A Simple Efficient Method for Realistic Animation of Clouds

Yoshinori Dobashi Kazufumi Kaneda   Hideo Yamashita
Tsuyoshi Okita   Tomoyuki Nishita
Year 2000

Presented By:
Neha Dhamija
Vipin Vishvkarma

Contents

- Introduction
- Simulation
- Rendering
- Results
- Conclusion

Problem Definition

- Model clouds in runtime
- Animate them realistically
- Solution should be efficient

- It can be broken into two sub-problems
  - Simulate the process of cloud formation
  - Rendering of clouds

Simulation: Possible solution

- Simulate physical process of fluid dynamics
  - Very accurate
  - Computationally expensive

- Simulate clouds using an heuristic approach
  - Computationally inexpensive
  - Easier to implement
  - Requires tweaking of acceptable result

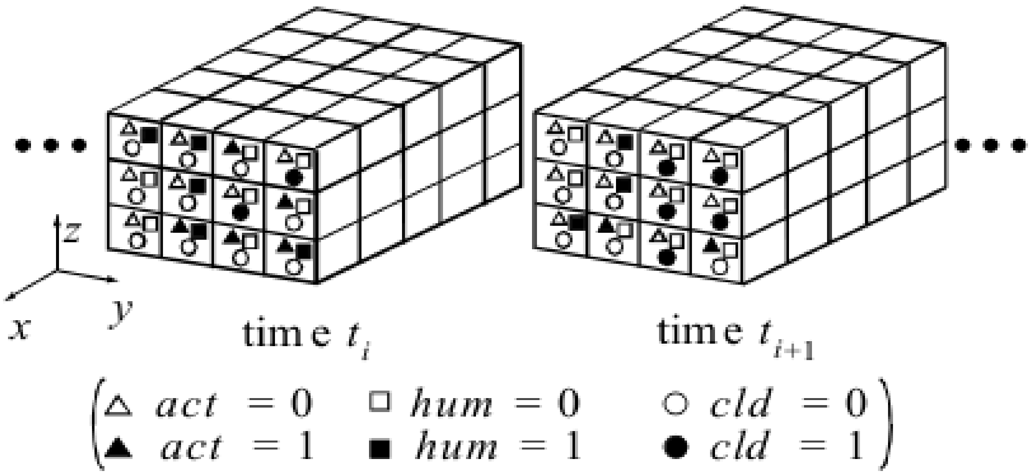
Simulation: Author’s approach

- Intermediate between the two
- Model clouds using cellular automaton and Boolean operations
- Considers the following
    - Extinction of clouds
    - Wind effects
    - Controlling cloud motion
    - Speeding up of the simulation

Simulation: Cloud formation

- When wet air particles move upward and reach the height of dew point, clouds are formed
- Use Nagel's method to simulate formation
  - Divide 3-D space evenly into 3-D cells
  - Assign Boolean variables to each cell

Simulation: Cellular Automaton



- Voxels correspond to cells
- Three logical Boolean variables at each cell
    - hum: indicates whether cell has enough water vapor to form clouds
    - act: indicates whether phase transition is ready to occur
    - cld: indicates whether cell contains clouds

Simulation : Cloud Growth

- Cell properties in the current animation frame ti are used to compute the cell properties in the next frame ti+1:

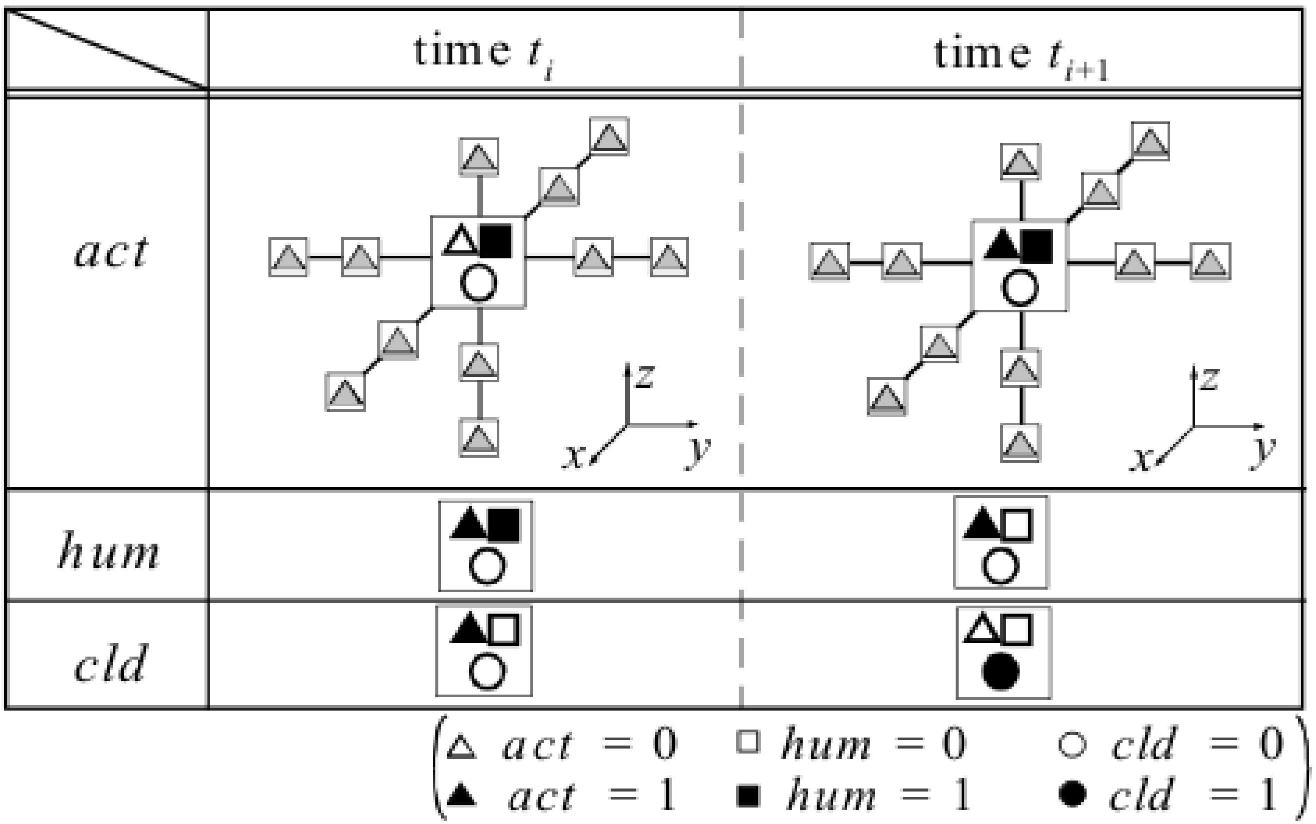
$$hum(x, y, z, ti+1) = hum(x, y, z, ti) \wedge \neg act(x, y, z, ti)$$

$$cld(x, y, z, ti+1) = cld(x, y, z, ti) \vee act(x, y, z, ti)$$

$$act(x, y, z, ti+1) = \neg act(x, y, z, ti) \wedge hum(x, y, z, ti) \wedge f act(x, y, z, ti)$$

f*act* is a boolean function and its value is calculated by the status of *act* in the surrounding cells.

Simulation: Cloud Growth



Simulation: Cloud Extinction

- Extension to Nagel's method:

$$cld(x, y, z, ti+1) = cld(x, y, z, ti) \wedge IS(rnd > pext(x, y, z, ti))$$

$$hum(x, y, z, ti+1) = hum(x, y, z, ti) \vee IS(rnd < phum(x, y, z, ti))$$

$$act(x, y, z, ti+1) = act(x, y, z, ti) \vee IS(rnd < pact(x, y, z, ti))$$

  - *rnd :* uniform random number
  - *pext :* probability of cloud extinction
  - *phum :* probability of vapor forming
  - *pact :* probability of phase transition occurence

Simulation: Advection by Wind

- Clouds move, blown by winds
- Wind velocity is different depending on the height from the ground

$$hum(i,j,k,t_{i+1}) = \begin{cases} hum(i-v(z_k),j,k,t_i), & i-v(z_k)>0 \\ 0, & \text{otherwise} \end{cases},$$

$$cld(i,j,k,t_{i+1}) = \begin{cases} cld(i-v(z_k),j,k,t_i), & i-v(z_k)>0 \\ 0, & \text{otherwise} \end{cases},$$

$$act(i,j,k,t_{i+1}) = \begin{cases} act(i-v(z_k),j,k,t_i), & i-v(z_k)>0 \\ 0, & \text{otherwise} \end{cases},$$

  - *v(z) :* wind velocity, piecewise linear function
  - Assumption: wind blows towards the direction of x-axis

Simulation: using Bit Field

- Each cell state (cld, act, hum) can be stored in a single bit
- Low memory requirements
- Fast computation of simulation process
- Problem: Random numbers
  Solution: Precalculated look-up tables

Simulation: cloud motion

- Motion of cloud can be controlled by using ellipsoids
- Ellipsoids simulate air parcels
- Vapor and phase transition probability:
  - higher at center / lower at edge
- Cloud extinction probability:
  - Lower at center / higher at edge
- Ellipsoids move in direction of wind
- Different kinds of clouds by controlling ellipsoid parameters (sizes and position)

Rendering

Rendering: Previous Work

- Using Direct Mapping Techniques

  simple & efficient but unrealistic & static
- Using Physical Models

  with scattering/absorption

  with multiple scattering and skylight
- Using 3D textures

  simple, efficient, expensive hardware, not realistic enough

Rendering

- Shafts of light using ray-tracing

  highly inefficient

- Shafts of light using scan-line accumulation buffer

  faster but still inefficient

- THE PROPOSED METHOD: making use of graphics hardware(OpenGL) for displaying clouds and shafts of light.

Continuous Density Distribution

- 0-1 values obtained for simulation smoothened to continuous values.
- Space and time distribution of each cell:

$$q(i,j,k,t_i) = \frac{1}{(2t_0+1)(2k_0+1)(2j_0+1)(2i_0+1)} \sum_{t'=-t_0}^{t_0} \sum_{k'=-k_0}^{k_0} \sum_{j'=-j_0}^{j_0} \sum_{i'=-i_0}^{i_0} w(i',j',k',t')\, cld(i+i', j+j', k+k', t_i+t'),$$

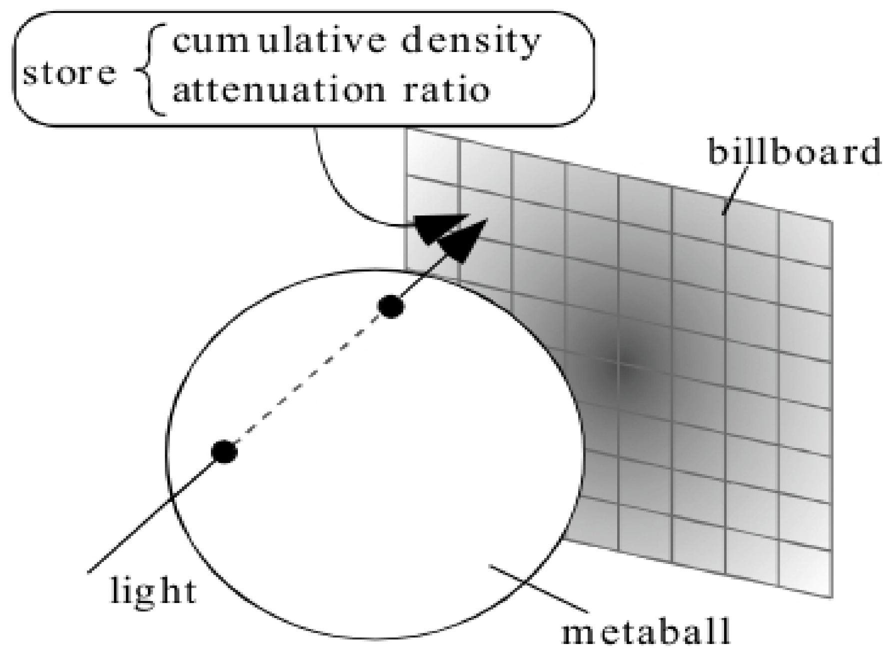
- Clouds rendered as 3D Metaballs. Cell density distributed over these.

$$\rho(\mathbf{x}, t_i) = \sum_{i,j,k \in \Omega(\mathbf{x},R)}^{N} q(i,j,k,t_i) f(|\mathbf{x} - \mathbf{x}_{i,j,k}|),$$

No animation frames. Runtime computations.

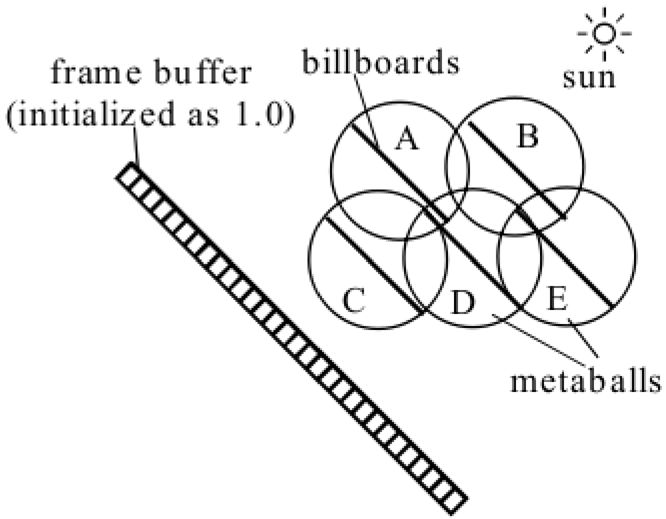
Rendering

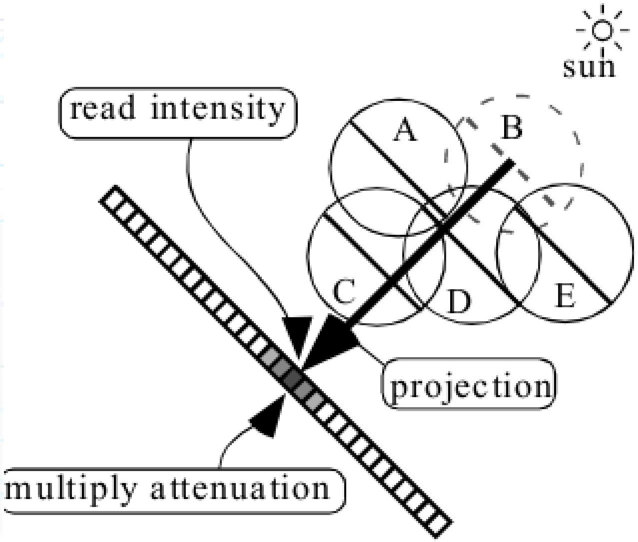
- 2D texture generated on billboard for each metaball using splatting



- Discretization of density due to huge memory requirement.
- Closest billboard mapped to metaball.

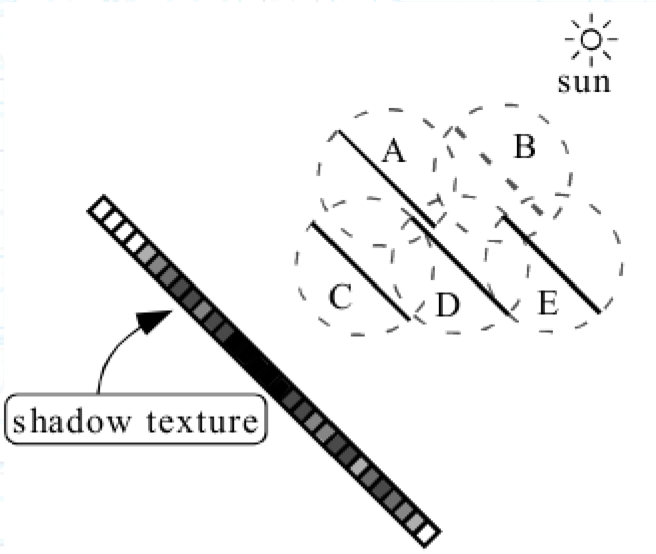
Step1: Intensity Computation





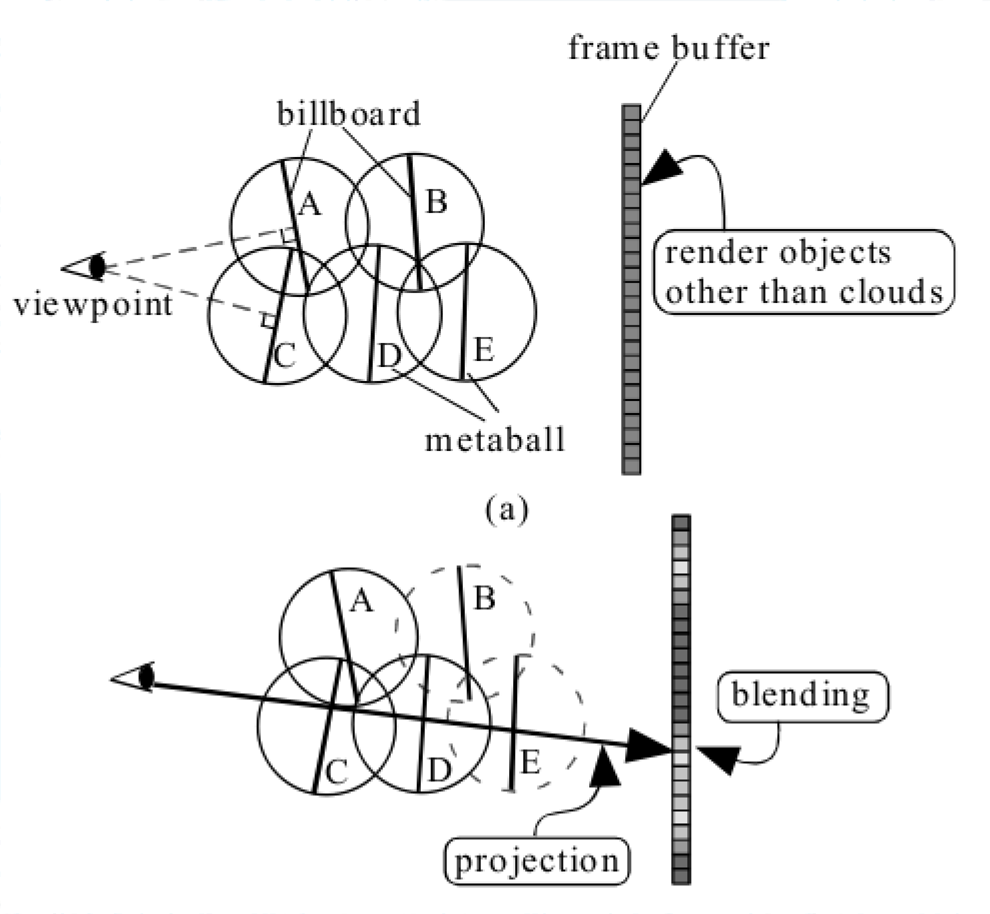
- View from Sun's direction
- Process metaballs in decreasing depth order

Step1: Intensity Computation



- Final image used as light map texture for shadows

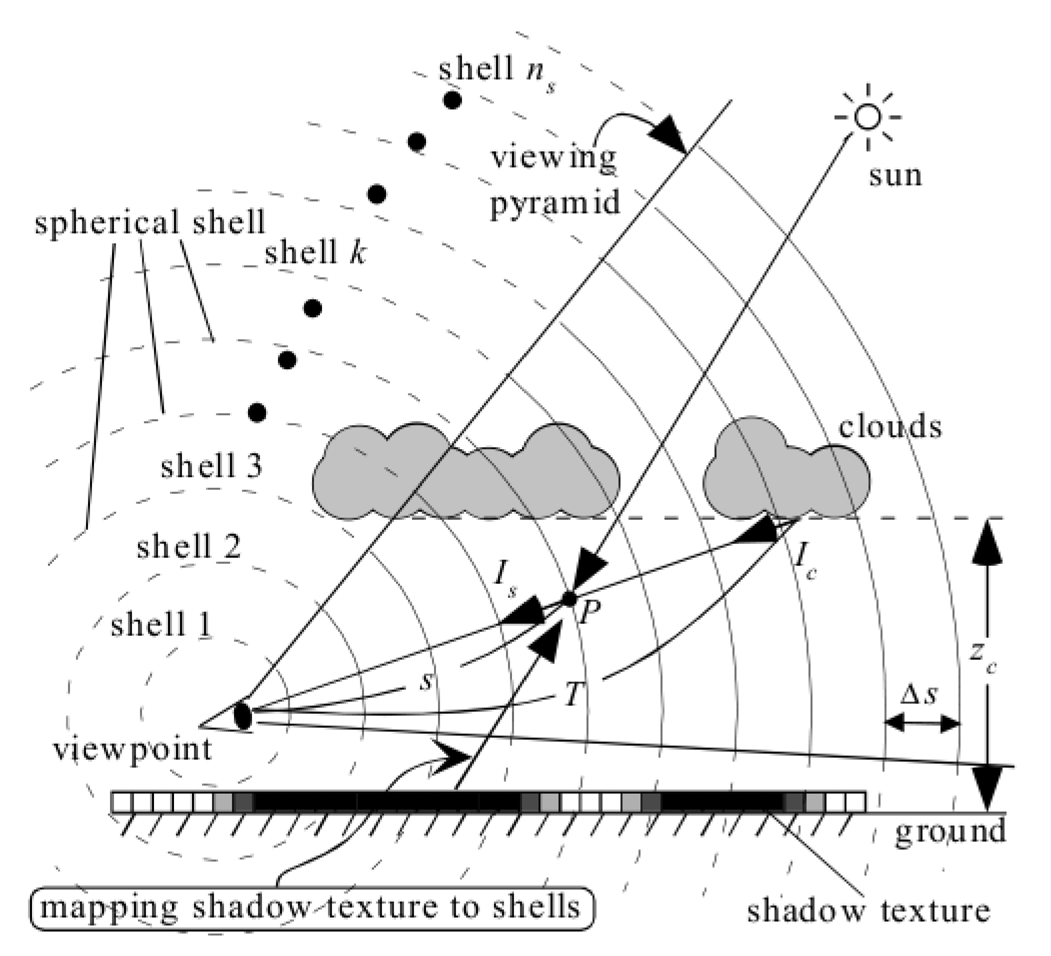
Step2: Image Generation



- Render Background
- Draw metaballs in increasing depth order using the color computed

Step3: Shafts of Light

- Scattering caused due to particles in atmosphere



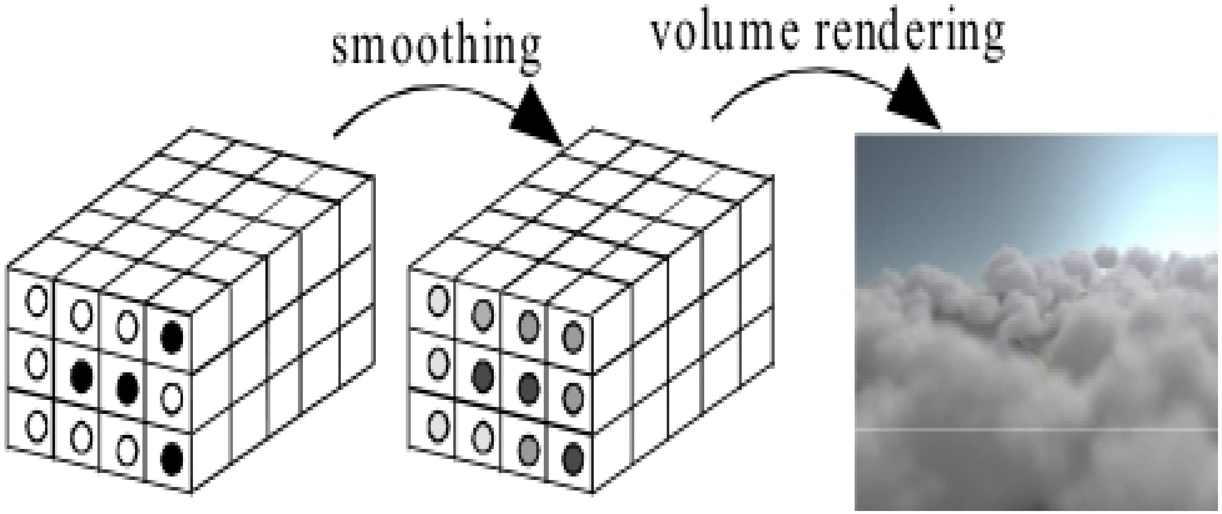
- Intensity actually reaching the view point:

$$I = I_c \beta(T) + \int_0^T \gamma(s) I_s(s) \beta(s) ds ,$$

- Particle density exponentially reduces with height. βs & I(s) analytically computed.
- Discretized Equation

$$I = I_c \beta(T) + \sum_{k=0}^{n_s} \gamma(k\Delta s) I_s(k\Delta s) \beta(k\Delta s) \Delta s ,$$

Results



Conclusion

- Advantages:
  - Simulation requires little computation
  - Memory requirements are small
  - Rendering is fast by making use of graphics hardware
  - Shadows of clouds and shafts of light can also be rendered
- Possible improvements:
  - Effects of terrain under clouds
  - Handle multiple wind direction, wind field

References

- Y. Dobashi, T. Nishita, T. Okita, “Animation of Clouds Using Cellular Automaton,” Proc. of Computer Graphics and Imaging’98, 1998, pp. 251-256
- Y. Dobashi, T. Nishita, H. Yamashita, T. Okita, “Using Metaballs to Modeling and Animate Clouds from Satellite Images,” The Visual Computer, Vol. 15, No. 9, 1998, pp.471-482.

THANK YOU !!