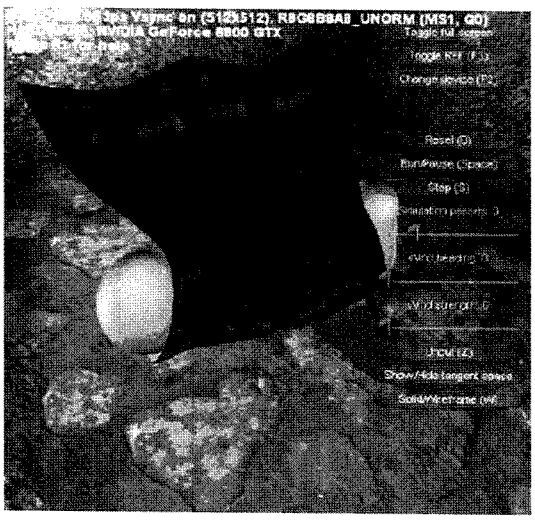
특집

Nvidia 옷 시뮬레이션기법

김종진(한국폴리텍1 서울강서대학), 이면재(남서울 대학교), 강호석(홍익대학교)

I. 서 론

옷 시뮬레이션이란 3D애니메이션이나 게임중에 캐릭터가 입고 있는 옷이 실제 옷과 같이 움직일 수 있도록 제공해 주는 기법을 말한다. 이러한 옷 시뮬레이션의 계산은 CPU에서 GPU로 옮겨가고 있고, 보다 효율적이고 정확하고 빠른 계산을 하기 위한 연구가 진행되고 있다. 본 고에서는 Ndvida Corporation의 옷 시뮬레이션을 위한 White Paper 자료와 다른 자료를 소개하고자 한다.



〈그림 1〉 옷 시뮬레이션의 예

II. 모 델

옷은 파티클의 정렬로 만들어지며 각각의 파티클은 다음과 같은 사항을 고려해야한다.

- 중력이나, 바람과 같이 외부에서 주어지는 힘
- 물체의 모양을 유지하려고 하는 제약 사항 (스프링 제약 사항)
- 환경에 의한 상호 침투를 막는 제약 사항 (충돌 제약 사항)

각각의 파티클은 힘에 영향을 받기 쉽고 그것의 운동 등식은 Verlet 적분을 이용하여 적분되어진다. 지속적인 시뮬레이션 Δt의 경우에, 파티클의 새로운 위치는 식(1)과 같이 계산된다.

$$P(t + \Delta t) = P(t) + k\,(P(t) - P(t - \Delta t)) + \Delta t^2 F(t) / m \quad \ldots\ldots (1)$$

P(t)는 시간 t 일 때 파티클의 위치,

F(t)는 시간 t 일 때의 힘, m은 파티클의 질량, k는 보통 1에 매우 가까운 임의적 제동 계수이다.

식(1)은 $P(t + \Delta t)$ 와 $P(t - \Delta t)$에 가까운 Taylor 전개의 첫 번째 형식에 의해 파생되었다.

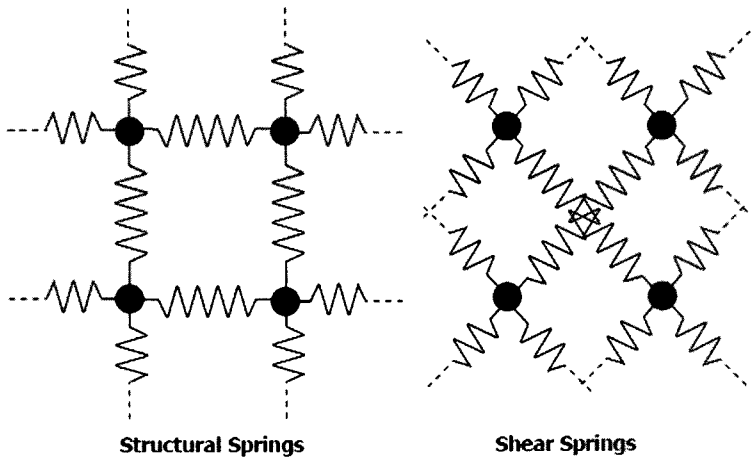
$$P(t + \Delta t) \sim P(t) + \Delta t\, P'(t) + \Delta t^2\, P''(t) / 2$$
$$P(t - \Delta t) \sim P(t) - \Delta t\, P'(t) + \Delta t^2\, P''(t) / 2$$

이 두 식을 더하고 P(t)가 F(t)와 같다는 것을 주의한다. 어떠한 임의의 힘도 이용자의 상호작용에 해 고정되거나 움직이기 위해 선택했던 파티클에 영향을 주지 않는다. 또한, 파티클은 <그림 2>에서와 같이 구조(structural) 스프링과 Shear 스프링에 의해 서로 연결되어 있다.

스프링들은 아주 강하게 연결되어져 있다고 가정한다. 평평한 직사각형 옷의 크기(ex, ey)에서 파티클(W x H)의 정렬에 의해 만들어진다.

$$d_{structural,x} = ex / (W - 1)$$
$$d_{structural,y} = ey / (H - 1)$$
$$d_{shear} = \sqrt{d^2_{structural,x} + d^2_{structural,y}}$$

직사각형이 아니거나 평평하지 않은 옷에서, 모든 스프링은 정지한 때 특정한 길이를 가진다. 그러나 각각의 파티클은 구, 박스 타원체 같은 물체와 충돌함으로서 제약을 받을 수 있다. 완화 기술은 모든 제약이 동시에 만족할 수 있는 상태 쪽으로 집중하기 위해 이용되어진다. 이것은 그것이 충분한 것으로 간주될 때까지 반복해서 교대로 강요되어지는 것으로 이루어져있다. P1과 P2 사이의 거리 제약은 스프링 P1과 P2의 반응성이 되는 (s1, s2)를 가진 치환에 의해 강요되어진다. Dist(P1, P2)는 스프링 P1과 P2가 정지했을 때의 거리이다. 반응성은 두개의 파티클이 자유롭게 움직이는 경우 (0.5,0.5)이 되고 P1이나 P2가 고정되어 있다면 (0, 1)또는 (1, 0)이 된다.



〈그림 2〉 이웃된 파티클 사이의 스프링 제약

P1 by s1 (1 - d / Dist(P1, P2)) * (P2 - P1) ...(2)(a)

P2 by - s2 (1 - d / Dist(P1, P2)) * (P2 - P1) ...(2)(b)

파티클과 충돌물체 사이의 충돌은 파티클이 물체의 내부에 있는지 외부에 있는지에 따라 구분된다. 만약 내부에 있다면 파티클을 물체의 표면쪽으로 움직여서 충돌물체가 파티클의 현재 위치와 가까운지 여부를 확인함으로써 파악된다. 타원체의 경우 타원과 파티클의 접촉위치 중 가장 가까운 위치값을 계산하기 위해서는 반복적인 계산이 요구된다. 그래서, 이를 단순화하여 속도를 향상시키기 위하여 타원체의 중심부터 파티클의 현재의 위치까지에 이르는 라인과 타원체와의 교점까지 움직인다.

구멍을 가진 옷은 원래 옷 메쉬의 기하학적인 이미지를 계산하고, 정지시 스프링의 길이를 나타내는 파티클 사이의 길이를 사용하여 계산된다. 기하학 이미지는 주어진 메쉬를 에지 패스의 네트워크를 따라 절단함으로써 만들어진다. 이것은 기하학적으로 원반모양이 되고, 절단 후에 이미지의 각 픽셀의 꼭지점의 위치를 얻기위해 격자모양으로 결과를 샘플링 한다. 절단면을 따라 위치한 파티클들은 기하학 이미지의 경계면의 다른위치에서 복제되어진다. 그러한 파티클에 상응하는 복제품들은 평균에 도달함으로써 충돌 단계 전에 이동적분과 거리 강제 단계 그리고 그들의 위치가 일치되어지는 동안 독립적으로 시뮬레이션된다.

III. 옷 시뮬레이션을 위한 알고리즘

옷 시뮬레이션을 위한 알고리즘은 다음과 같다.

① 1단계 : 사용자에 의해 움직이거나 고정되어 있지 않은 모든 파티클 :
- 힘을 식(1)에 적용한다.

② 2단계 : 모든 완화 단계 :
- 2a 단계 : 모든 거리 제약 :
  - (2a)와 (2b) 등식을 적용한다.
- 2b 단계 : 모든 파티클 :
  - 만약 그 파티클이 seam(접합된 곳)의 부분이라면 :
    - 그것의 위치를 모든 상응하는 복제물의 평균 위치로 대체하라.

③ 모든 충돌 물체 :
- 만약 그 파티클이 내부에 있다면 :
  - 파티클을 물체 밖으로 움직여라

(W x H) 파티클을 구성하는 옷의 파티클의 위치는 3개의 순환하는 (W x H) 부동 소수점 구조에 저장되고 몇개의 화소 농도에 의해 성공적으로 처리된다. 한 구조는 과거 위치를 유지하지만 다른 구조는 현재위치를 유지하고 세번째 구조는 각 픽셀 쉐이더 패스 통화 후 결과를 저장하기 위해 이용된다. 픽셀 쉐이더 단계는 (W x H) 픽셀을 렌더링함으로써 수행된다.

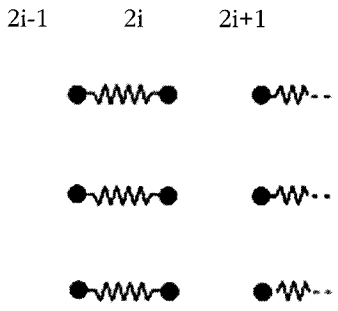
4개의 부동연산 요소 중 w- 요소는 파티클의 고정 여부와 접합된 파티클인지를 판단하고 만약 파티클들이 서로 접합되었다면 이 유사한 파티클을 복제를 통하여 파티클 구조에 속한 1개의 파티클을 부호화 하는데 사용된다.

시뮬레이션에 이용되는 모든 픽셀 쉐이더는 clothSim.fx에 있다.

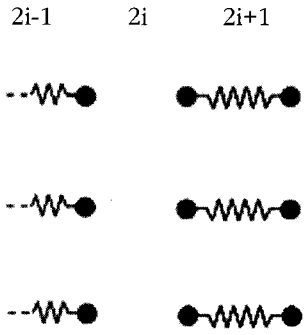
1단계는 ApplyForce에 의해 수행되어진다. ApplyForce는 고정되어있지 않거나 사용자가 움직이는 모든 파티클을 움직인다. 고정되어 있거나 사용자가 움직이는 파티클의 위치는 SetPosition과 TransformPosition에 의해 배치된다.

2a단계는 스프링의 분리된 줄과 칸을 다루는 픽셀 쉐이더에 의해 수행되어진다.

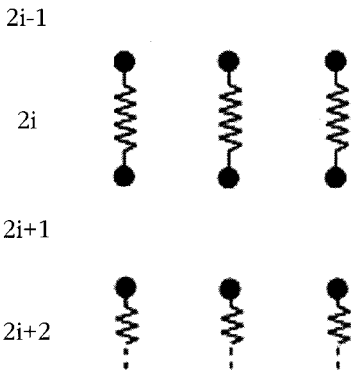
• SatisfySpringConstraintXSpringEven :



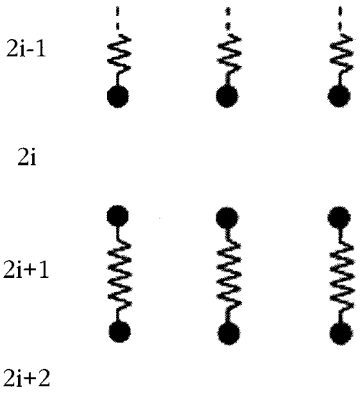
• SatisfySpringConstraintXSpringOdd :



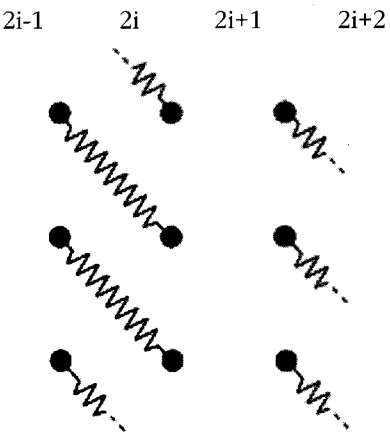
• SatisfySpringConstraintYSpringEven :



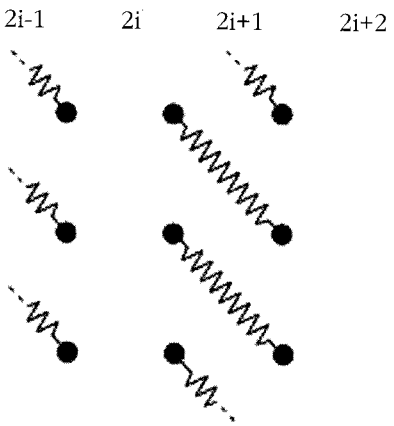
• SatisfySpringConstraintYSpringOdd :



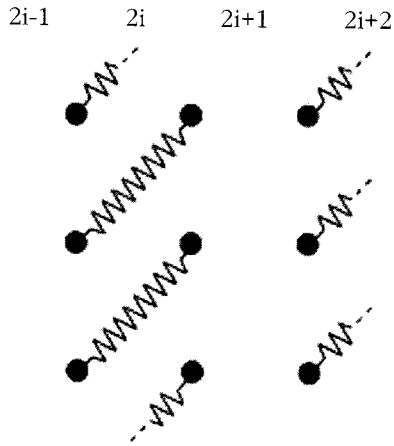
• SatisfySpringConstraintXYSpringDownEven :



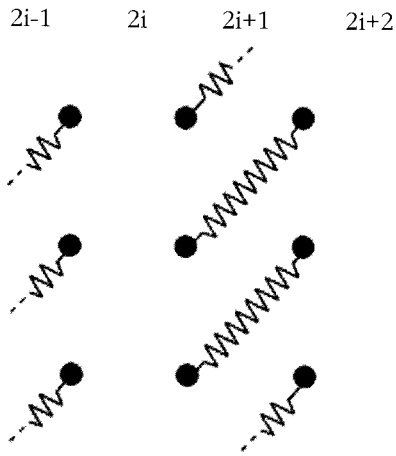
• SatisfySpringConstraintXYSpringDownOdd :



• SatisfySpringConstraintXYSpringUpEven :



• SatisfySpringConstraintXYSpringUpOdd :



반응성은 위의 8개의 pixel shader의 실행 동안 각각 한번 이용 되어지는 8개의 구조로 저장된다. 이러한 구조들은 사용자가 파티클을 선택하거나 선택하지 않을 때 혹은 옷를 자를 때마다 갱신되어진다. P1과 P2 사이의 스프링이 잘라질 때, 반응성은 두 위치에서 0에 놓인다.

정지시 스프링의 길이는 일정하며, 하나의 픽셀 쉐이더가 사용됨에 따라 갱신된 모든 스프링에서 동일한 경우에서의 상수나, 한 스프링과 다른 스프링이 다른 경우에서의 8개의 구조 또한 동일하다. 이러한 8개의 픽셀 쉐이더 패스는 다른 명령에 적용될 수 있다는 것을 주의하라.

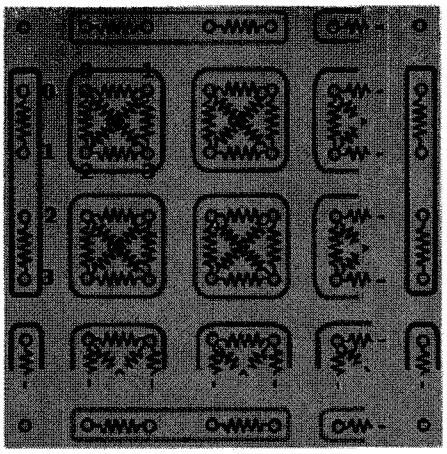
2b 단계는 SatisfySeamAndCollisionConstraints에 의해 수행된다. 충돌물체는 새로운 물체가 더해지거나 존재하는 물체가 제거되거나 움직일 때마다 갱신되는 구조에 저장된다.

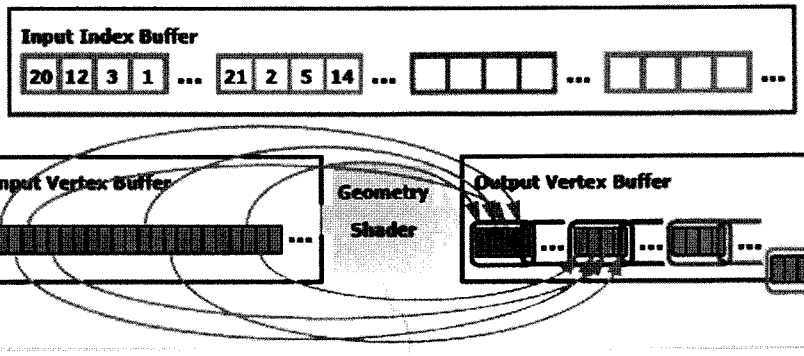
충돌 물체의 유형마다 하나의 구조가 있다. 일단 시뮬레이션이 행해지면, 마지막 파티클 위치를 포함하는 구조는 (W x H)꼭지점의 버텍스 버퍼의 꼭지점 위치를 배치하기 위해 vertex shader SimulatedVS에 의해 조사된다. 이 버텍스 버퍼는 옷을 전시하기 위해 pixel shader SimulatedPS를 이용하여 표현된다.

옷 절단은 다음의 단계에 의해 행해진다:

- 윈도우 포인트 A에서 다른 포인트 B까지 마우스를 드래그할 때 사용자가 자르는 옷 삼각형 중 하나를 결정한다.
- 상응하는 반응성을 갱신한다.
- 이러한 삼각형을 옷 인덱스 완충 기억장치로부터 제거한다.

위의 첫번째 단계는 컷 기법(Cut technique)에 의해 수행되고, 카메라의 위치와 A와 B가 투영된 장면의 두 점에 의해 형성된 삼각형과 함께 교차하는 모든 옷 삼각형으로 이루어져있다.





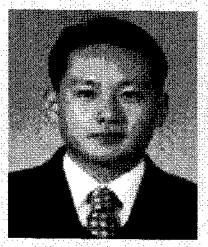

```
void SatisfySpringConstraintsGS(lineadj Particle particle[4],
                                inout PointStream<Particle> stream)
{
    switch (connectionConfiguration) {
        case 0:
            SatisfyDistanceConstraint(particle[0], RIGHT, particle[1]);
            SatisfyDistanceConstraint(particle[0], DOWN,  particle[2]);
            SatisfyDistanceConstraint(particle[2], RIGHT, particle[3]);
            // etc.
            break;
        case 1:
            SatisfyDistanceConstraint(particle[0], DOWN, particle[1]);
            SatisfyDistanceConstraint(particle[2], DOWN, particle[3]);
            break;
        case 2:
            ...
        case 4:
            break;
    }
    for (int i = 0; i < 4; ++i) stream.Append(particle[i]);
}
```

〈그림 3〉 스프링 제약의 Index Buffer Layout

참고문헌

[1] NVIDIA, "SDK White Paper: 옷", 2005
[2] NVIDIA, "White Paper: 옷 시뮬레이션", 2007
[2] Jakobsen, T, "Advanced character Physics". 2001 Game Developer's Conference
[3] Gu X, Gortler S, and Hoppe H, "Geometry Images."ACM SIGGRAPH 2002,355-361.

저자소개



김 종 진

1998년 2월 국민대학교 경영학과(경영학사)
2000년 2월 국민대학교 경영학과(경영정보전공)(경영학석사)
2004년 2월 홍익대학교 컴퓨터공학과 박사과정 수료
2005년 3월-현재 한국폴리텍1 서울강서대학 컴퓨터게임과 조교수

주관심 분야 : 게임 프로그래밍, 인공위성, 네트워크, 컴퓨터 교육

저자소개

강 호 석

2002년 2월 홍익대학교 전자계산학과 석사과정 졸업
2005년 2월 홍익대학교 컴퓨터공학과 박사과정 수료
2007년 현재 홍익대학교 컴퓨터 광학 박사과정

주관심 분야 : 모바일 컴퓨팅, 컴퓨터 보안, 컴퓨터 네트워크

이 면 재

1992년 홍익대학교 전자계산학과 졸업(학사)
1994년 홍익대학교 전자계산학과(석사)
2006년 홍익대학교 전자계산학과(박사)
2006-현재 남서울대학교 멀티미디어학과 교수

주관심 분야 : 게임 제작, 멀티미디어 통신, 멀티미디어 데이터 방송