DOI: 10.3745/KIPSTA.2009.16A.5.307

비행 시뮬레이션을 위한 구름 모델링 및 렌더링

도 주 영[†] · 백 낙 훈[††] · 이 창 우[†††] · 유 관 우[††††]

요 약

컴퓨터 그래픽스에서 구름과 같은 대기 현상을 모델링하고 렌더링하는 것은 그 복잡성과 규모, 편재성 등으로 인해 상당히 까다로운 연구과제들 중의 하나이다. 본 논문은 컴퓨터 게임이나 항공 시뮬레이션 분야에서 요구되는, 실시간에 처리될 수 있는 구름 모델링과 렌더링 방법을 제안한다. 제안하는 방법은 사용자가 직관적이고 대화형 편집 과정을 거쳐 권운, 층운, 적운 등의 다양한 형태를 생성할 수 있다. 또한, 메타볼과 계층적 구형 파티클을 사용하여, 세부 묘사를 자동으로 추가할 수 있다. 생성된 파티클들은 다중 순방향 산란과 이방성 산란을 고려하여 빌보드 방식으로 출력함으로써 실시간 처리가 가능하다.

키워드 : 구름, 렌더링, 모델링, 비행 시뮬레이션, 메타볼, 대기현상, 자연현상, 계층적 구형 파티클, 실시간

Modeling and Rendering of Clouds for Real-time Flight Simulation

Joo-young Do[†] · Nakhoon Baek[††] · Chang-woo Lee[†††] · Kwan-woo Ryu[††††]

ABSTRACT

Modeling and rendering of atmospheric phenomena such as clouds is one of most difficult research themes in the field of computer graphics, mainly due to its complexity, huge volume, ubiquitousness, etc. In this paper, we represent a system for real-time modeling and rendering of clouds, mainly aiming at the computer games and flight simulation applications. Our implementation generates various kinds of clouds including cirrus, stratus, and cumulus, through intuitive real-timeuser interactions. Then, additional details are automatically attached to them, using our own methods based on meta-balls or hierarchical spherical particles. After processing multiple scattering and anisotropic scattering, resulting particles are rendered into billboards, to finally achieve real-time processing.

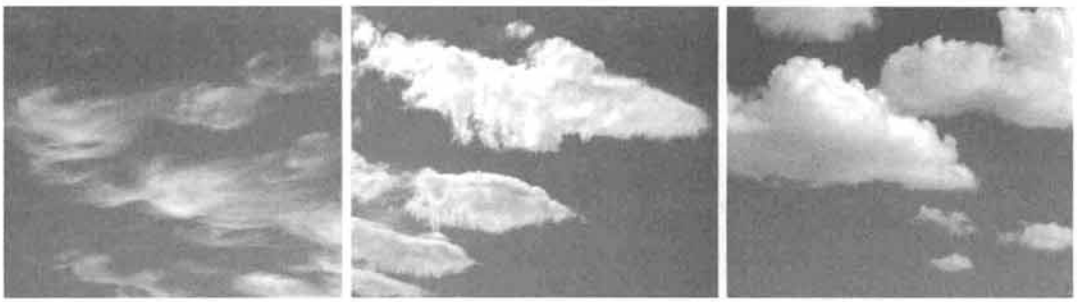
Keywords : Clouds, Rendering, Modeling, Flight Simulation, Metaball, Atmospheric Phenomena, Natural Phenomena, Hierarchical Particles, Real-Time

1. 서 론

구름은 자연스러운 야외장면의 생성에 꼭 필요한 요소로서 특히 컴퓨터 게임, 비행 시뮬레이션과 같은 실시간 응용 프로그램에서 사용자 몰입도를 높이는데 중요한 역할을 한다. 이에 따라, 사실적이고 부피감 있는 구름 주변을 날아다니거나 뚫고 지나가는 효과에 관한 요구가 꾸준히 증가하고 있다. 컴퓨터 그래픽스 분야에서 구름은 작은 입자들, 즉, 수증기와 공기로 구성되어 매우 무질서하고 복잡한 형태를 띄며 바람, 온도, 압력 등의 변화에 따라 지속적으로 그 형태가 변화하는 3 차원 볼륨으로 모델링 할 수 있다. 이러한 구름은 일정한 형태가 없고 경계가 모호한 경우가 많아 컴퓨터로 이를 자연스럽게 모델링 하기가 쉽지 않다. 또한, 자연현상에서와 같은 사실적인 구름을 모델링 하기 위해서는 구름 쉐이딩 효과도 함께 고려해야 한다. 구름을 구성하는 입자들은 상황에 따라 빛을 통과시키거나 산란시키는 특성을 가지므로, 주변상황이나 시점에 따라 변화된 구름의 색상을 계산하기 위해서는 많은 메모리 공간과 시간을 필요로 한다. 따라서 이제까지는 실시간으로 구름 모델을 렌더링 한 결과를 확인하기가 까다로웠다.

자연 현상에서 대기 속 수증기는 기온이 높아져 상승하면서 공기 중의 해염 입자나 미세먼지, 토양물질과 같은 부유물질들과 결합하여 응결핵(congelation)을 형성한다. 이러한 응결핵이 대기 중에서 과포화 상태가 되면 온도에 따라 물방울이 되거나, 승화해서 빙정(nucleate)으로 모여서 구름을

※ 이 논문은 2009년 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(과제번호 : 2009-0088544).
† 준 회 원 : 경북대학교 컴퓨터공학과 박사과정
†† 종신회원 : 경북대학교 전자전기컴퓨터학부 교수
††† 정 회 원 : ㈜케이오지 연구원
†††† 정 회 원 : 경북대학교 컴퓨터공학과 교수
논문접수 : 2007년 10월 10일
수 정 일 : 1차 2008년 6월 2일, 2차 2009년 6월 2일, 3차 2009년 7월 6일
심사완료 : 2009년 7월 14일



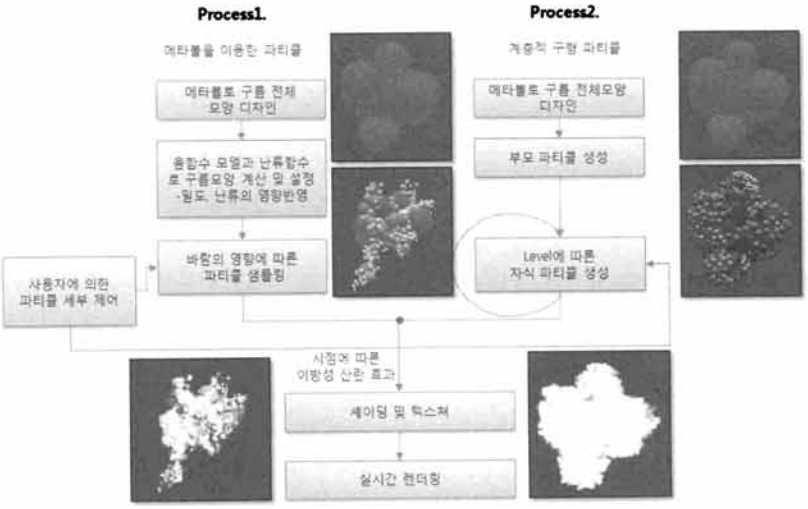
(a) 권운, 새털구름 (b) 층운, 안개구름 (c) 적운, 뭉게구름

(그림 1) 구름 사진

형성한다. 구름 종류나 성장단계는 생성되는 입자의 크기나 농도에 따라 달라진다. 급속히 생긴 구름은 입자의 크기가 거의 비슷하나 천천히 생긴 구름은 크고 작은 입자가 섞여서 폭넓게 분포되는 형태로 나타난다. 이러한 구름 모양을 운형이라고 하며 국제 구름 도감에서는 10종의 기본 운형으로 분류한다[1]. 크게는 구름이 발생하는 높이가 거의 정해져 있는 층상운(상층운, 중층운, 하층운 등)과 높이가 정해져 있지 않은 대류운(적운·적란운 등)으로 나뉜다. 이 중에서 구름이 가지는 외형적 특징을 두드러지게 잘 나타내고 있는 운형은 (그림 1)에 제시된 권운, 층운, 적운으로 3종류이다.

본 연구는 여러 가지 다양한 형태의 구름을 쉽고 빠르게 생성할 수 있고 모델링된 결과를 실시간 렌더링으로 즉시 확인할 수 있는 파티클 기반의 구름 생성 시스템을 제안한다. 모델링 되는 구름은 크게 권운(cirrus), 층운(stratus), 적운(cumulus) 세 가지 형태로 구분된다. 이러한 세 가지 구름을 입자 농도, 흩어짐 정도와 같은 직관적인 매개변수를 통해 쉽게 생성할 수 있도록 하고 음함수(implicit function)를 이용한 메타볼과 계층적 구형 파티클을 이용하여 구름의 전체적인 형태를 사용자가 원하는 대로 쉽게 조작할 수 있는 방법을 제공한다. 또한 모델링 된 구름을 파티클로 샘플링하여 빛의 투과, 다중 산란(multiple scattering) 효과 등을 고려한 셰이딩을 통해 자연스러운 구름 모습을 렌더링 할 수 있도록 하고, 이를 통해 실시간으로 모델링된 구름의 형태를 확인할 수 있도록 하여 구름 모델링을 편리하게 한다. 파티클 기반 렌더링 방법은 시점이 변할 경우 광원에 따른 구름의 다중 산란을 실시간으로 표현 할 수 있다. 이러한 특징들은 게임과 같은 실시간 그래픽스 분야에서 사용자가 원하는 디자인으로 구름을 제공할 수 있게 한다.

제안된 시스템에서는 구름 모델링을 위해 메타볼을 이용한 기법과 계층적 구형 파티클을 이용한 기법, 두 가지 방법을 사용한다. 제안하는 두 가지 구름 모델링 기법은 각기 다음과 같은 과정을 거친다. 먼저 첫번째 단계에서 구름이 가지는 전체적인 모양을 정하기 위해 사용자가 3차원 공간상에 메타볼을 배치하는 것은 두 방법에서 모두 동일하다. 두 번째 단계에서 메타볼을 이용한 기법은 원하는 구름 유형에 따라 매개변수를 설정하여 3차원 볼륨 데이터를 생성한 후, 이를 파티클로 샘플링 한다. 이때 샘플링 된 파티클에 바람에 따른 영향을 추가하여 좀더 흩어진 형태의 구름을 모델링할 수 있다. 계층적 구형 파티클 기법에서는 부모 파티클을 기반으로 각 단계별 자식 파티클을 생성한다. 사용자는 생성된 구름 파티클들을 직접 조작하여 구름의 형태를 세밀하게 조작할 수 있다. 마지막으로 각각의 방법으로 모델링이 끝난 구름은 여러 광원에 의한 다중 산란 효과를 반영하기 위해 빛과 시점이 이루는 각도에 의한 이방성 산란



(그림 2) 구름 모델링 및 렌더링 과정

(anisotropic scattering)을 고려하여 최종적인 구름 렌더링이 이루어지게 된다. 전체 알고리즘 구성도는 (그림 2)와 같다.

본 연구에서 제안한 방법은 적운뿐만 아니라 권운, 층운 등 다양한 형태의 구름을 메타볼과 계층적 구형 파티클을 사용하는 두 가지 모델링 프로세서를 통해 일반 사용자도 쉽고 빠르게 생성할 수 있도록 하고, 또한 이렇게 생성한 구름 모델을 컴퓨터 게임이나 가상현실, 컴퓨터 애니메이션 등에서 효과적으로 다룰 수 있도록 하는 데에 목적을 두어 실시간 처리에 초점을 맞추었다.

2. 관련 연구

이제까지 구름을 모델링하는 방법에는 물리기반 모델링 기법과 절차적 기법을 이용한 방법이 있어 왔다. 물리기반 기법들은 구름을 구성하는 입자들의 물리적 변화 과정을 추적하여 시뮬레이션 함으로써 정확하게 구름의 형태를 모델링 할 수 있다. 하지만 구름의 규모가 큰 경우에는 계산량이 많아 실시간에 모델링 결과를 확인하기 어렵다. 또한 물리적인 변수들의 조작을 통해 구름의 모양을 변경해야 하기 때문에 특정한 형태의 구름을 인위적으로 만드는 등의 조작은 기대하기 어려웠다. 이에 반해 절차적 기법의 경우, 상대적으로 빠른 모델링이 가능하고 직관적인 매개변수 값의 조절로 사용자가 쉽게 구름의 형태를 설정할 수 있다.

Ebert와 Schpok는 음함수 모델(implicit model)로 구름의 전체 형태를 모델링하고 난류(turbulence)와 노이즈(noise)에 의해 구름의 세부적인 모습을 모델링 할 수 있는 절차적 모델링 방법을 제안하였다[2, 3]. 이들은 슬라이스 기반의 볼륨 기법을 사용하여 시점이 변할 때 마다 매번 여러 장의 텍스쳐를 갱신하여야 하기 때문에, 실시간 렌더링이 어렵다는 단점이 있다. Nishita와 Harris는 보다 자연스럽고 사실적인 구름을 표현하기 위한 렌더링 방법에 더욱 중점을 두었다[4, 5]. Harris의 실시간 구름 렌더링 기법은 Nishita의 빛의 산란을 통한 쉐이딩 방법을 개선하여 여러 개의 방향성 광원에 의한 다중산란과 시점에 따른 이방성 산란을 쉐이딩에 적용하였다. 하지만 시점이 변할 때 마다 각 슬라이스 별로 광원과의 연산을 다시 해야 하기 때문에 시점의 변환이 자유롭지 못하다는 제약이 있다. 본 연구가 제안하는 시스템에서는 광원이나 시점의 변화에 따른 재계산을 피하기 위해, 모델링 된 볼륨 데이터를 메타볼이나 파티클 형태로 저장한다.

물리기반 모델링 연구에서는 구름과 같은 유체의 모델링과 애니메이션을 표현하기 위해 계산 유체역학 기법을 사용한 연구들도 있어 왔다. Kajiya는 광선 추적 알고리즘(ray tracing algorithm)을 적용한 구름 데이터를 생성하기 위해 편미분방정식(PDEs)을 사용한 간단한 유체 방정식 풀이를 적용하였다[6]. 하지만 사실적인 적운의 형태를 표현하지는 못하였다.

물리기반 연구에서 구름과 같이 많은 양의 데이터에 대해 병렬 처리가 가능한 GPU를 유체 방정식과 열역학 방정식들을 풀이하는데 활용하여 보다 빠른 구름 시뮬레이션을 가능하게 하였지만 여전히 많은 계산 시간을 필요로 하며 사용자가 구름의 형태를 직접적으로 모델링 할 수 없다는 제한이 있다.

Dobashi는 간단한 셀룰라 오토마타(cellular automata)를 이용하여 보다 빠르게 구름 형태와 움직임을 모델링하는 방법을 제안 하였다[7]. 하지만 구름 형성을 위한 물리적 과정을 상당히 단순화시켰기 때문에 복잡한 형태의 구름은 생성할 수 없고 적운만을 모델링 할 수 있다. 또한 3차원 볼륨 데이터를 저장하기 위해 많은 메모리 공간을 필요로 하고 여전히 많은 계산량을 필요로 한다. 이러한 물리 기반의 계산 유체역학 기법들은 많은 계산량을 요구하므로 실시간 적용이 불가능하다.

구름 렌더링 시스템에 관한 연구로는 Wang의 texture splatting 방법을 사용한 구름 렌더링과 Rana의 OpenGL API를 이용해 하드웨어 가속시킨 실시간 구름 렌더링 연구가 있다.[8, 9]. Rana의 연구에서 사용한 빌보드는 128개로 한정되어 있으며 최대 빌보드에서 최대 렌더링 속도인 401.65 fps를 기준으로 보았을때 본 연구와 비슷한 속도를 보이고 있다. 하지만 제안한 시스템에서는 3,000개 이상을 구성하여도 140 fps 속도로 안정적인 렌더링 결과를 보여주고 있다. 또한 렌더링된 구름의 모양에서도 큰 차이를 볼 수 있다.Wang의 방법은 구름 모델을 모델링할 때 다양한 크기의 직육면체 모델들에 구름 텍스쳐를 여러 개 겹쳐서 조합하여 쉽게 구름을 제어할 수 있게 하였다. 하지만 3D Studio MAX의 Plug-in을 통해 결과를 얻을 수 있다는 단점이 있다. Wang의 연구와 유사한 직육면체 큐브(cube)를 사용하여 2D 텍스쳐 매핑 기법을 사용하는 Rana의 모델링 방법은 빠른 렌더링을 제공하지만 텍스쳐의 제약에 따라서 다양한 형태의 구름을 나타낼 수 없다는 한계가 있다. 또한 렌더링된 구름의 모양에서도 큰 차이를 볼 수 있다. 본 연구의 시스템에서는 세 가지 형태의 적운, 층운, 권운과 함께 사용자가 원하는 디자인의 구름을 생성할 수 있도록 하는 반면, Rana의 연구와 Harris, Wang 등의 실시간 구름 모델링 방법에서는 구름의 모양이 적운의 형태로 제한되어 있다는 단점이 있다.

최근 구름에 관한 연구는 정적인 구름과 시뮬레이션을 위한 동적인 구름에 관한 영역으로 분류될 수 있다[10, 11, 12, 13]. 동적인 구름 모델은 보다 빠른 렌더링 시간을 목적으로 텍스쳐 매핑을 이용한 단순화된 구름의 표현만을 제공하여 사실성이 떨어진다. 또한 기존의 정적 구름에 관한 연구 결과들은 대부분 구름을 사실성이나 물리적 현상을 나타내는 연구에 집중해서 대부분의 경우 계산량이 많거나 사용자와의 상호작용이 어려워 원하는 구름 모델을 실시간에 확인할 수 없는 단점을 가지고 있다.

본 논문에서는 사용자가 원하는 다양한 형태의 정적인 구름을 모델링하면서 실시간 렌더링으로 빠르게 결과를 확인하고, 쉽게 수정할 수 있는 비행시뮬레이션을 위한 구름 시스템을 제안하고 있다.

3. 구름의 모델링

본 연구에서는 구름 모델링을 위해 두 가지 방법을 제안한다. 첫 번째 방법은 절차적 기법에 의해 모델링 된 볼륨 데이터를 적절한 크기의 파티클로 샘플링 함으로써 적운뿐만 아니라 권운, 층운 등도 일괄적으로 모델링이 가능한 방법으로 다양하고 자세한 구름을 생성할 수 있다. 두 번째는 계층구조를 이루는 파티클을 이용하는 방법으로 사용자의 제어를 최소화하면서 효과적으로 원하는 구름을 생성할 수 있다. 이 방법에서는 사용자의 입력을 최소로 하는 대신 파티클의 텍스쳐를 다르게 함으로써 다양한 모양의 구름을 나타낼 수 있다. 이들 각각을 차례로설명하겠다.

3.1 메타볼을 이용한 구름 모델링

Ebert는 구름의 전체 형태를 나타내는 음함수 모델과 세부 형태를 나타내는 난류와 노이즈 함수를 통해 구름 내부의 물 입자의 밀도를 지정하여 표현하고 이를 관측 평면(view plane)에 평행한 여러 장의 텍스쳐로 나누어 샘플링하여 이를 화면에 겹쳐 그리는 방식으로 구름을 렌더링 하였다[14]. 하지만 이 방법은 시점의 위치가 바뀔 때 마다 새로 텍스쳐를 갱신하여야 하는 단점이 있어 구름을 실시간으로 표현하기에 적합하지 않았고, 해상도가 증가하면 더많은 텍스쳐 메모리를 필요로 하는 단점이 있다. 본 연구에서는 Ebert의 방법에 기초하여, 음함수 모델로 전체적인 구름의 형태를 표현하되, 메타볼로 전체 형태를 설정하고, 이를 파티클로 샘플링하여 구름을 모델링 하는 방법을 제안한다. 이 방법은 여러장의 텍스쳐가 아니라, 파티클 단위의 처리를 수행함으로써, 적은 메모리 공간으로도 다중 산란 등의 다양한 렌더링 효과를 반영할 수 있는 장점이 있다.

우선, 구름의 전체 형태를 잡기 위해서는 Wyvill의 방법을 사용하여 음함수 모델의 밀도 혼합 함수를 다음과 같이 설정하였다[15, 16].

$$h_i(r)=\begin{cases}1-\frac{4}{9}\frac{r^6}{R^6}+\frac{17}{9}\frac{r^4}{R^4}-\frac{22}{9}\frac{r^2}{R^2}, & (r\le R),\\ 0, & (r>R)\cdot\end{cases} \tag{1}$$

여기서 R은 주어진 메타볼의 반지름이고, r은 메타볼의 중심에서 현재 위치까지의 거리이다. $r=0$일 때 $h(r)=1$이 되고, $r=R$일 때 $h(r)=0$이 된다. 이때 i는 i번째 생성된 메타볼을 나타낸다.

구름과 같이 경계가 곡선으로 이루어진 물체를 모델링하기 위해서는 가우시안 분포처럼 밀도 분포가 연속적으로 부드럽게 변해야 한다. 위의 밀도 혼합 함수의 장점은 수식이 간단하여 계산이 빠르고 가우시안 분포와 흡사하기 때문에 여러 원시 요소들 사이에 밀도가 부드럽게 혼합되어 원하는 형태의 구름을 만들기가 쉽다. 따라서 위의 밀도 혼합 음함수에 구름 덩어리의 크기를 설정하여 원하는 메타볼을 공간상에 배치하는 것만으로 간단하게 전체적인 구름의 형태를 지정할 수 있다. 또한 $r=R$부근에서 $h(r)$이 빠르게 0으로 수렴하기 때문에 경계가 비교적 뚜렷하다. 최종 볼륨 밀도 함수는 각 원시 음함수 요소의 가중치에 의한 합으로 다음과 같이 정의된다.

$$D_{implicit}(p)=\sum_i(w_i\cdot h_i(p-c_i)). \tag{2}$$

여기서 w_i는 i번째 원시 요소의 가중치이고 c_i는 원시 요소의 중심 위치이다. 위 식을 통해 최종 볼륨 밀도 함수로 구름의 전체 형태가 결정되면 구름의 세부 형태를 결정하기 위해 다음과 같이 난류(turbulence)함수를 사용해서불규칙적인 밀도 분포를 표현한다.

$$D_{turbulence}(p)=(\alpha\times turb(p)) \tag{3}$$

위 식에서 $turb(p)$는 난류 함수를 나타내며, 0에서 1까지의 값을 가진다. α는 난류의 양(amount)을 나타내는 상수값이다. 난류 함수는 여러 옥타브(octave)의 노이즈(perlin noise) 함수를 삼선형 보간법(trilinear interpolation)으로 합성하여 다음 식과 같이 생성한다. 이 보간법을 사용하는 노이즈 함수는 선형 보간을 사용하는 백색 노이즈(white noise)보다 연속적인 값의 분포를 가지는 특징이 있어 보다 구름과 같은 부드러운 형태를표현하기에 적합하다 [14].

$$turb(p)=\sum_{j=0}^{n}S_{noise}(\frac{x}{O_c^j},\frac{y}{O_c^j},\frac{z}{O_c^j})\cdot O_c^j \tag{4}$$

위 식에서 O_c는 각 옥타브의 수를 나타내고, j개의 각 옥타브의 노이즈 함수를 더하여생성된 $turb(p)$의 범위는 0에서 1사이의 값이 된다. S_{noise} 함수는 삼선형보간(trilinear-interpolation)을 의미한다.

Ebert[15]는 아래의 식에서와 같이 메타볼에 대한 음함수 모델과 난류함수가 혼합되는 비율인 blend 값과 밀도의 변동(wispiness) 값인 wisp 값을 조절하여 완전히 음함수 모델에 의해 만들어지는 고형의 구름 형태로부터 완전히 난류함수에 의해 생성된 불규칙한 형태의 구름까지 생성할 수 있도록 하였다.

$$D(p)=\{(1-blend)D_{implicit}+blend\cdot D_{turbulence}\}^{wisp} \tag{5}$$

하지만 본 연구에서의 실험결과로는 규모가 큰 구름을 위해 메타볼의 크기를 크게 모델링 할 경우, 난류 함수만으로는 불규칙한 구름의 형태를 생성하기가 곤란하다는 점이 부각되었다. 따라서 본 연구에서는 Ebert의 방법으로 모델링한 후 파티클로 샘플링하는 단계에서 바람의 영향을 추가하였다. 파티클의 흩어진 분포를 위해 파티클에 작용하는 힘에 항력 방정식(Stroke's drag equation)을 사용한다. 공기 저항

도 점성 저항과 같이 속도에 비례하므로 다음과 같이 구할 수 있다.

$$\vec{F}_{wind} = \beta\vec{v} \tag{6}$$

식(6)에서 $\vec{F}_{wind}$는 파티클에 작용하는 힘을 나타내고, β는 사용자가 설정하는 바람의 세기값이며, $\vec{v}$는 바람의 방향을 설정하는 단위 벡터이다.

이런 방식으로 생성된 구름의 볼륨 데이터를 렌더링 하기 위해서는 파티클로 샘플링하는 방법을 사용한다. 파티클 시스템에서 파티클을 샘플링 할 때는 두 가지 방법을 사용할 수 있다. 하나는 무작위 샘플링(non-uniform sampling) 방법이고 다른 하나는 균일 샘플링(uniform sampling) 방법이다[16]. 무작위 샘플링 방법은 빠르고 간편하지만 볼륨 밀도 데이터의 특징을 정확하게 추출하기 어려운 단점이 있다. 반면, 균일 샘플링 방법은 샘플링 시간이 오래 걸리고 상대적으로 많은 수의 파티클을 필요로 하지만 볼륨 데이터를 비교적 정확하게 샘플링 할 수 있어서 자세한 구름의 형태를 표현하기에 적합하다. 따라서 정확한 구름 모델을 위해서 균일 샘플링 방법을 선택하고, 다음과 같은 과정을 거친다.

1. 지정된 해상도의 3 차원 복셀의 각 위치에서 볼륨 밀도 데이터를 균일하게 샘플링 한다.
2. 샘플링 된 데이터를 탐색하여 이웃 파티클 사이의 밀도가 비슷한 경우에 이들 파티클을 합쳐서 하나의 큰 파티클로 대체한다.
3. 파티클을 더 이상 병합할 수 없을 때까지 2의 과정을 반복한다.

아래의 (그림 3)은 한 개의 메타볼에서 생성된 파티클들의 밀도분포를 나타낸다.

Ebert의 음함수 모델을 수정하여 최종적으로 모델링 되는 메타볼의 볼륨 밀도와 이에 따른 파티클의 생성 순서는 다음과 같은 처리를 거친다.

Procedurecompute density of metaballs

Compute the implicit density, $D_{implicit}(p)$.

Compute the turbulence density, $D_{turbulence}(p)$.

Calculate the blended density, $D(p) = \{(1-blend)D_{implicit} + blend \cdot D_{turbulence}\}^{wisp}$

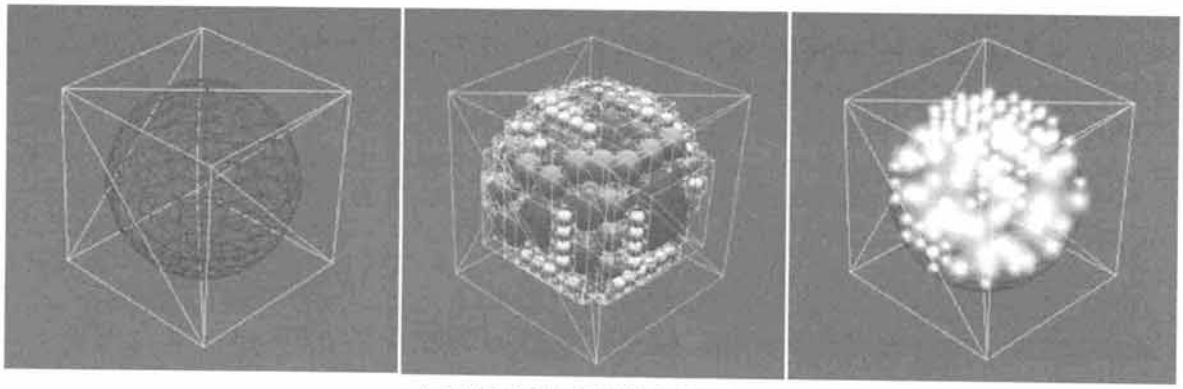
return density

절차적 난류함수에 의해 생성된 불규칙한 밀도 분포를 이용하는 모델링 방법은 권운이나 층운, 적운과 같은 다양한 형태를 표현할 수 있으며, 사용자에게 밀도 절대치와 밀도 분포 변화를 위한 매개변수들을 제공한다. 권운의 경우 높은 고도의 구름으로 밀도가 낮고 입자들의 층이 얇으며 많이 흩어져 있어 뚜렷한 형체를 이루지 않고 새털처럼 보인다. 그리고 층운은 권운에 비해 밀도가 높고 비교적 입자들이 뭉쳐있지만, 적운에 비해서는 상대적으로 밀도가 낮고 입자들이 흩어져 있다. 이러한 형태의 구름은 wispiness를 조절함과 동시에 난류함수의 크기를 조절 함으로써 부드럽게 분포된 구름을 생성할 수 있다.

3.2 계층적 구형 파티클을 이용한 구름 모델링

계층적 구형 파티클(hierarchy of spherical particles)을 이용한 구름 모델링은 파티클들 사이의 계층 구조를 이용한 모델링 방법이다. 앞서 3.1절의 메타볼을 이용한 구름 모델링 방법에서는 사용자의 요구에 따라 각 매개변수 값을 세밀하게 조절하여 사용자가 원하는 모양의 구름을 모델링하는 것에 중점을 두었다. 반면에 계층적 구형 파티클을 이용하는 방법은 사용자와의 상호작용을 최소화하면서 빠르게 구름의 형태를 생성하고 렌더링 된 결과를 확인할 수 있도록 한다.

구름의 전체적인 형태는 계층 구조의 최상위에 있는 씨앗 파티클(seed particle)에 의해 정해지고, 이 씨앗 파티클을 중심으로 그 표면에 여러 계층의 파티클 집합을 생성하여 구름의 세부적인 모습을 표현한다. 즉, 구름을 이루는 파티클 집합은 계층적 구조를 이루어 하나의 파티클 표면에 다음 단계의 파티클 집합이 붙어서 배치되고, 그 위에 다시 다음 단계의 파티클이 붙어 있는 형태이다. 각 파티클들은



(그림 3) 음함수 모델의 파티클 생성

중심 위치와 반발 반경(repulsionn radius), 그리고 실제 화면에 렌더링될 때의 파티클 반경을 가지며 두 파티클의 중심 사이의 거리는 두 파티클의 반발 반경의 합에 비례한다. 파티클의 계층 구조를 생성하는 과정은 다음과 같다[16].

1. $i-1$번째 단계의 파티클 P_{i-1}의 표면 위에 임의로 i번째 단계의 파티클 $P^0{}_i$를 생성한다.
2. 생성된 파티클 $P^0{}_i$(0번째P_i)를 중심으로 그에 접하는 i번째 단계의 이웃 파티클 P_i^j를 생성한다.
3. 생성된 이웃 파티클 P_i^j를 중심으로 반발 반경을 고려하여 파티클 P_{i-1}의 표면에 더 이상 파티클을 배치할 수 없을 때까지 과정 2를 반복하면서 i번째 단계의 파티클 집합을 생성한다.
4. i번째 단계의 파티클들을 모두 생성한 후 i번째 단계의 각각의 파티클을 중심으로 다시 과정 1을 반복하여 $i+1$번째 단계의 파티클 집합을 생성한다.

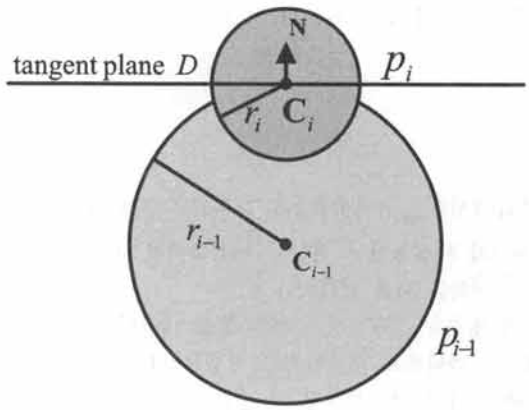
(그림 4)는 $i-1$번째 단계의 파티클 P_{i-1}의 표면에 임의로 i번째 단계의 파티클 P_i를 생성하는 과정을 나타낸다. 이때, N은 파티클 P_{i-1}의 노말 벡터이고, i번째 단계의 파티클의 반경 r_i는 다음과 같이 결정한다.

$$r_i = m_i \times (r_{i-1} + d_i \times rand) \tag{7}$$

먼저 사용자가 정한 파티클 반경편차 d_i에 실수화 랜덤값($0 \le rand \le 1$)을 곱한 후, 이전 파티클의 반경 r_{i-1}를 더한다. 그리고 파티클의 반경 비율 m_i를 추가로 곱하면 파티클 r_i의 크기가 결정된다. 이때 m_i와 d_i는 사용자에 의해 입력된다.

i번째 단계의 자식 파티클의 반발 반경(repulsion radius)도 같은 방법으로 계산된다. 파티클 P_i가 생성되면 (그림 4)과 같이 P_i의 중심 C_i에서 P_{i-1}의 접평면(tangent plane) 위에서 P_i를 중심으로 이웃하는 파티클 P_i^j를 생성하게 된다.

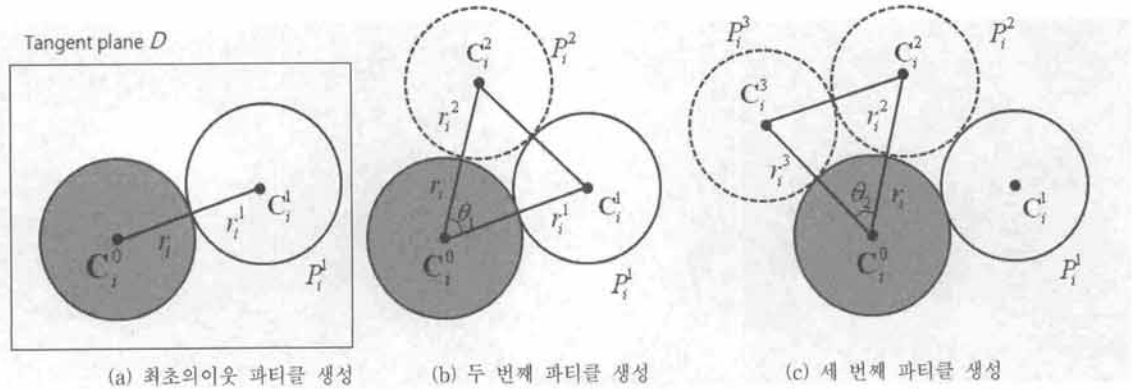
(그림 5)(a)에서 보듯이 파티클의 반경 비율 m_i와 파티클



(그림 4) 최초 자식 파티클의 생성

반경편차 d_i에 의해 첫 번째 이웃 파티클 P_i^1의 반지름 r_i^1를 결정한 후 이를 이용해 파티클 P_i^1의 중심 위치 C_i^1를 계산한다. 다음으로 두 번째 이웃 파티클 P_i^2의 반지름 r_i^2를 정한 후, r_i와 r_i^1을 이용하여 (그림 5)(b)에서 보듯이 세 구의 중심이 이루는 삼각형의 한 꼭지각 θ_1을 헤론의 공식을 이용하여 계산함으로써 두 번째 이웃 파티클 P_i^2의 중심위치 C_i^2를 계산한다. (그림 5)(c)에서 세 번째 이웃 파티클 역시 이와 같은 방법으로 생성하여 더 이상 이웃하는 파티클을 만들 수 없을 때까지 계속하여 이웃 파티클 집합을 생성하게 된다.

이때 생성된 이웃 파티클들은접평면 D위에서 만들어지기 때문에 그 중심 위치가 $i-1$단계의 파티클 P_{i-1}의 표면에 있지 않기 때문에 이를 보정하여 주어야 한다. (그림 6)와 같이 파티클 P_i의 중심 위치 C_i에서 파티클 P_{i-1}의 법선 벡터 N과 C_i에서 C_i^1로 향하는 접선 벡터 T로 이루어진 평면상에 C_i를 중심으로 반지름이 $r_i + r_i^1$인 원을 따라 파티클 P_i^1의 중심 C_i^1을 새로운 위치 $\hat{C}_i^1$로 이동시킨다. 이때 다음 수식에 따라 시계방향으로 회전하며 $\hat{C}_i^1$의 보정 위치를 계산한다. 따라서 보정 이동된 위치 $\hat{C}_i^1$는 이동하기 이전의 P_i^1의 x축, y축 좌표에서 각각 접선 벡터 T와 노말 벡터 N



(a) 최초의이웃 파티클 생성 (b) 두 번째 파티클 생성 (c) 세 번째 파티클 생성

(그림 5) 이웃 파티클의 생성 과정

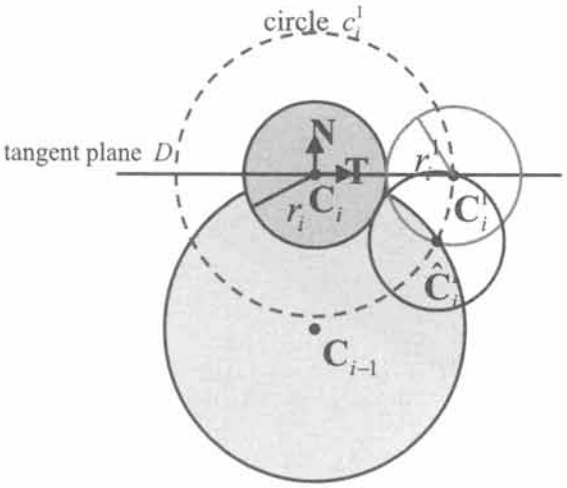
의 내적으로 구한다.

$$\hat{C}_i^j(x,y) = p_x \cdot \vec{T}_x + p_y \cdot \vec{N} \quad (8)$$

각 단계의 파티클들은 이와 같이 이전 단계의 파티클의 표면에 그 중심이 위치하고 서로 이웃하는 파티클 사이의 거리가 반발 반경에 비례하여 배치된다. 파티클 간의 반발 반경에 포함되는 파티클이 있는지 빠르게 검색하기 위해 옥트리를 이용한 공간 분할 구조(space partitioning structure)를 사용하여 생성되는 모든 파티클들을 공간에 따라 분할하여 저장한다. 이 방법에서도 음함수를 이용해 구름의 전체 형태를 지정하는 것과 같이 메타볼을 이용하여 씨앗 파티클을 배치함으로써 구름의 전체적인 형태를 모델링 한다. 파티클의 수와 분포와 크기를 조절하기 위해 모델링에 사용된 구에 비례하는 씨앗 파티클의 크기(size), 각 단계별 파티클의 반발 반경의 변화 비율(gradual ratio)과 표준 편차(standard deviation), 파티클의 실제 렌더링 반경의 변화 비율과 표준 편차를 사용자가 지정할 수 있다. 반발 반경이 파티클 반경보다 큰 경우에는 듬성듬성한 파티클의 분포를 생성하여 권운과 층운의 형태를 디자인 할 수 있고, 반발 반경이 파티클 반경보다 작은 경우에는 파티클들이 서로 겹쳐져 두루뭉술한 뭉게구름인 적운을 만들 수 있다. 이러한 계층적 구형 파티클 기반의 모델링 방법은 사용자의 입력을 최소화 하면서 간단하고 빠르게 구름의 모양을 모델링 하고 렌더링한 결과를 실시간으로 게임 등에 적용하고자 할 때 효과적으로 사용될 수 있다.

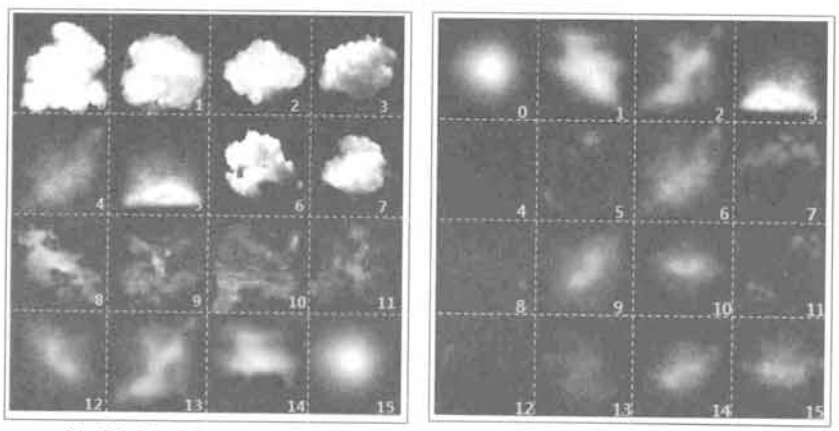
3.3 구름 텍스쳐와 렌더링

파티클을 이용한 구름 렌더링은 간단하면서도 효과적으로 볼륨 렌더링과 같은 입체적인 셰이딩 효과를 나타낼 수 있는 방법이다. 본 논문에서 사용된 구름 파티클의 셰이딩 방법은 하나의 광원으로부터 입사한 빛이 다른 입자들에 의해 산란되는 다중 순방향 산란(multiple forward scattering)과 뷰 포인트에서 도달하는 빛의 양을 계산한 이방성 일차 산란(anisotropic first-order scattering)을 계산한 Harris[5, 17]의 방법과 유사하다. 차이점은 Harris는 가우시안(gaussian) 텍스쳐를 사용하여 상대적으로 흐릿한 형태의 적운만을 렌더링 하였고, 또한 다중 순방향 산란 계산 방식은 모든 파티클에 대해 하나의 광원당 한번씩 계산된 광원에 대한 결과를 저장하고, 렌더링 시점에 다시 저장된 결과를 호출하므로 많은 시간이 소요된다. 따라서 이 과정을 전처리(preprocess)로 수행해야만 했다.



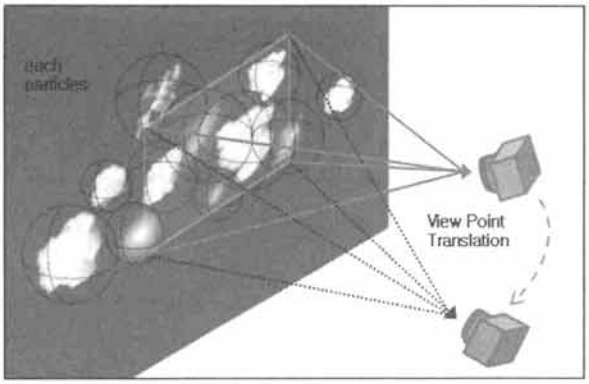
(그림 6) 부모 파티클 표면으로 자식 파티클 중심을 투사하는 과정

현재의 시스템에서는 파티클마다 사용자에 의해 정교하게 만들어진 서로 다른 형태의 파티클 텍스쳐를 사용할 수 있도록 하고, 파티클의 크기에 관계없이 2D 텍스쳐의 인덱스 넘버에 따라 랜덤(random)하게 매핑시킨다. 이러한 텍스쳐 매핑 방식은Niniane Wang의 적운 구현에서 사용되었다[8]. 텍스쳐는 (그림 7)에서 보듯이 메타볼을 이용한 기법에서 사용하는 텍스쳐와 계층적 구형 파티클에서 사용하는 텍스쳐로 구분하여 사용하였다. 그리고 각 텍스쳐는 256 x 256 사이즈로 color와 alpha값을 가지는 32bit 이미지이다. 텍스쳐 매핑을 할 때는 (그림 8)에서 보듯이 각 파티클에 대해



(a) 메타볼을 이용한 기법에서의 텍스쳐 (b) 계층적 구형 파티클에서의 텍스쳐

(그림 7) 구름 텍스쳐의 예



(그림 8) 텍스쳐의 빌보드

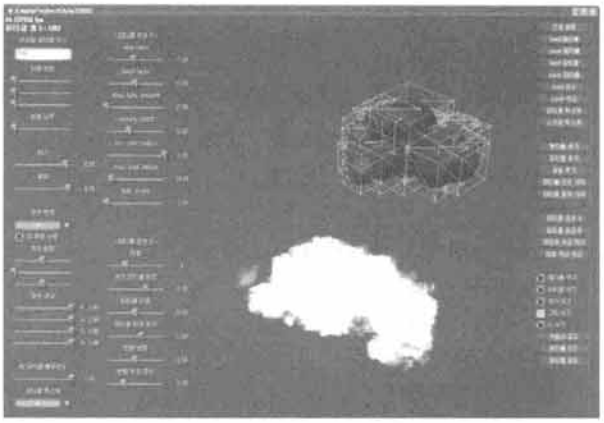
DirectX의 빌보드를 이용한다. 텍스쳐의 정면이 항상 카메라를 향하도록 함으로써 빠른 렌더링 속도를얻을 수 있고, 카메라의 시점이 바뀌더라도 구름의 전체적인 모양이 유지될 수 있다.

텍스쳐 매핑 후 광원에 의한 다중 순방향 산란의 계산을 수행하기 위해서 우선 텍스쳐 매핑된 파티클들을 광원의 위치로부터 떨어진 거리에 따라 배열로 저장한다. 그리고 광원에서부터 각 파티클의 중심 위치까지 도달하는 빛의 방향으로 근사화(approximation)된 빛의 산란을 계산한다. 즉, 광원에서부터 가까운 파티클이 먼저 렌더링되는데, 이때 렌더링 결과를 보다 빠르게 확인할 수 있도록 파티클을 개수별로 묶어서 그룹 단위로 근사화함으로써 실시간 렌더링을 제공할 수 있게 한다. 이처럼 각 파티클들은 광원으로부터 도달하는 빛의 양을 계산하여 파티클 위치와 사이즈, 색상, 알베도, alpha 요소 등에대한 데이터와 함께 저장한다.

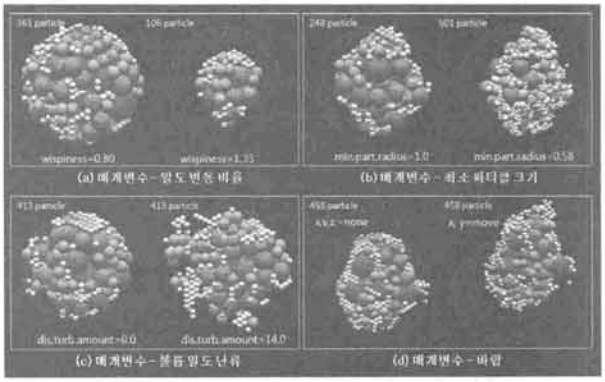
그리고 렌더링 시에카메라에 최종적으로 도달하는 빛의 양을 계산하기 위해 이방성 일차 산란 고려해야 한다. 이것은 주어진 입사 빛의 방향에 대해 산란되는 빛의 분포를 계산하는 것으로 파티클의 위치와 카메라의 시점 사이의 벡터로 정의되고, 파티클에 의해 산란된 빛의 양과 투과된 빛의 양을 합한 것으로 계산된다.

따라서 본 논문의 결과에서는 다중 순방향 산란과 이방성 일차 산란을 통해 적운뿐만 아니라 권운, 층운과 같은 다른 유형의 구름도 나타낼 수 있으며, 보다 자연스러운 구름의 경계를 나타낼 수 있도록 하였다. 또한 다중 순방향 산란과 함께 이방성 일차 산란을 근사하여 빛을 정면을 바라보았을 때 구름의 가장자리가 밝게 빛나는 실버-라이닝 (silver lining)과 같은 효과를 처리하기 위해, DirectX의 하드웨어 블렌딩(hardware blending), 렌더러블 텍스쳐(renderable texture), 프레임 버퍼 다시읽기(frame buffer read back) 기능을 사용하여 구현 했다.

(그림 9)는 메타볼을 이용한 파티클 생성 기법으로 나타낸 구름 모델을 본 연구에서 제안하는 방법으로 구현한 에디터 시스템에서 실행한 화면이다. 사용자는 그림과 같이 구를 배치하여 구름의 형태를 지정하고 원하는 구름 형태에 따라 매개변수를 지정한 후 파티클을 생성할 수 있다. 아래의 (그림 10)은 메타볼을 이용한 파티클 생성 기법에서 한



(그림 9) 구름 에디터 실행화면



(그림 10) 매개변수에 따른 구름 파티클의 변화

개의 메타볼을 이용해 각 매개변수를 변화시켰을 때 생성되는 파티클의 모습을 비교한 것이다.

4. 실험 결과 및 분석

아래의 그림들은 펜티엄 IV 3.2GHz, 시스템 메모리 1GB와 NVIDIA GeForce 6600의 그래픽 카드를 장착한 시스템에서 본 논문에서 사용한 두 가지 방법으로 모델링 된 결과물이다. (그림 11)은 음함수 모델과 절차적 기법을 기반으로 하여 메타볼을 이용한 파티클 생성 기법으로 처리된 결과이고, (그림 12)은 계층적 구형 파티클 생성 기법의 결과이다. 두 프로세스 모두 권운, 층운, 적운 형태의 구름을 모델링 할 수 있다. 적운은 가장 두꺼운 파티클 층을 바탕으로 선명한 구름 파티클 텍스쳐를 사용하여 모델링 되었고, 층운은 이보다 얇은 파티클 층을 이루며 경계가 적운에 비해 모호하고 파티클이 적운에 비해 흩어져 있도록 모델링 되었다. 권운은 상대적으로 가장 얇은 파티클 층을 가지며 파티클이 가장 넓게 흩어져 있다. 또한 각 파티클의 투명도가 높고, 사용되는 구름 파티클 텍스쳐도 흐릿한 경계를 가지는 것만을 사용하도록 하였다.

(그림 11) 에서 사용된 파티클의 수는 800~1,400개이며, (그림 7)에서와 같은 16개의 서로 다른 구름 파티클 텍스쳐를 각 파티클 마다 무작위로 지정하여 렌더링 하였다. 아래

(a) 적운의 생성 (왼쪽:메타볼, 중간:파티클 오른쪽:파티클에 구름 텍스쳐를 입힌 모습)

(b) 층운의 생성

(c) 권운의 생성

(그림 11) 메타볼을 이용한 파티클 생성 기법 결과

(a) 적운의 생성

(b) 층운의 생성

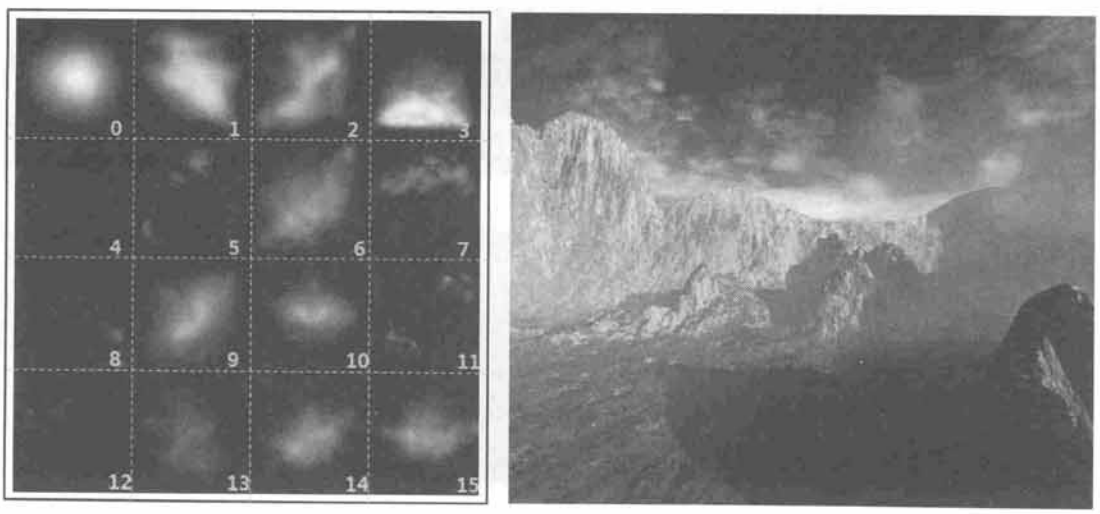
(c) 권운의 생성

(그림 12) 계층적 구형 파티클 기법의 결과

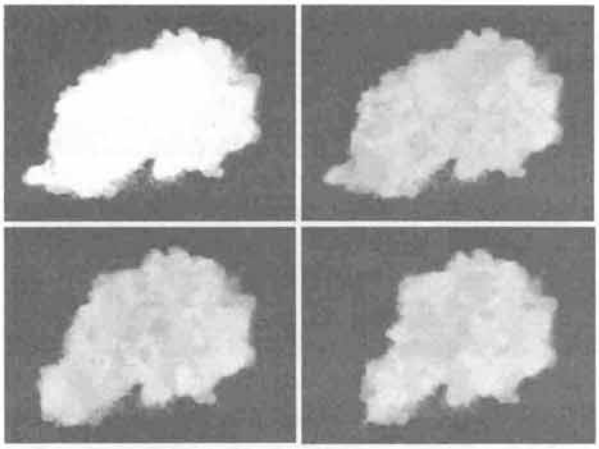
의 (그림 12)는 파티클 계층구조를 이용한 모델링 방법에 의해 생성된 구름의 모습을 나타낸다. 사용된 파티클의 수는 100~1,000개이며 층운과 권운을 위해 그림 13의 파티클 텍스쳐를 랜덤하게 사용하였다. (그림 12)(a)의 적운은 파티클 들이 서로 뭉쳐 있어 경계가 분명하기 때문에 그 전체적인 형태가 사용자가 음함수 모델을 통해 지정한 형태를 크게 벗어나지 않는다. 그렇기 때문에 이와 같이 파티클의 표면에 다시 파티클을 배치하는 방식으로 적운을 모델링하는 것이 가능하다.

(그림 12)(a)의 결과는 14개의 씨앗 파티클을 배치하여 전체 구름 형태를 지정하고 2 단계까지의 파티클 계층구조를 생성하도록 하여 얻은 것이다. 이때 반발 반경보다 파티클 반경을 크게 하여 파티클들이 서로 겹치도록 함으로써 뭉게구름과 같은 모습을 나타내도록 하였다. 사용된 파티클의 수는 940개이다. (그림 12)(b)와 (c)는 층운과 권운을 나타내기 위해 적운에 비해 상대적으로 적은 수의 메타볼을 이용해 2단계까지 파티클 계층구조를 생성하였으며, 각 파티클의 수는 층운은 193개, 권운은 128개이다.

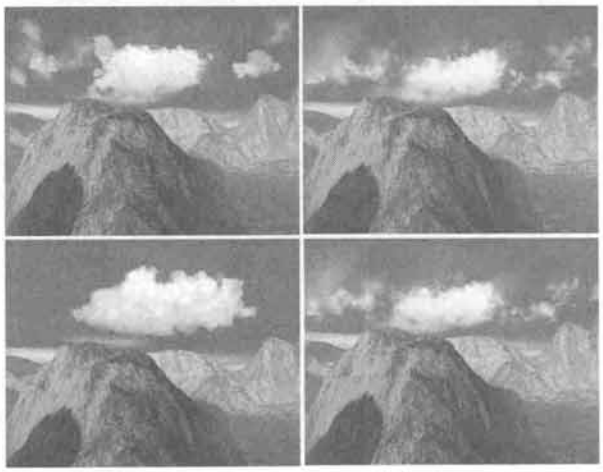
(그림 13)은 계층적구형 파티클을 이용하여 권운 모양의 구름을 생성하기 위해 적운에 비해 상대적으로 적은 108개의 파티클을 사용하였다. 파티클에 매핑되는 텍스쳐 또한 권운의 형태를 표현하기 적합한 형태로 바꾸었다. (그림 14)는 메타볼을 이용한 파티클 생성 기법으로 적운을 생성하고, 미리 계산된 색상값에 따른 변화를 보여주고 있다. (그림 15)(a)는 메타볼을 이용한 파티클 생성 방법을 사용하여 렌더링한 결과이고 (b)는 계층적 구형 파티클 방법을 사용한



(그림 13) 구름 텍스쳐와 계층적 구형 파티클을 이용한 권운



(그림 14) 적운(2,568 particle)의 색상 조절



(a) (b)

(그림 15) 다양한 형태의 구름 렌더링(2,449 particle)

〈표 1〉 파티클 개수와 렌더링 속도

파티클 개수	렌더링 속도 (fps)
100	400.23
1,000	250.61
2,000	200.80
3,000	140.00

결과이다.

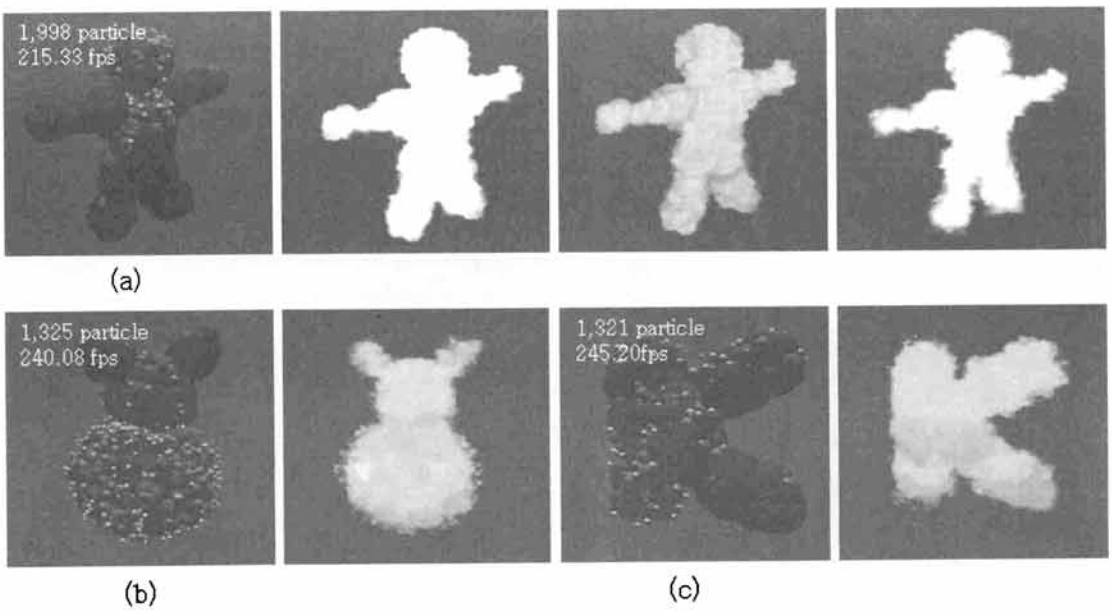
렌더링 속도는 파티클의 개수와 화면에 투사된 파티클의 크기에 따라 크게 좌우되는데 본 시스템에서는 약 800~1,400개의 파티클을 렌더링하는데 40~110fps의 렌더링 속도를 얻을 수 있었다. 본 시스템에서는 파티클의 최소, 최대 크기와 생성될 파티클의 개수를 지정함으로써 렌더링 속도와 렌더링 결과 사이의 접점을 사용자가 조절할 수 있도록 하였다.

<표 1>은 파티클의 개수에 따른 렌더링 속도를 측정한 것이다. 이때 렌더링 되는 화면의 해상도는 1,280 x 1,024이었다.

(그림 16)은 사용자가 원하는 형태의 구름 모양을 디자인하기 위해서 계층적 구형 파티클생성 기법으로 나타낸 결과이다. (그림 16)(a)는 사람 모형으로 1,998개의 파티클을 사용하였고, (b)는 눈사람 형태로 1,325개, (c)는 영문자 K를 나타내며 1,321개의 파티클을 사용하였다.

5. 결 론

본 연구에서 제안한 방법은 적운뿐만 아니라 권운, 층운 등 다양한 형태의 구름을 프로그래머가 아닌 일반 사용자도



(그림 16) 다양한 모양의 구름 디자인

쉽고 빠르게 생성할 수 있는 모델링 및 렌더링 시스템을 제안하였다. 사용자는 공간상에 메타볼을 배치하여 구름의 전체 구조를 지정하고 음함수 모델과 매개변수를 이용하거나 또는 계층적 구형 파티클을 이용하여 구름을 실시간에 생성할 수 있다. 또한 전처리로 수행되는 다중 산란 및 이방성 산란을 고려한 셰이딩을 파티클 기반 렌더링으로 수행함으로써 시점이 바뀔 때 마다 광원과의 연산을 재계산해야하는 부담을 덜었다. 따라서 자연스러운 구름의 모습을 모델링하고, 렌더링 된 구름의 형태를 실시간으로 확인할 수 있기 때문에 사용자와의 상호작용이 간편하다. 기존의 볼륨 슬라이스를 사용한 방법에 비하여 파티클을 사용하기 때문에 시점의 변화에 따른 추가적인 부하가 없으며 실시간으로 다중 산란 및 이방성 산란 등을 고려한 렌더링 기법을 사용할 수 있다. 따라서 실시간 처리가 요구되는 컴퓨터 게임이나 항공 훈련 시뮬레이션, 모바일 환경 등의 응용 프로그램에 적용될 수 있다.

향후 연구과제로서 정적인 구름 모델링을 개선하여 렌즈구름, 회오리 구름 등과 같은 다양하고 특수한 형태의 구름을 모델링 할 수 있고, 또한 학, 호랑이 등과 같이 보다 명확하고 복잡한 형태의 구름을 쉽게 생성할 수 있는 시스템을 개발하고자 한다. 또한 정적인 구름뿐만 아니라 사용자가 원하는 형태와 움직임으로 자연스럽게 변하는 구름 동영상을 생성할 수 있는 구름 애니메이션 시스템도 개발하여 사용자가 지정한 경로를 그대로 따라가면서 주변환경의 변화, 즉, 물체와의 충돌, 바람, 온도변화 등에 의해 시뮬레이션 되어 보다 자연스러운 구름의 움직임을 생성할 수 있도록 하고자 한다.

참 고 문 헌

[1] Ninomiya, Kozo, 대기과학을 위한 물리, 서울 : 동화기술, 2002.

[2] David S. Ebert, "Volumetric modeling with implicit functions: A cloud is born," In Visual Proceedings of ACM SIGGRAPH 1997, *Computer Graphics Proceedings*, pp.147, 1997.

[3] Schpok J. Simons J. Ebert D.S, Hansen C., "A real-time cloud modeling, rendering, and animation system," *Symposium on Computer Animation'03*, pp.160-166, 2003.

[4] T. Nishita, Y. Dobashi, E. Nakamae, "Display of Clouds Taking into Account Multiple Anisotropic Scattering and Sky Light, In *Proceedings of ACM SIGGRAPH 96*, *Computer Graphics Proceedings*, pp.379-386, 1996.

[5] Mark J. Harris, William V. Baxter III, Thorsten Scheuermann, Anselmo Lastra, "Simulation of Cloud Dynamics on Graphics Hardware," *Proceedings of Graphics Hardware*, 2003.

[6] Kajiya, J., von Herzen, B., "Ray Tracing Volume Densities," *Computer Graphics, SIGGRAPH '84 Conference Proceedings*, 1984.

[7] Y. Dobashi, K. Kaneda, H. Yamashita, T. Okita, and T. Nishita. "A Simple, Efficient Method for Realistic Animation of Clouds," In *Proceedings of SIGGRAPH 2000, Computer Graphics Proceedings*, pp.19-28, 2000.

[8] Niniane Wang, Realistic and Fast Cloud Rendering, *Journal of graphics tools*, pp.9(3):21-40, 2004.

[9] M.A. Rana, M.S. Sunar, M.Nor Hayat, Framework for Real Time Cloud Rendering, In *CGIV '04: Proceedings of the International Conference on Computer Graphics, Imaging and Visualization*, pp.56-61, 2004.

[10] Yuki S., Mikio S., Michio S., Takahiro H., Real-time rendering of dynamic clouds, In *SIGGRAPH '07: ACM SIGGRAPH 2007 posters, Article No.175*, 2007.

[11] A.Herout, P. Zemčík Animated particle rendering in DSP and FPGA, In *Proceedings of the 20th spring conference on Computer graphics*, pp.220-225, 2004.

[12] T. Roden, I. Parberry, Clouds and stars: efficient real-time procedural sky rendering using 3D hardware, In *ACE '05: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pp.434-437, 2005.

[13] A. Bouthors, F. Neyret, N. Max, E.Bruneton, C. Crassin, Interactive multiple anisotropic scattering in clouds, In *SI3D '08: Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pp.173-182, 2008.

[14] Ebert, D. S., Musgrave, F. K., Peachey, D., Perlin, K., and Worley, S. *Texturing & Modeling, A Procedural Approach, Third Edition*, Morgan Kaufman, 2002.

[15] Ebert, D. S. "Volumetric modeling with implicit functions: a cloud is born," In *ACM SIGGRAPH 97 Visual Proceedings: The art and interdisciplinary programs of SIGGRAPH '97*, pp.245, 1997.

[16] F. Levet, X. Granier, and C. Schlick. Fast sampling of implicit surfaces by particle systems, In *Shape Modeling International* 2006 (Short Papers), 2006.

[17] Mark J. Harris, William V. Baxter III, Thorsten Scheuermann, University of North Carolina, Technical Report #TR03-040, 2003.

도 주 영

e-mail : voronoi@nate.com

1999년 경성대학교 전산통계학과(이학사)

2001년 경성대학교 컴퓨터과학과(석사)

2001년~현 재 경북대학교 컴퓨터공학과 공학박사

관심분야 : 컴퓨터 그래픽스, 멀티미디어, 물리 기반 렌더링 등

백 낙 훈

e-mail : oceancru@gmail.com
1990년 한국과학기술원 전산학과(학사)
1992년 한국과학기술원 전산학과(공학석사)
1997년 한국과학기술원 전산학과(공학박사)
2004년~현 재 경북대학교 전자전기컴퓨터학부 교수
관심분야 : 모바일 그래픽스, 리얼타임 그래픽스

유 관 우

e-mail : kwryu@bh.knu.ac.kr
1980년 국립경북대학교 전자공학(공학사)
1982년 한국과학기술원 전산공학(공학석사)
1990년 메릴랜드대학교 전산공학(공학박사)
1991년~현 재 경북대학교 컴퓨터공학과 교수
관심분야 : 멀티패러다임 프로그래밍, 병렬 알고리즘, 컴퓨터 그래픽스 등

이 창 우

e-mail : asirion0@hanmail.net
2001년 경북대학교 컴퓨터공학과(학사)
2009년 경북대학교 컴퓨터공학과(공학석사)
2009년~현 재 ㈜케이오지 연구원
관심분야 : 컴퓨터 그래픽스, 컴퓨터 게임 등