실시간 응용을 위한 인위적인 바람의 생성
(Generating Artificial Winds for Real-time Applications)

이 남 경 † 백 낙 훈 †† 이 종 원 ††† 류 관 우 ††††
(Namkyung Lee) (Nakhoon Baek) (J. Won Lee) (Kwanwoo Ryu)

요 약 실세계에서의 바람은 자연 발생적인 것과 인위적으로 생성한 것으로 분류할 수 있다. 이제까지의 연구 결과들은 자연 현상으로서의 바람을 모델링하였다. 본 논문에서는 사람의 입이나 선풍기, 에어컨 등에서 발생하는 인공적인 바람을 모델링하기 위한 바람 모델을 제시한다. 본 논문의 바람 모델에서는 생성된 바람이 도달하는 물체를 찾아내고, 그 물체에 가해지는 힘을 계산하는 방법을 제공한다. 특히, 이 모델은 가상 현실과 같은 실시간 처리가 필요한 분야들에서 사용 가능하도록 최적화된 계산을 수행하도록 설계되었다. 본 논문에서 제시한 방법은 기존의 자연 발생적인 바람 모델들과는 보완적인 관계에 있다. 이들 모델들을 통합하여 종합적인 바람 생성 시스템을 구성할 수 있을 것으로 기대된다.

Abstract Real world wind can be classified into two categories: natural wind and artificial wind. Artificial wind can be generated by human beings, air conditioners, electric fans, etc. In this paper, a model for artificial wind is presented. We also present methods to efficiently calculate the forces applied to the objects under influence of the artificial wind. Our model is designed for real-time applications such as virtual environments. A general wind generating system can be established through integrating our model with previous wind models those are concentrated on the natural wind generation.

1. 서 론

컴퓨터 그래픽스가 발달함에 따라 그 응용 분야는 영화, 광고, 게임, 애니메이션 등으로 다양해지고 있다. 이처럼 컴퓨터 그래픽스의 응용 분야가 넓어짐에 따라, 자연 현상(natural phenomena)을 컴퓨터 그래픽스 기법으로 표현하여야 할 필요성은 더욱 증가하고 있다[1]. 반면에 물, 불, 공기의 흐름 등과 같은 자연 현상들은 불규칙적인 외형과 다양한 변화를 가지고 있어서 수학적으로 모델링하기가 까다롭다. 이러한 이유로 자연 현상들에 대한 연구 결과는 그리 많이 알려져 있지 않다. 특히, 바람(wind)의 경우에는 유체의 흐름을 다루어야 하는 어려움이 있다.

컴퓨터 그래픽스 분야에서의, 바람에 대한 이제까지의 연구 결과들은 자연 현상으로서의 바람을 표현하는 데에 초점을 맞추어 왔다[2,3,4,5]. 반면에, 실세계에서의 바람에 관계된 현상들 중에는 자연적으로 발생한 바람 못지 않게 인위적으로 생성되는 경우도 상당한 비중을 차지한다. 즉, 인간의 입으로 생성할 수 있는 바람이나, 선풍기, 에어컨, 환풍기 등에서 만들어지는 바람은 자연적으로 생성된 바람들과는 다른 방법으로 모델링되어야 할 것이다.

본 논문에서는 인위적으로 생성되는, 방향성을 가지는 바람에 대한 모델을 제시하고, 이 모델을 이용하여 가상 환경에서 바람에 의한 힘(force)을 효과적으로 반영하는 방법을 제안하고자 한다. 본 논문이 제시하는 방법은 인공적인 바람을 가상 현실(virtual reality)이나, 컴퓨터 애니메이션 등에서 효과적으로 다룰 수 있도록 하는 데에 목적을 둔다. 특히, 가상 현실에서의 적용을 고려하여 실시간 처리(real-time processing)가 가능하도록 하였다. 본 논문에서 제시하는 모델은 자연적인 바

· 본 연구는 2000년도 두뇌한국21 사업에 의하여 일부 지원되었습니다.
† 비 회 원 : 경북대학교 컴퓨터공학과
lnk@comeeng.ce.knu.ac.kr
†† 정 회 원 : 경북대학교 전자전기공학부 교수
nhbaek@ee.knu.ac.kr
††† 비 회 원 : 조지워싱턴대학교 전산학과
won@comeng.ce.knu.ac.kr
†††† 종신회원 : 경북대학교 컴퓨터공학과 교수
kwryu@bh.knu.ac.kr
논문접수 : 2000년 2월 15일
심사완료 : 2000년 6월 30일

람에 대한 기존의 연구 결과들과 상호 보완적으로 쓰일 수 있다.

또한 본 논문에서는 제한된 컴퓨터의 계산 능력을 효율적으로 사용하기 위해 LOD(level of detail) 기법을 사용하여 바람의 힘을 계산하는 방법을 제시한다. 마지막으로 바람을 이용한 새로운 사용자 인터페이스(user interface)를 제시한다.

본 논문의 구성은 다음과 같다. 2절에서는 바람에 관련된 지금까지의 연구들을 소개하고, 3절에서는 본 논문에서 사용하는 바람 모델과, 이를 이용하여 바람이 물체에 가하는 힘을 계산하는 방법을 제시한다. 또 가상의 바람에 의해 영향을 받는 물체들을 효과적으로 찾는 방법을 제시한다. 4절에서는 제안된 바람 모델을 확장해서 LOD 기법을 이용한 표본점의 수를 변화시키는 방법과 바람을 이용한 사용자 인터페이스에 대해 기술한다. 5절에서는 본 논문에서 제시한 바람 모델을 입자와 가상 모빌에 적용한 실험 결과를 보이고 마지막으로 6절에서 결론과 앞으로의 연구 방향을 제시한다.

2. 관련 연구

이제까지 컴퓨터 그래픽스 분야에서의 바람에 대한 연구는 자연 현상을 표현하는 데에 치중하여 왔다. 따라서, 바람은 주어진 환경 전체에 영향을 미치는 장(field)의 형태로 표현되어 왔다.

Reeves와 Blau는 나무와 숲을 표현하는 과정에서 structured particle system을 이용하여 간단한 바람 모델을 제시하였다[2]. 이 방법에서는 사용자가 바람이 영향을 미치는 지역, 바람의 평균 속도와 방향을 명시하면 시스템은 명시한 바람의 방향으로 퍼져 가는 파형(wave)을 생성하는 작은 돌풍(turbulence)들을 생성한다. 이때 파형의 세기는 무작위하게 결정된다. 특정 위치에서의 바람의 영향은 주위에 있는 돌풍들에 의해 만들어지는 파형으로 결정된다. 이처럼 2차원의 바람 지도(wind map)를 각 frame에 할당해서 바람에 의한 풀잎이나 나뭇잎의 움직임을 표현하였다. Reeves와 Blau의 방법은 확률적인 바람 함수와 바람 지도를 이용해서 간단한 방법으로 바람을 만들었지만, 바람에 대한 세밀한 제어는 어렵다.

Wejchert와 Haumann는 바람에 날리는 낙엽과 같이, 유체 속에서의 물체의 움직임을 시뮬레이션하기 위해 공기역학(aerodynamics)을 이용했다[3]. 유체의 흐름을 표현하기 위해서 flow primitive를 제시했고, flow primitive들을 조합해서 표현하고자 하는 바람의 장을 만든 후 바람의 장 안에서 물체의 움직임을 표현했다. 또 물리적 계산의 편의를 위해서 전체적인 구성을 fluid flow 영역과 object boundary 영역으로 분리하여 계산하는 방법을 제시했다. 계산량을 줄이기 위해서 fluid flow 영역에서는 유체의 흐름을 수치적으로 계산하는 것이 아니라 flow primitive의 물리적 성질을 분석하여 구했다. object boundary 영역에서는 물체 근처에서의 유체의 흐름을 스토크의 항력 방정식 (Stoke's drag equation)을 이용하여 계산했다.

이 방법은 낙엽과 같은 가벼운 물체가 바람에 의해 움직이는 현상을 비교적 자연스럽게 표현할 수 있다. 반면에 바람의 장(wind field)이 고정된 형태로 제한되어, 자연 현상으로서의 바람이 가지는 무작위성(randomness)을 제대로 표현할 수 없다는 단점이 있다.

Shinya와 Fourinier는 건축공학에서 연구된 바람의 자료들을 바탕으로 바람의 확률적인(stochastic) 성질을 첨가한 모델을 제안했다[4]. 이 바람 모델은 바람의 장을 만들기 위해서 실제로 측정한 바람의 자료를 퓨리어 변환(Fourier transform)하여 spatiotemporal frequency 영역으로 변환시킨 뒤, 바람의 확률적인 성질을 반영하기 위해서 무작위한 값을 각 자료에 더했다. 수정된 자료를 다시 역퓨리어 변환(inverse Fourier transform)을 해서 애니메이션에 적용될 바람의 장(wind field)을 만들었다. 이 방법을 이용하여 만든 바람의 장은 건축 공학에서의 결과에 바탕을 두었기 때문에 인위적인 바람을 표현하기에는 적합하지 않다.

Stam과 Fiume은 바람과 같이, 흐르는 유체 내에서의 연기(smoke) 흐름을 확률적인 과정을 통해서 표현하는 방법을 제안했다[5]. 바람의 장을 구하기 위해서는 나비어-스토크 방정식(Navier-Stokes equations)을 이용하여 유체의 속도와 압력의 변화를 시공간(space-time) 상에서 풀었다. 이 방법에서 Stam과 Fiume은 바람의 장(wind field)을 large-scale 요소(component)와 small-scale 요소로 분리하여 구성했다. large-scale 요소는 점성이 높은 유체에서의 결과를 이용하여 전체 바람의 장을 설정하는데 이용된다. small-scale 요소는 빠른 역퓨리어 변환 방법(fast inverse Fourier transform method)을 이용하여 전체 바람의 장에 삽입될 돌풍을 만들어 낸다.

Stam과 Fiume의 방법은 사용자가 대략의 바람의 형태를 설정하면 세부적인 바람을 확률적인 방법으로 생성하여 최종적인 바람의 장을 구성한다. 이 방법으로 만든 바람의 장은 인위적인 바람의 특징을 가지고 있지 않으므로 인위적으로 생성되는 바람에 적용하기는 곤란하다.

이처럼 기존의 연구 결과들은 모두 자연발생적인 바람을 모델링하기 위해서 제안되었기 때문에 인위적으로 생성되는 바람에 적용할 수 없다. 따라서 본 논문에서는 인위적으로 생성되는 바람 모델을 제시한다.

3. 바람 모델

가상의 바람을 생성하기 위해서는 크게 3가지 문제를 다루어야 한다. 우선, 가상의 바람이 진행해 나가는 형태를 전체적으로 모델링하는 방법이 필요하다. 또, 바람이 물체에 부딪혔을 때, 해당 물체에 가해지는 힘을 계산할 수 있어야 한다. 마지막으로, 어느 물체가 바람의 영향을 받는 지를 알아낼 수 있는 방법이 필요하다. 본 절에서는 이 세 가지 문제를 차례로 다룬다.

3.1 가상 바람

기존의 연구 결과들은 자연 현상으로서의 바람을 모델링하였기 때문에, 환경 전체에 걸친 바람의 장(wind field)을 설정하였다. 반면에, 사람의 입에서 나오는 바람이나, 선풍기가 만들어내는 인위적인 바람은 국지적으로 영향을 끼친다고 가정할 수 있으므로, 국지적인 바람 모델을 사용할 수 있다. 이 경우는 계산량의 감소로 빠른 처리가 가능하다는 장점을 가진다.

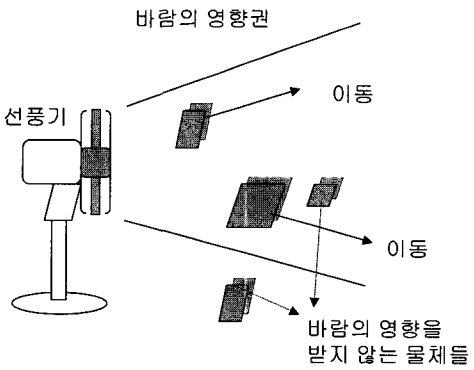
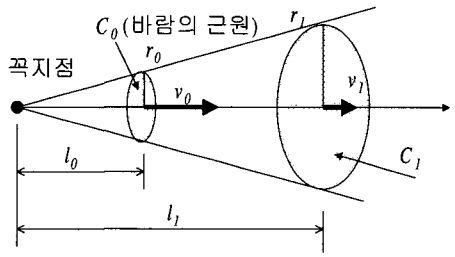


그림 1 선풍기에 의해 만들어지는 바람

그림 1에서는 인위적으로 바람을 생성시키는 것들 중의 하나인 선풍기를 예로 들었다. 선풍기가 생성하는 바람은 모터를 중심으로 원뿔 형태로 바람의 영향권을 형성한다. 바람의 영향을 받는 물체는 바람의 영향권 내에 있어야 하며, 또 직접적으로 바람과 부딪쳐야만 한다. 즉 바람의 영향권 밖에 있거나 앞의 물체에 의해 가려진 물체는 바람의 영향을 받지 않는다.

본 논문에서는 이러한 관찰을 기초로 인위적으로 생성되는 바람이 그림 2에서와 같이, 그 근원(source)으로부터 원뿔 모양으로 퍼져 나간다고 가정한다. 바람의 근원(source of wind)은 반지름 r_0의 원으로 설정하고, 이 원의 중심에서 l_0의 거리에 원뿔의 꼭지점이 위치한다. 가상 바람은 주어진 원으로부터 $v_0(t)$의 풍속(風速)으로 꼭지점에서 멀어지는 방향으로 퍼져 나간다. 풍속은 시간에 따라 변할 수 있으므로, 시간 t에 대한 함수로 정의된다. r_0, l_0, $v_0(t)$는 바람의 형태를 조정하기 위해, 사용자가 설정할 수 있다.



$A_0 v_0 = A_1 v_1$

그림 2 바람 모델

3.2 바람의 힘

바람은 본질적으로 공기라는 유체의 흐름이다. 가상 바람을 효과적으로 모델링하기 위해, 본 논문에서는 공기는 비점착성(inviscid)의 압축되지 않는 유체이고, 이 유체의 흐름은 주어

진 원뿔 내로 폐쇄된다고 가정한다. 이러한 가정은 정상 풍속의 바람에 적용될 수 있다[3].

가상 바람의 근원을 면적 A_0를 가지는 단면 C_0라 하면, 그림 2의 원뿔 모델에서 면적 A_1을 가지는 임의의 단면 C_1을 설정할 수 있다. 폐쇄된 유체의 흐름에서는 단위 시간에 단면 C_0와 C_1을 각각 통과하는 유체의 질량이 같아야 한다. 따라서, 이 두 단면 사이에서의 유체의 흐름은 연속 방정식(continuity equation)으로 다음과 같이 표현할 수 있다[6].

$$A_0 v_0 = A_1 v_1 \tag{1}$$

이 때, 두 단면 C_0, C_1에 대하여, $l_0 : l_1 = r_0 : r_1$이고, 식 (1)로부터 풍속 v_0, v_1의 관계를 다음과 같이 구할 수 있다.

$$\frac{v_1}{v_0} = \frac{A_0}{A_1} = \frac{\pi r_0^2}{\pi r_1^2} = \frac{\pi l_0^2}{\pi l_1^2} = \frac{l_0^2}{l_1^2} \tag{2}$$

위와 같이 모델링된 바람이 물체에 부딪혔을 때, 해당 물체에 작용하는 힘을 계산하기 위해서는 다음과 같은 스토크 항력 방정식(Stoke drag equation)을 사용할 수 있다[3].

$$\vec{F}_{stoke} = \rho A v^2 (\vec{n_v} \cdot \vec{n_a}) \vec{n_a} \quad (3)$$

여기서, A와 v는 각각 바람을 받는 부분의 표면적과 풍속이다. 또, $\vec{n_v}$와 $\vec{n_a}$는 각각 유체의 흐름을 나타내는 단위 벡터와 물체 표면의 단위 법선 벡터이다. ρ는 유체의 밀도를 나타내는데, 바람의 경우는 상수값이 된다. 그림 3에서 제시한 바람 모델에서는, 스토크 항력 방정식으로부터 단면 C_1에 위치한 면적 A를 가지는 평면에 작용하는 힘을 아래와 같이 계산할 수 있다.

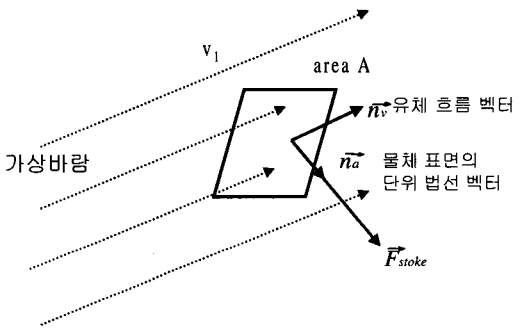


그림 3 바람에 의한 힘

$$\vec{F}_{stoke} = \rho A v_1^2 (\vec{n_v} \cdot \vec{n_a}) \vec{n_a} \quad (4)$$

또, 유체의 흐름 방향을 나타내는 단위 벡터 $\vec{n_v}$는 원뿔의 꼭지점에서 물체의 해당 지점을 연결한 벡터를 정규화(normalize)하여 구할 수 있다. 단면 C_1에서의 풍속 v_1은 식 (2)를 이용하여 단면 C_0에서의 풍속 v_0로 표시할 수 있으므로, 다음과 같이 표현할 수 있다.

$$\begin{aligned} \vec{F}_{stoke} &= \rho A \frac{l_0^4}{l_1^4} v_0^2 (\vec{n_v} \cdot \vec{n_a}) \vec{n_a} \\ &= \alpha A \frac{v_0^2}{l_1^4} (\vec{n_v} \cdot \vec{n_a}) \vec{n_a} \end{aligned} \quad (5)$$

식 (5)에서 v_0는 사용자의 설정에 따라 바람의 형태를 결정하는 데에 사용된다. 또, 상수값 α는 원뿔의 형태에 따라 미리 계산되어질 수 있다. 단면적 A, 꼭지점으로부터의 거리 l_1, 단위 벡터 $\vec{n_v}$와 $\vec{n_a}$들은 해당되는 평면에 따라 손쉽게 계산되어질 수 있다. 따라서, 가상의 바람이 물체 표면에 가하는 힘은 실시간으로 계산되어 질 수 있고, 가상 현실 등의 실시간 응용에서 사용할 수 있다.

실제 세계에서의 인위적인 바람은 물체와 부딪치면서 굴절하거나 회절하는 현상이 일어날수 있다. 특히, 상대적으로 작은 물체의 뒷부분에서는 회절된 바람이 영향을 미칠 수도 있다. 또, 바람 자체가 가지는 무질서성을 반영하여야 할 필요도 있다. 본 논문에서는 이러한 점들을 고려하기 위해, 식 (5)에서 계산한 힘에 무질서한 항을 더하여 최종적으로 다음과 같은 식을 사용한다.

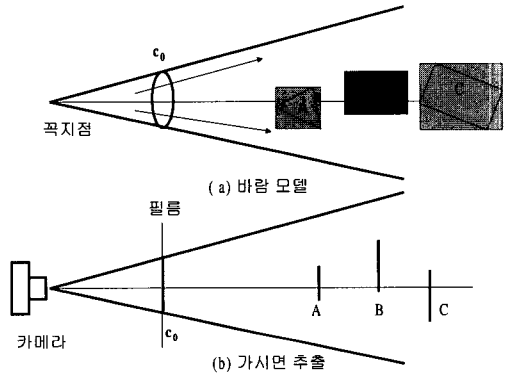


그림 4 가시면 추출 방법의 응용

$$\vec{F}_{wind} = \vec{F}_{stoke} + \vec{F}_{rand} \quad (6)$$

무질서하게 더해지는 $\vec{F}_{rand}$는 사용자가 정의하는 상수값 β를 이용하여, $|\vec{F}_{rand}| < \beta |\vec{F}_{stoke}|$의 관계를 만족시키도록 설정된다. $\vec{F}_{rand}$의 방향은 무작위로 설정된다.

3.3 바람의 영향을 받는 면 추출

3.1절과 3.2절에서 각각 바람의 전체적인 모델과 특정 물체에 가하는 힘을 계산하는 방법을 보였다. 가상 바람의 처리에 있어서는 어느 물체가 실제로 바람을 받게 되는지를 처리하는 것이 상당한 비중을 차지한다. 본 논문에서는 가상 바람의 실시간 처리를 위하여, 바람이 물체에 직접 부딪히는 경우에만 힘을 가한다고 가정한다.

3.1절에서 제시한 바람 모델에서는 원뿔의 내부에 포함되는 면에만 힘이 가해지게 된다. 또, 특정한 면이 다른 물체에 의해 가려지는 경우에는 바람의 직접적인 영향을 받지 않으므 로, 힘이 가해지지 않는다. 따라서, 바람의 영향을 받는 면들은 원뿔의 꼭지점으로부터 가시(visible)적이고, 원뿔의 내부에 포함된다는 조건을 만족

시킨다.

본 논문에서는 바람의 영향을 받는 면들을 알아내기 위해서 가시면 추출 방법(visible surface detection)을 응용한다[7]. 그림 4에서처럼 가시면 추출 방법에서의 카메라의 위치를 바람 모델의 원뿔의 꼭지점에 대응시키고, 바람의 근원에 해당하는 단면 C_0를 포함하는 평면을 카메라 필름의 위치에 대응시킨다. 원뿔 내에 포함되는 지의 여부는 필름 상에서 단면 C_0에 대응되는 원의 내부에 포함되는 지를 묻는 2차원 문제가 된다.

```
Precedure zBuffer;
  var pz: integer;   {Polygon's z at pixel coords (x,y)}
Begin
  for y := 0 to YMAX do
        for x := 0 to XMAX do
           begin
                WritePixel(x,y,BACKGROUND_VALUE);
                WriteZ(x,y,0);
           end
for each polygon do
  for each pixel in polygon's projection do
        begin
           pz := polygon's z-value at pixel coords(x,y);
           if pz >= ReadZ(x,y) then
                begin { New point is not farther }
                  WriteBuffer(x,y,pz);
                  WritePixel(x,y,polygon's color at pixel coords(x,y));
                end
        end
End;
```

그림 5 깊이 버퍼 방법(z-buffer method)

```
Procedure Findface
  var pixel: real;
begin
  for each polygon do
     for each point i(x,y,z) in sampling_point_list do
          begin
            Project(x,y,z) to the screen coordinate (wx,wy)
            ReadPixel(wx,wy,pixel);
            if pixel = i's index in sampling_point_list
                  i is visible;
          end
End
```

그림 6 바람의 영향을 받는 점의 추출 방법

컴퓨터 그래픽스에서 사용되는 가시면 추출 방법들에는 여러 가지가 있다. 본 논문에서는 일반적으로 적용가능하고, 처리 시간을 향상시키기 위해 깊이 버퍼 기법(depth buffer method)을 사용한다[7,8]. 이 방법은 그래픽스 하드웨어의 지원을 받을 수 있다는 장점이 있다. 원래의 깊이 버퍼 기법에서는 그림 5와 같은 방법으로 가시면들이 표시된 화면이 최종 결과가 된다. 하지만 바람 모델의 경우에는 어느 면이 보이는 지를 찾아내야 한다. 이를 위해서, 깊이 버퍼에 면의 색상 정보 대신, 어느 면인지를 나타내는 인덱스 값을 저장하도록 하고, 처리가 끝난 깊이 버퍼에서 이 인덱스 값들을 읽어들여 가시면들이 어느 것인지를 찾아낼 수 있다.

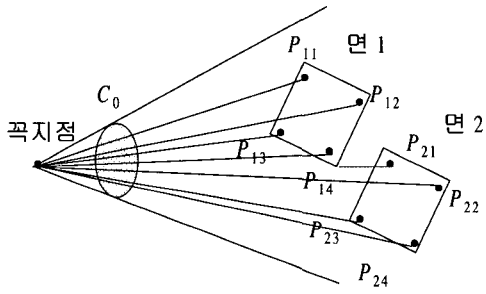


그림 7 바람의 영향을 받는 면의 추출

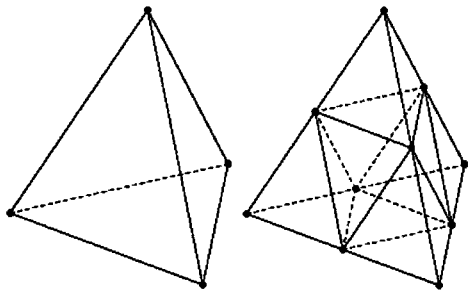
또, 바람의 모델링에 있어서는 그 특성상 면 상에서 영향을 받는 부분을 정확히 찾아낼 필요는 없다. 본 논문에서는 각각의 면 위에 표본점들을 분포시킨 후, 각 표본점들이 대응되는 면 상의 일정 부분의 가시성을 결정하도록 하여 처리 속도를 향상시켰다.

본 논문에서는 모든 표본점에 대해 그림 6의 과정을 통해 바람의 영향을 받는 점들을 추 출해 낸다. 전처리 작업(preprocessing)으로 *sampling_point_list*라는 자료구조에 모든 표본점들의 위치 정보와 인덱스 정보를 저장한다. 3차원 상에 표현된 물체들이 2차원으로 투영되면 가시면만 나타난다. OpenGL의 gluProject함수를 이용하여 표본점이 투영될 2차원의 좌표값을 읽어온 후 각 좌표의 픽셀(pixel)정보는 glReadPixel함수를 이용하여 가져온다[9]. 픽셀 값과 미리 구성해 둔 표본점 리스트의 값이 일치한다면 투영된 표본점은 바람의 영향을 받는 점으로 판정된다.

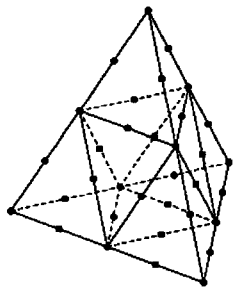
그림 7은 면 1과 면 2에 대해서 바람의 영향을 받는 부분을 찾아내는 예이다. 각 물체들에 대해 깊이 버퍼 기법을 적용하면 동일한 위치에 투영될 점들을 비교해서 앞의 물체에 의해 가려지는 점들은 깊이 버퍼에 저장되지 않는다. 처리가 끝난 깊이 버퍼의 정보를 이용해서 바람의 영향을 받는 점들을 찾아낸다. 면 1의 경우는 표본점 4개가 모두 보이지만, 면 2에서는 표본점 P_{21}이 보이지 않으므로, 면 2의 P_{22}, P_{23}, P_{24}에 대응되는 부분에만 바람에 의한 힘이 작용한다.

이 과정을 통해 바람의 영향을 받는 것으로 결정된

면 상의 각 부분에는 3.2절에서 제시한 방법으로 계산된 힘이 적용된다. 각 물체에 가해지는 힘이 결정되면 물체들의 무게를 알고 있으므로 각 물체의 선가속도와 각가속도를 계산할 수 있다. 계산된 선가속도와 각가속도로부터 차례로 선속도, 각속도를 계산할 수 있다. 이를 이용해 최종적으로 다음 frame에 서의 각 물체들의 위치와 회전각을 계산한다.



level 0 level 1



level 2

그림 8 표본점 구하기

4. 개선 및 확장

4.1 표본점의 개수 조절

물체를 움직이는 바람의 힘을 정밀하게 계산하기 위해서는 많은 수의 표본점이 필요하지만 컴퓨터의 계산 능력(computing power)은 제한되어 있으므로 많은 표본점을 사용하면 계산 시간이 길어진다. 본 논문에서는 앞 절에서 제시한 바람 모델을 실시간에 적용 가능하게 하는 방법을 제시한다.

카메라의 시점과 바람의 근원은 동일한 위치에 있으므로 카메라로부터 먼 거리에 있는 물체는 바람의 영향을 적게 받는다. 본 논문에서는 물체와 바람의 근원 사이의 거리에 반비례하게 표본점을 할당한다. 즉 멀리 있는 물체에는 적은 수의 표본점을, 카메라의 시점에서 가까이에 있는 물체에는 많은 표본점을 할당한다. 이 경우 카메라에 가까운 물체는 상대적으로 많은 표본점을 가지고 있으므로 보다 정확히 바람의 힘을 계산할 수 있다. 또한 컴퓨터에 따라서 계산 능력이 다르므로 계산 능력이 작은 컴퓨터에서는 표본점의 수를 적게 하여 계산하게 하고 반대로 계산 능력이 큰 컴퓨터에서는 표본점의 수를 증가시켜 보다 세밀하게 바람의 힘을 계산할 수 있다. 이 개념은 그래픽스 분야에서 이미 사용되고 있는 LOD 기법과 유사하다. 이 개념을 사용할 때 유의할 점은 동일한 세기의 바람이 한 물체에 가해질 때 표본점의 개수가 변하여도 그 물체가 받는 힘의 총량은 변하지 않아야 한다는 것이다. 본 논문에서는 각 물체마다 변수 γ를 두어 표본점의 수가 변함에 따라서 γ값을 변화시켜 동일한 세기의 바람이 물체에 가하는 힘을 표본점의 수의 변화와 관계없이 동일하게 만들어 준다. 즉 γ값과 표본점의 개수는 반비례관계에 있다.

본 논문에서는 그림 8에서처럼 삼각뿔을 이용해서 각 물체의 표본점을 정한다. 삼각뿔의 꼭지점들이 기본 표본점이 된다. 한 단계(level)씩 증가할 때마다 그림 8에서처럼 삼각분할을 해서 새로운 분할점들을 표본점에 추가한다.

실제로 바람의 영향을 받는 점들은 삼각뿔 상의 점들이 아니라 물체 표면의 점들이므로 삼각뿔 상에서 표본점들이 정해지면 물체 표면의 점으로 사상(mapping)시켜야 한다. 삼각뿔 상의 표본점을 물체 표면의 점으로 사상시키기 위해서 먼저 물체의 무게중심을 삼각뿔의 중심으로 이동시킨다. 그런 다음 삼각뿔의 중심과 표본점으로 이루어지는 직선 위에서 레이캐스팅(ray casting)을 사용하여 표본점을 삼각뿔 내부나 외부로 이동시킨다. 물체의 표면과 만나는 점이 삼각뿔 상의 표본점이 사상될 위치이다.

물체들의 표본점을 정한 후 시점에서의 거리를 비교해 가며 각 물체들의 표본점의 수를 변화시킨다. 물체가 시점에서 i 단계에 설정된 거리 이내로 떨어져 있으면 그 물체의 i 단계이하의 표본점들을 활성화시킨다. 이때 활성화된 표본점의 수를 고려해서 변수 γ값을 변화시킨다. 이 과정은 전처리가 가능하므로, 실제 수행시간에는 영향을 미치지 않는다.

4.2 사용자 인터페이스

컴퓨터가 발전함에 따라서 사용자 인터페이스는 컴퓨터 분야에 있어서 중요한 부분으로 자리매김되고 있다. 본 논문에서 제시한 바람 모델을 가상 현실과 같은 대화형 환경에서 이용하려면 적절한 사용자 인터페이스가 필요하다.

바람을 이용한 사용자 인터페이스의 가장 대표적인

예는 마이크를 이용하는 것이다. 마이크는 사람의 음성 신호를 입력 받아서 이 음성 신호를 증폭시킨 값을 스피커를 통해 출력하는 장치이다. 바람의 영향을 받는 물체의 움직임을 마이크를 이용하여 바람의 변화를 인터렉티브(interactive)하게 입력받으면 보다 현실감있게 시뮬레이션할 수 있다.

마이크를 통해 들어오는 입력은 음성의 진폭(amplitude)이다. 직관적으로 생각할 때 바람이 세게 불면 마이크를 통해 입력되는 음성 신호도 클 것이다. 또한 바람의 세기는 바람의 속력에 비례하므로 본 논문에서 제시한 바람 모델에서 속력항, 즉 $v_0(t)$에 마이크를 통해 받아들인 입력을 대치하여 사용할 수 있다. 이 방법은 가상 현실에서 사람이 입으로 바람을 불어 물체를 움직이고자 할 때 적용할 수 있다. 마이크보다 더 정밀하게 바람의 세기를 측정할 수 있는 입력 장치도 있지만, 마이크는 PC 환경에서 손쉽게 구할 수 있고 가격도 저렴하다는 장점이 있다.

5. 실험 결과

이 절에서는 본 논문에서 제시한 바람 모델을 적용시킨 예들을 보이고자 한다. 각 예들은 Pentium 계열의 CPU를 장착한 PC환경에서 Visual C++ 컴파일러와 OpenGL 라이브러리로 구현되었다.

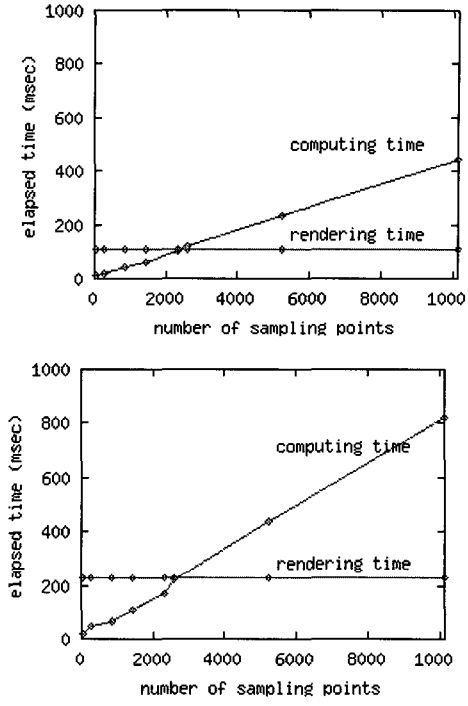
그림 10은 입자들이 선풍기 바람에 의해 퍼져 나가는 모습을 시뮬레이션한 예이다. 입자 모델은 기존의 입자 모델링 기법[2]으로 모델링되었고, 화면의 왼쪽에서 불어 오는 바람에 의해 각 입자들이 날리는 모습을 볼 수 있다.

표 1 폴리곤의 수, 렌더링 시간, 계산시간 간의 관계

	폴리곤 수	렌더링 시간	표본점 수	계산 시간
입자 모델	564	0.06 sec/frame (≃ 0.11 frame/sec)	60	0.0024 sec/frame
가상 모빌	1788	0.11 sec/frame (≃ 9.09 frame/sec)	290	0.02 sec/frame

표 2 표본점과 계산시간의 관계 (단위 : sec/frame)

	렌더링 시간	계산 시간 (sec/frame)							
표본점 수		60개	290개	870개	1450개	2320개	2610개	5220개	10150개
컴퓨터 1	0.11	0.010	0.020	0.041	0.060	0.100	0.121	0.231	0.441
컴퓨터 2	0.23	0.020	0.046	0.065	0.110	0.170	0.220	0.440	0.820



(a) 컴퓨터 1 (Pentium II 400MHz)
(b) 컴퓨터 2 (Pentium 200MHz)

그림 9 표본점과 계산시간과의 관계

그림 11의 경우는 물리 기반 모델링 기법(physically based modeling)으로 모델링된 가상의 모빌[10]에 바람 모델을 적용시킨 예이다. 특히, 이 시뮬레이션에서는 대상이 되는 모빌 자체의 움직임이 물리 기반 모델링으로 비교적 세밀하게 제어되기 때문에 더욱 사실감 있는 움직임을 볼 수 있다. 이 예에서 바람이 정면으로부터 모빌을 향해 불고 있다.

128Mbyte의 RAM을 가진 Pentium II 400MHz 기종에서 각 시뮬레이션을 행한 결과를 표 1에 제시되어 있다. 이 결과에서 볼 수 있듯이, 본 논문의 바람 모델을 적용시키는 데에는 상당히 짧은 계산 시간으로도 충분하다. 표에 제시된 정도의 계산 시간은 실제 모델을 렌더링하는 시간에 비해서는 거의 무시할 수 있을 정도이므로, 본 논문이 목적했던, 가상 환경에서의 실시간처리에 사용가능하다.

4절에서 설명한 표본점의 개수 조절 기법을 분석하기 위해서는 서로 다른 성능을 가진 2대의 컴퓨터들을 사용하였다. 컴퓨터 1은 128Mbyte의 RAM을 가진

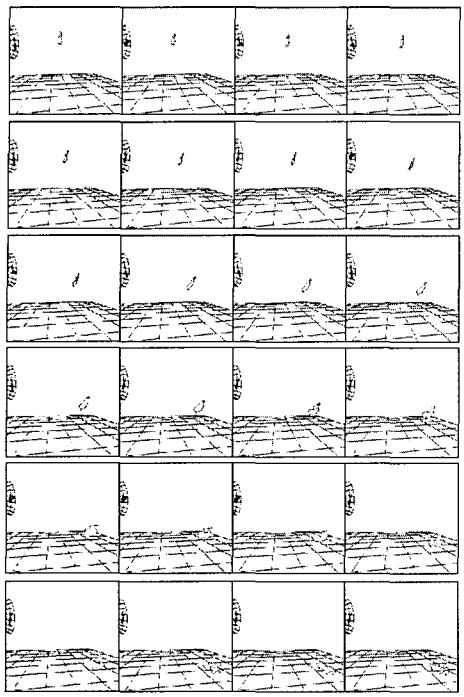


그림 10 먼지에서의 적용

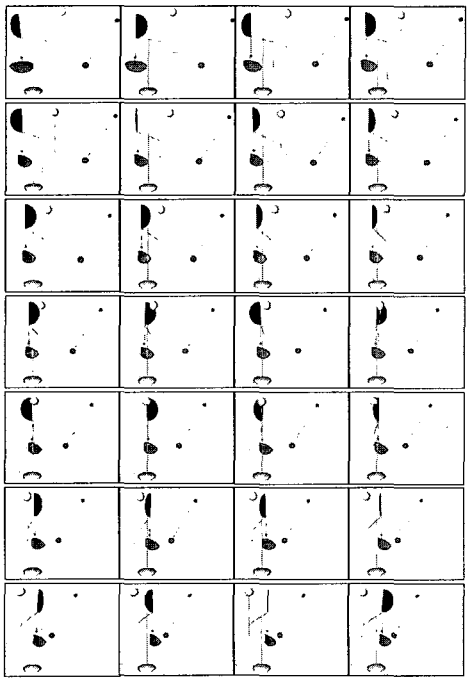


그림 11 가상 모빌에서의 적용

Pentium II 400MHz 기종이고, 컴퓨터 2는 32Mbyte의 RAM을 가진 Pentium 200MHz 기종이다.

가상 모빌의 시뮬레이션 시에 표본점의 수를 60개에서 10150개까지 다양하게 변화시킨 예는 표 2에 제시되어 있다. 표본점의 개수를 변화하더라도 렌더링을 위해서는 동일한 모델을 사용하였기 때문에, 렌더링시간에는 차이가 없다. 반면에, 표본점의 개수를 변화시킴에 따라 바람 모델의 계산시간은 그림 9에서와 같이 선형으로 증가한다.

바람 모델에서의 계산 시간은 사용된 기종에 따라 당연히 차이를 보인다. 실제 적용에 있어서는 표 2의 자료를 이용하여, 사용 기종과 원하는 초당 프레임(frame/sec)에 따라 전체 표본점의 수를 조절함으로써 주어진 환경에서 가장 적합한 시뮬레이션 결과를 얻을 수 있다. 4절에서 설명한 표본점의 개수 조절 기법이 목적했던 바가 바로 이러한 환경차이에 따른 모델의 계산시간 차이를 극복하고자 하는 것이었고, 본 논문에서 제시한 표본점 개수 조절 기법이 유용함을 알 수 있다.

6. 결 론

실세계에서의 바람은 크게 자연 발생적인 것과 인위적으로 생성되는 것으로 나눌 수 있다. 이제까지의 연구결과들은 주로 자연 현상으로서의 바람을 모델링하였다. 본 논문에서는 사람의 입이나 선풍기, 환풍기, 에어컨 등에 의해 생성되는 바람을 모델링하기 위한 바람 모델을 제시하였다.

본 논문이 제안한 바람 모델은 가상 현실과 같은, 실시간 처리가 필요한 분야에서도 사용하기 위해서 계산량을 줄일 수 있도록 설계되었다. 또, 주어진 환경에서 가상의 바람을 받게 되는 물체와, 그 물체에 가해지는 힘을 계산하는 방법을 제시하였다. 이러한 방법들은 인위적으로 생성된 바람이 주어진 환경에 미치는 영향을 효과적으로 반영할 수 있다.

또 각 물체에 가해지는 바람의 힘을 계산할 때, LOD와 유사한 표본점의 개수 조절 기법을 사용해서 가까이 있는 물체들은 보다 정확하게 계산하고, 멀리 있는 물체들은 상대적으로 덜 정확하게 계산하는 방법을 제시했다. 이 방법으로 컴퓨터의 제한된 계산 능력을 효율적으로 사용할 수 있다.

본 논문에서 제시한 방법은 기존 연구들에서 제안한, 자연 발생적인 바람의 모델링과는 서로 보완적인 역할을 담당한다. 따라서 앞으로는 두 모델들간의 통합이 필요하다. 이를 통해 종합적인 바람 생성 모델을 설계할 수 있을 것이다. 동역학에 기초한 애니메이션 시스템과

의 통합 역시 앞으로의 연구 과제이다. 또한 바람을 이용하는 사용자 인터페이스는 가상 현실이나 사용자 인터페이스 분야에서 향후에 다루어질 연구 과제이다.

참 고 문 헌

[1] J. D. Foley, A. van Dam, S. K. Feiner and J. F. Hughes, *Computer Graphics - principles and practice*, Addison Wesley, 2nd edition, 1990.

[2] W. T. Reeves and R. Blau, Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle System. In *SIGGRAPH'85*, volume 19, pages 313-322, 1985.

[3] J. Wejchert and D. Haumann, Animation Aerodynamics. In *SIGGRAPH'91*, volume 25, pages 19-22, 1991.

[4] M. Shinya and A. Fournier, Stochastic Motion - Motion Under the Influence of Wind. In *Eurographics'92*, pp.119-128, (September 1992).

[5] J. Stam and E. Fiume, Turbulent Wind Fields for Gaseous Phenomena. In *SIGGRAPH'93*, pages 369-376, 1993.

[6] V. L. Streeter and E. Benjamin Wylie, *Fluid Mechanics*, McGraw-Hill, 1998.

[7] D. Hearn and M. P. Baker, *Computer Graphics*, Prentice-Hall, Inc., 1997.

[8] E. Angel, *Interactive Computer Graphics : A top-down approach with OpenGL*, Addison-Wesley, 1997.

[9] J. Neider, T. Davis, M. Woo, *OpenGL Programming Guide : The official guide to Learning OpenGL, release 1*, Addison-Wesley, 1993.

[10] D. Lee et al, Reproducing works of Calder, *Journal of Visualization and Computer Animation*, (submitted).

이 남 경

1998년 경북대학교 공과대학 컴퓨터공학과 졸업 2000년 경북대학교 컴퓨터공학과 공학석사 2000년 ~ 현재 경북대학교 컴퓨터공학과 박사과정. 관심분야는 컴퓨터 그래픽스, 알고리즘

백 낙 훈

1990년 한국과학기술원 과학기술대학 전산학과 졸업. 1992년 한국과학기술원 전산학과 공학석사. 1997년 한국과학기술원 전산학과 공학박사. 1997년 ~ 1998년 미국 죠오지워싱턴 대학 초청 연구원. 1998년 ~ 2000년 경북대학교 전자전기공학부 초빙교수. 2000년 ~ 현재 경북대학교 전자전기공학부 BK21 계약교수. 관심분야는 컴퓨터 그래픽스, 계산기하학, 계산 이론, 멀티미디어 시스템

이 종 원

1982년 ~ 1886년 경북대학교 수학과 학사졸업. 1989년 ~ 1992년 미국 일리노이 공과대학 Computer Science 석사 졸업. 1992년 ~ 현재 미국죠지워싱턴 대학교 Computer Science, Computer Graphics 전공, 박사과정. 관심분야는 컴퓨터 그래픽스, 계산기하학

유 관 우

1980년 경북대학교 전자공학과 졸업(공학사). 1982년 한국과학기술원 전산학과 졸업(이학석사). 1990년 미국 메릴랜드대학(University of Maryland) 졸업. 1982년 ~ 현재 경북대학교 컴퓨터공학과 부교수.