis indicates the distance from the center of a metaball. As shown in Fig. 2(a), metaballs are spheres in which a field function is defined. A metaball has two parameters, that is, density at the center and effective radius. Fig. 2(b) shows the idea of calculating continuous distribution using metaballs. In the figure, for simplicity, the density distribution is represented in one dimension. As shown in Fig. 2(b), metaballs are placed at each grid point and the continuous distribution is represented as a weighted sum of the field functions. Continuous distribution can be obtained by adjusting densities at their centers and their effective radii, based on the binary distribution. In the proposed method, the effective radius  $R$  is specified by the user. The density  $q$  at the center of each metaball is determined as follows.

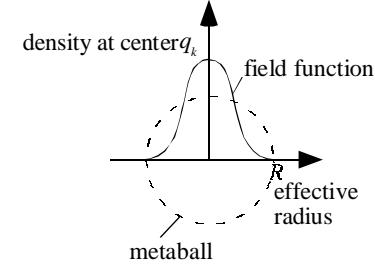
The binary distribution is considered to be the one obtained by halftoning the continuous distribution. This implies that a density at a point is approximately calculated by averaging the binary distribution around it. That is, the density  $q_{i,j,k}$  of the metaball placed at a grid  $(i,j,k)$  is calculated by the following equation.

$$q_{i,j,k} = \frac{1}{n_c} \times \sum_{l,m,n \in \Omega(\mathbf{x}_{i,j,k})}^{n_c} cld(\mathbf{x}_{l,m,n}) \quad (8)$$

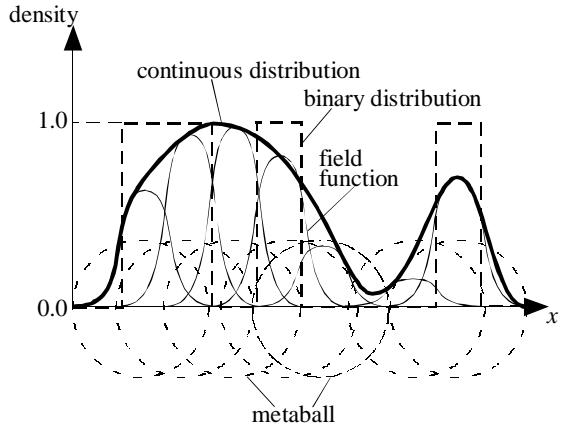
where  $cld$  is the binary distribution and  $\mathbf{x}_{l,m,n}$  is the coordinate of the grid  $(l,m,n)$ .  $\Omega(\mathbf{x}_{i,j,k})$  indicates a set of grids satisfying  $|\mathbf{x}_{i,j,k} - \mathbf{x}_{l,m,n}| < R$  and  $n_c$  is the number of such grids.  $R$  is the effective radius specified by the user. Thus, continuous distribution is given by the following equation.

$$\rho(\mathbf{x}) = \sum_{i,j,k \in \Omega(\mathbf{x})}^N q_{i,j,k} f(|\mathbf{x} - \mathbf{x}_{i,j,k}|) \quad (9)$$

where  $N$  is the number of metaballs that include the point  $\mathbf{x}$  inside  $R$  and  $f$  is the field function (see Appendix).



(A) DEFINITION OF METABALL.



(B) CONTINUOUS DISTRIBUTION REPRESENTED BY METABALLS.

FIGURE 2: CALCULATION OF CONTINUOUS DISTRIBUTION USING METABALLS.

## 7. EXAMPLES

We created a simple animation to investigate the usefulness of the proposed method. 100 steps are calculated. Figs. 3(a) through (f) are images at the steps 10, 20, 30, 40, 50, and 60, respectively. The viewpoint is above the clouds. In this example, 3D grids of 80x80x10 are used for the simulation and continuous distribution is generated by using the proposed method. To simulate clouds advected by wind, the rule expressed by Eq. (5) is used for  $fact(\bullet)$ . To calculate the color of clouds, only the single scattering of light is taken into account. The computation time for the simulation takes 19 seconds, that is, 0.19 seconds per one step. The computer is SGI Indigo2 MAXIMUM IMPACT(R10000 195 MHz). This

example demonstrates that the proposed method can generate realistic movement of clouds and the simulation can be done very quickly.

Next, we applied the proposed method to animation of natural scenery. Figs. 4 shows examples from the animation. In this example, the viewpoint is also changed. Fig. 5 shows another examples from the same animation. In this example, the sun position is changed from daytime to evening. The color of the sky is calculated by using the method proposed by Dobashi et al [17]. As shown in these images, realistic animation can be created by using the proposed method. Additionally, these examples show that the color of clouds depends greatly on the viewpoint and the sun position. In particular, the color turns red in the evening and this results in a fascinating animation when combined with their movement.

## 8. CONCLUSION

This paper has proposed a method for simulating the dynamic movement of clouds using CA and demonstrated its usefulness by realistic animations. The proposed method has the following features.

- (1) The simulation requires a small amount of computation since the dynamics of clouds is expressed by a simple Boolean operation.
- (2) The movement of clouds can be controlled by specifying the probability distributions for *hum*, *act*, *ext*.
- (3) The continuous density distribution is obtained by using metaballs. This makes it possible to create a realistic animation.

## ACKNOWLEDGEMENT

The authors would like to thank Prof. H. Yamashita, Dr. K. Kaneda, and Mr. N. Nishimoto in Hiroshima University for their helpful discussions.

## REFERENCES

- [1] R. Voss, Fourier Synthesis of Gaussian Fractals: 1/f noises, landscapes, and flakes, *SIGGRAPH'83: Tutorial on State of the Art Image Synthesis*, 10, 1983.
- [2] G.Y. Gardener, Visual Simulation of Clouds, *Computer Graphics*, 19(3), 1985, 279-303.
- [3] D. S. Ebert and R. E. Parent, Rendering and Animation of Gaseous Phenomena by Combining Fast Volume and Scanline A-Buffer Techniques, *Computer Graphics*, 24, 1990, 357-366.
- [4] K. Kaneda, T. Okamoto, E. Nakamae, and T.

Nishita, Photorealistic Image Synthesis for Outdoor Scenery under Various Atmospheric Conditions, *The Visual Computer*, 7(5&6), 1991, 247-258.

- [5] N. Max, Efficient Light Propagation for Multiple Anisotropic Volume Scattering, *Proc. of the Fifth Eurographics Workshop on Rendering*, 1994, 87-104.
- [6] J. Stam, Stochastic Rendering of Density Fields, *Proc. of Graphics Interface'94*, 1994, 51-58.
- [7] T. Nishita, Y. Dobashi, E. Nakamae, Display of Clouds Taking into Account Multiple Anisotropic Scattering and Sky Light, *Proc. of SIGGRAPH'96*, 1996, 379-386.
- [8] D. Thalmann, A "Lifegame" Approach to Surface Modeling and Renderling, *The Visual Computer*, 2(6), 1986, 384-390.
- [9] J. T. Kajiya and B. P. V. Herzen, Ray Tracing Volume Densities, *Computer Graphics*, 1984, 18, 165-174.
- [10] J. Stam, E. Fiume, Turbulent Wind Fields for Gaseous Phenomena, *Proc. of SIGGRAPH'93*, 1994, 369-376.
- [11] J. Stam, E. Fiume, Dipicting Fire and Other Gaseous Phenomena Using Diffusion Processes, *Proc. of SIGGRAPH'95*, 1995, 129-136.
- [12] D. S. Ebert, Volumetric Modeling with Implicit Functions: A Cloud is Born, *Visual Proc. of SIGGRAPH'97*, 1997, 147.
- [13] F. Neyret, Qualitative Simulation of Convective Clouds Formation and Evolution, *EGCAS'97*, 1997, 113-124.
- [14] T. Kikuchi, K. Muraoka, and N. Chiba, Visual Simulation of Cumulonimbus Clouds, *The Journal of The Institute of Image Electronics and Electronics Engineers of Japan*, 27(4), 1998, 317-326 (in Japanese).
- [15] K. Nagel and E. Raschke, Self-Organizing Criticality in Cloud Formation?, *Phisica A*, 182, 1992, 519-531.
- [16] G. Wyvill and A. Trotman, Ray-Tracing Soft Objects, *Proc. of CG International*, 1990, 439-475.
- [17] Y. Dobashi, T. Nishita, K. Kaneda, and H. Yamashita, A Fast Display Method of Sky Color Using Basis Functions, *The Journal of Visualization and Computer Graphics*, 8(2), 1997, 115-127.

## APPENDIX

The field function proposed by Wyvill et al. is given by the following equation [16].

$$f(r) = \begin{cases} -\frac{4}{9}a^6 + \frac{17}{9}a^4 - \frac{22}{9}a^2 + 1, & (r \leq R) \\ 0, & (r > R) \end{cases}, \quad (\text{A.1})$$

where  $a = r / R$ ,  $r$  is the distance from the center of a metaball to a calculation point, and  $R$  the effective radius of the metaball (see Fig 2(a)).

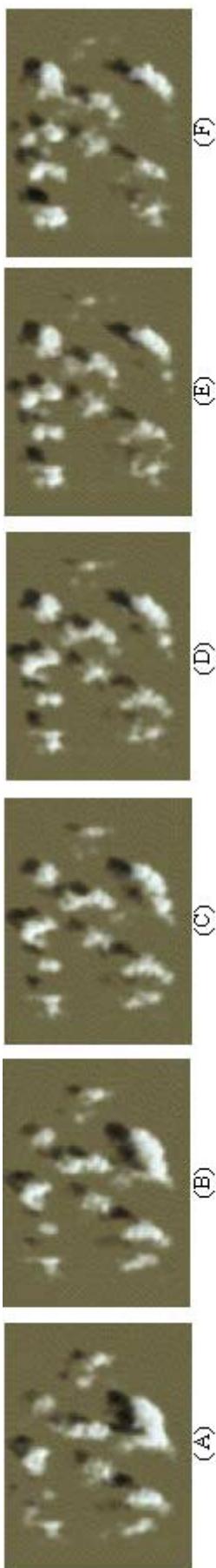


FIGURE 3: EXAMPLES OF A SIMPLE ANIMATION.



FIGURE 4: EXAMPLES FROM ANIMATION OF NATURAL SCENERY.

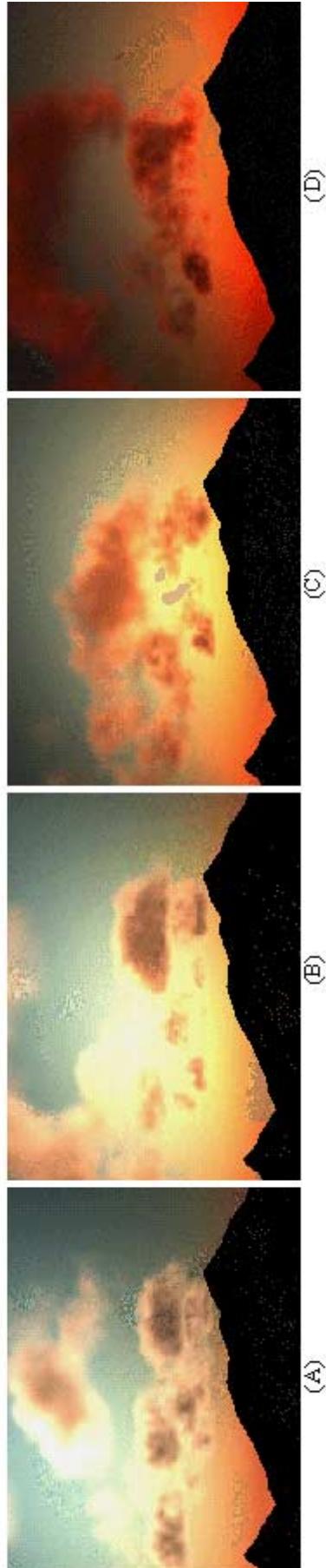


FIGURE 5: EXAMPLES FROM ANIMATION OF NATURAL SCENERY.