사실적인 연기 시뮬레이션을 위한 이류항 계산의 수치적 개선

*장문희[O] *박수완 **김은주 *유관우
*경북대학교 컴퓨터공학과
**동명정보대학교 정보통신공학과
munhee3@nate.com[O], jpsuwan@naver.com, ejkim@tit.ac.kr, kwryu@yahoo.co.kr

Numerical Improvement of Advection Term for Realistic Smoke Simulation

*MunHee Chang[O] *SuWan Park **Eunju Kim *KwanWoo Ryu
*Dept. of Computer Engineering, Kyungpook National University
**Dept. of Information & Communication Engineering, Tongmyong University of Information Technology

요 약

자연 현상에서 나타나는 연기나 난류의 움직임을 사실적으로 시뮬레이션을 할 때 Navier-Stokes 방정식을 이용한다. 이 방정식을 이용한 구현은 방대한 연산량과 계산의 복잡성으로 인하여 실시간 시뮬레이션이 어렵다. 이 때문에 실시간 처리를 위하여 복잡한 수식을 근사화한다. 유체 시뮬레이션의 이류(advect) 과정에서 근사화를 위해 Semi-Lagrangian 방법을 이용할 때, 연기 시뮬레이션은 시간이 지남에 따라 밀도가 현저히 줄어들고 소규모의 소용돌이(small-scale vorticity) 현상이 급격히 감소하는 등의 수치적 소실이 발생한다. 본 논문에서는 이 문제를 해결하기 위해 이류항(advection term)을 계산할 때 새로운 수치적 방법을 제안한다. 본 논문에서는 이류항의 값을 구할 때, 현재 격자 주변의 값 중에서 다음 단계에 현재 격자의 위치로 오는 속도를 가진 격자를 찾아, 그 격자의 속도를 이류 속도 벡터로 활용한다. 이는 밀도와 소용돌이 현상의 수치적 소실을 줄여서 사실성을 높이고 실시간 처리도 가능하게 한다. 또한 본 논문에서는 GPU 구현을 통해 벡터 연산 등의 효율성을 높이며 시뮬레이션의 속도를 향상시킨다.

1. 서론

자연 현상에서 나타나는 연기나 난류의 움직임을 사실적으로 시뮬레이션을 할 때 주로 Navier-Stokes 방정식을 이용한다. 그러나 이 방정식을 이용한 구현은 비선형 방정식을 풀어야 하기 때문에 연산량이 많고 계산이 복잡해서 실시간 처리가 어렵고 수치적으로도 불안정하다. 1996년에 Foster와 Metaxas는 3차원 유체의 움직임을 시뮬레이션하기 위해 Navier-Stokes 방정식을 유한 차분(finite difference) 법을 이용하여 계산하였다[1]. 하지만 이 방법은 명시적 솔버(explicit solver)를 사용하였기 때문에 큰 시간 간격에 대해서는 시스템이 불안정하여 깨어지는 현상이 발생하였다. 이로 인해 대화형이고 실시간이 되는 유체 시뮬레이터에는 사용할 수가 없었다.

1999년 Stam은 안정성이 있으며 긴 시간 간격으로 시뮬레이션 가능한 Semi-Lagrangian라는 방법을 제안하였다[2]. 이 방법은 Navier-Stokes 방정식을 풀 때 이류항을 계산하는 부분을 근사화하는 것이다. 현재 격자의 이류항을 구할 때 현재 격자의 속도를 가지고 역 추적해서 추적한 위치의 격자로부터 속도를 가져오는 것이다. 하지만 Stam의 방식은 수치적 소실이 커서 연기의 섬세하고 역동적인 움직임을 다루는데 한계가 있었다. 이 문제는 2001년 Fedkiw의 연구에서 개선이 이루어졌다. 이 논문은 계산량이 많은 점성 Navier-Stokes 방정식 대신 가스 표현에 적합한 비점성의 유체표현을 위한 Euler식을 사용함으로써 기존 방법들보다 좀 더 넓은 간격의 격자에서 빠르고 효율적으로 연기를 표현할 수 있는 수치적 방법을 제안하였다[3]. 또한 성긴 격자에서 표현하기 어려운 작은 크기의 회전 특성을 모델링하기위해 와도 제한(vorticity confinement)항을 추가함으로써 이전 Stam의 연구에서 발생한 문제점을 해결하였다. 그러나 이 방법은 Stam의 Semi-Lagrangian 방법으로 이류항을 계산할 때 와도 제한 성분이 빠진 것으로 생각해서 추가한 것이다. 따라서 Semi-Lagrangian 방법 자체가 가지고 있는 수치적 소실의 문제를 해결하지는 못했다.

2005년 Hong은 연기의 와도를 유지하기 위해 이류항을 구하는 과정에서 발생하는 오차를 보정해서 수치적 소실을 줄이고, 와도의 이류를 첨가해서 모델링했다[4]. 이는 Stam의 Semi-Lagrangian 방법에다가 2차적인 정확도를 높인 방법으로 기존의 방법들보다 훨씬 사실성 있는 애니메이션을 보였다. 그러나 이 방법 또한 Semi-

Lagrangian 방법을 개선하기는 했으나 수치적 소실 문제를 해결하지는 못했다. 즉 역추적 방법을 통해서 이류항을 구할 때 현재 속도로 역추적하기 때문에 현재 속도가 0인 격자에서는 이류항의 값이 무조건 0이 된다. 또한 현재 속도로 역추적해서 찾은 위치의 속도가 0인 경우엔 소실이 더 커진다.

따라서 본 논문은 유체 시뮬레이션에서 이류항을 구할 때 기존의 Semi-Lagrangian 방법에서 생기는 소실을 최대한 줄이는 수치적 방법을 제한한다. 이는 이류항을 계산하기 위해 현재 격자 주변의 값 중에서 다음 단계에 현재 격자의 위치로 올 격자의 속도를 찾는 방법이다. 제안한 방법은 밀도와 와도가 이류할 때 발생하는 수치적 소실을 줄임으로써 연기 시뮬레이션의 사실성을 높인다. 연기는 비점성 유체이므로 Euler 식을 사용하며 이류항을 계산하는 부분을 제외하고는 기존의 Stam이 한 방식과 동일하게 구현한다. 즉, 비압축성 기체를 Navier-Stokes 수식기반으로 외부힘, 이류, 와도 제한힘 등을 적용하여 각 격자마다 반복적으로 수치연산을 수행한다. 이 과정들은 벡터 등의 반복 연산이 많으므로 GPU를 사용할 경우 속도가 향상된다[5]. 따라서 본 논문은 속도 개선을 위해서 GPU 상에서 구현한다.

본 논문의 구성은 다음과 같다. 2장에서는 기본적인 유체방정식 및 밀도와 와도를 유지하기 위한 이류의 역학적 모델을 기술하고, 3장에서는 실험 결과를 기술하고, 4장에서는 결론을 기술한다.

2. 본론

본 논문은 비압축성, 무점성의 기체를 가정하며, 기체의 시뮬레이션을 위해 Navier-Stokes 방정식을 사용한다. 격자의 중심에 속도와 밀도 값을 저장한다. 격자 모델은 1999년 Stam 방식을 따라 모델링한다[2].

2.1 기본적인 유체 방정식

속도 벡터장을 $\mathbf{u}=(u,v)$ 로 나타낼 때, Navier-Stokes 방정식은 다음과 같다.

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u}\cdot\nabla)\mathbf{u} - \frac{\nabla \mathrm{p}}{\rho} + \nu\nabla^2\mathbf{u} + \mathrm{f} \quad (1)$$

$$\nabla\cdot\mathbf{u} = 0 \quad (2)$$

여기서 f는 중력이나 부력과 같은 외력이다. P는 속도장에서의 압력이며 ρ는 밀도이다. ν는 점성계수이며 식 (2)는 비압축성일 때 유체의 질량이 보존되는 것을 의미한다. 무점성인 경우 Navier-Stokes 방정식은 확산항이 0인 무점성의 Euler 방정식이 되어 식 (3)과 같이 되며, 본 논문에서는 이 방정식을 사용한다.

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u}\cdot\nabla)\mathbf{u} - \frac{\nabla \mathrm{p}}{\rho} + \mathrm{f} \quad (3)$$

이 방정식을 푸는 방법을 더 쉽게 하기 위해서 일반적인 유체 시뮬레이션 처리과정을 생각해 보면 다음과 같다[2].

$$P\circ D\circ A\circ F$$

여기서 F는 외부힘, A는 이류항, D는 확산항, P는 정사형항이다. 이 과정이 반복되면서 유체 움직임의 시뮬레이션을 한다. 본 논문에서는 무점성 기체를 구현하기 때문에 점성과 관련된 확산항은 고려하지 않는다.

2.2 와도 제한(Vorticity confinement)

성긴 격자에서 표현하기 어려운 작은 소용돌이 특성을 모델링하기위해 와도 제한(소용돌이 제한 항)을 사용한다[3]. 와도 제한은 소용돌이와 같은 연기의 특징을 형성하는 힘을 생성한다. 그 연기의 움직임은 속도 u에 의해서 표현된다. 와도 w를 다음과 같이 나타낸다[6].

$$w = \nabla\times u \quad (4)$$

와도 w는 속도 장을 맴도(spin)는 회전의 중심축이다. 와도의 x, y 성분은 속도에 컬(cull)을 취해서 얻는다. 여기서 정규화 된 벡터장 N 을 얻을 수 있다.

$$\mathbf{N} = \frac{\eta}{|\eta|} \quad (\ \eta = \nabla|w|\ ) \quad (5)$$

식 (4)와 식 (5)로부터 시간 간격마다 속도 장을 회전하는데 필요한 힘의 방향과 크기를 결정할 수 있다. 와도 제한 힘(confinement force)은 다음과 같이 나타낸다.

$$f_{conf} = \epsilon h(\mathbf{N}\times w) \quad (6)$$

ϵ은 와도 제한 힘을 제어하는데 사용되는 상수 값이며, 식 (6)은 연기의 소용돌이치는 특징을 강조하는 힘을 제공한다.

2.3 소실을 줄이는 이류 방법

현재 속도 $\mathbf{u}_t$를 이용하여 이류항을 구하는 Semi-Lagrangian 방법은 현재 속도가 0일 경우, 실제와는 달리 이류항이 0이 된다. 즉 역추적 방법을 통해서 이류항을 구할 때 현재 속도로 역추적하기 때문에 현재 속도가 0인 격자에서는 이류항의 값이 무조건 0이 된다. 또한 현재 속도로 역추적해서 찾은 위치의 속도가 0인 경우엔 갑작스러운 값의 변화로 소실이 더 커진다. 이 문제를 해결하기 위해 본 논문에서는 현재 격자점 주변에 임계영역을 설정해서 격자점 중에서 현재 격자점으로 이류할 격자의 속도를 찾아 이류항으로 적용한다. 다음 단계에서 현재 격자정으로 이류하는 속도 $\mathbf{u}_{t-1}$를 구하는 과정은 다음과 같다.

이류항을 구하고하 하는 격자의 일정한 임계영역 내에 격자들이 각각 속도 $\mathbf{u}^n_{t-1}$가 있다고 가정하자. 구하고자 하는 격자의 현재 속력에 비례하는 임계영역을 그림 1과 같이 설정한다. 각 격자의 현재 속도의 크기 $|\mathbf{u}_t|$에 시간 간격 Δt를 곱한 값을 l이라 두자.

$$l = |\mathbf{u}_t|\Delta t \tag{7}$$

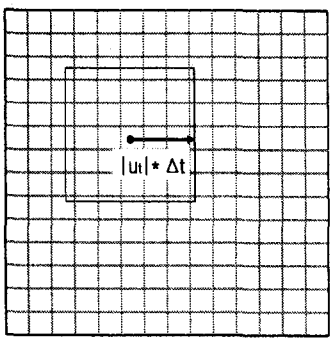


그림1 임계영역

한 변의 길이가 $2l$이고 이류항을 구하고자 하는 격자의 중심점이 임계 영역의 중심에 오도록 정사각형의 임계영역을 설정한다. 그러나 만일 l의 값이 사용자가 지정하는 임계영역 값보다 작으면 식 (8)을 사용한다. 이는 구하고자 하는 현재 격자의 속도 값이 작아서 주변을 찾는 범위가 좁아지는 것을 막기 위해서이다.

$$l = \alpha\Delta x \quad \text{if } l < \alpha\Delta x \tag{8}$$

Δx는 격자 한 개의 길이이고, α는 사용자가 지정하는 최소 임계영역을 위한 상수이다.

이렇게 임계영역의 범위를 지정한 후, 그 임계영역 내에 있는 격자들을 조사해서 현재 구하고자 하는 위치로 이류할 속도 $\mathbf{u}_{t-1}$를 갖고 있는 격자를 찾아낸다. 구체적인 과정은 다음과 같다.

그림2는 한 격자점의 임계영역을 나타낸다. 현재 이류항을 구하고자 하는 격자의 가운데 위치를 $\mathbf{P}_t$라고 하고, 임계영역 내에 있는 n번째 격자의 중심점의 위치를 $\mathbf{P}^n_t$, n번째 격자의 속도를 $\mathbf{u}^n_t$라고 두면, 각 격자의 중심점이 Δt시간 후에 이류하여 도착하는 위치는 $\tilde{\mathbf{P}}^n_{t-1}$이 된다. 이는 식 (9)를 통해 구한다.

$$\tilde{\mathbf{P}}^n_{t-1} = \mathbf{P}^n_t + \mathbf{u}^n_t\Delta t \tag{9}$$

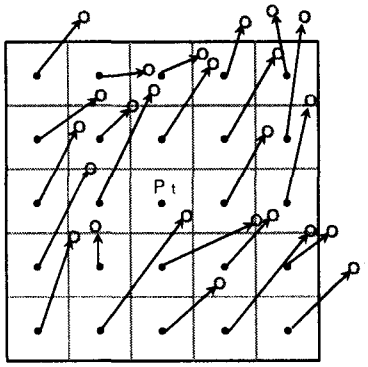


그림 2 Δt시간 후에 이류에 의해 이동한 점

$\tilde{\mathbf{P}}^n_{t-1}$ 중에 현재 위치 $\mathbf{P}_t$와의 거리가 가장 가까운 점 $\mathbf{P}_{min}$을 찾는다[그림 3]. 여기서 $\mathbf{P}^k_t$는 현재 구하고자 하는 격자의 위치로, 즉 이류해서 $\mathbf{P}_{min}$에 도달할 속도를 가지고 있는 격자의 위치이다.

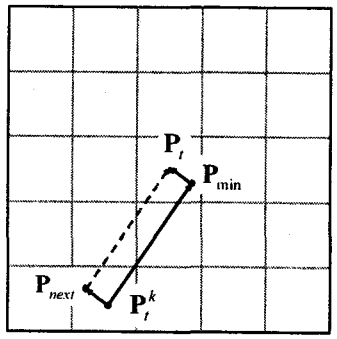


그림3 이류 할 속도를 찾는 과정

그러나 각 격자의 속도가 모두 격자의 중간 위치에서 정의 되어 있고, 이 속도를 기반으로 찾기 때문에 그림 3과 같이 $\overline{P_t P_{min}}$ 만큼 오차가 발생한다. 이 때 오차를 $\overrightarrow{err}$ 라고 둔다.

$$\overrightarrow{err} = \overrightarrow{P_t P_{min}} \tag{10}$$

오차를 보정하기 위해서 찾은 격자의 위치인 P_t^k에 $\overrightarrow{err}$ 를 더한다.

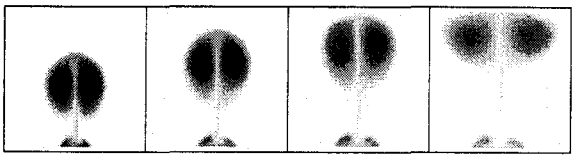
$$\overrightarrow{P_{next}} = \overrightarrow{P_t^k} + \overrightarrow{err} \tag{11}$$

결과적으로 P_{next} 위치에 있는 속도가 이류항의 속도 u_{t-1}이 된다.

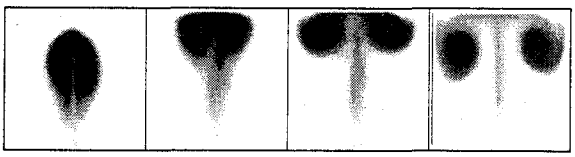
3. 실험결과

앞에서 제시한 방법을 사용하여 Pentium IV, 1G 메모리, 그래픽카드 geforce 5700(128M) 상에서 C++, cg, opengl로 구현하여 실험하였다. 격자수는 64×64이며 밀폐된 곳에서 일정량의 기체를 쏘아 올렸을 때의 동작을 표현했다. 아래 그림 4는 ϵ=0 즉, 와도 제한이 없을 경우, Semi-Lagrangian 방법과 제안한 방법을 비교한 결과이다. 밀폐된 공간에 있는 연기의 밀도는 계속 유지되어야 하지만 시간이 지남에 따라 Semi-Lagrangian 방법을 사용한 결과 흐려짐으로써 밀도의 소실이 있음을 보인다[그림4-(a)]. 반면에 같은 힘과 같은 밀도의 연기로 시작해서 제안한 방법을 사용한 결과 연기의 밀도가 거의 소실되지 않는 것을 알 수 있다[그림4-(b)]. 그림 5는 ϵ=0.155 정도의 와도 제한이 있을 때의 실험 결과이다. Semi-Lagrangian 방법을 사용한 그림 5-(a)을 보면 밀도는 비교적 유지가 되지만 와도가 빨리 사라진다. 제안한 방법을 사용한 그림 5-(b)를 보면 소규모의 와도가 계속 유지되는 것을 볼 수 있다.

네 가지 경우 모두 시뮬레이션 속도는 60 FPS 였다. 제안한 방법의 연산과정은 Semi-Lagrangian 방식보다 다소 추가한 부분이 있다. 하지만 GPU를 사용하기 때문에 프레임 수에는 큰 차이가 없다.

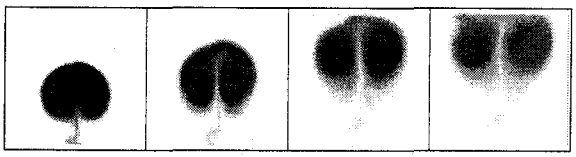


(a) Semi-Lagrangian 방법

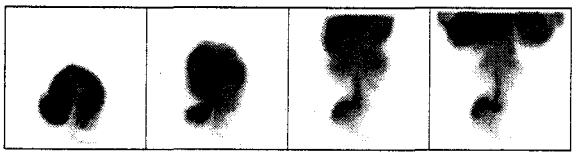


(b) 제안한 방법

그림 4 와도가 없을 경우 시뮬레이션 결과



(a) Semi-Lagrangian 방법



(b) 제안한 방법

그림 5 와도가 있을 경우 시뮬레이션 결과

4. 결론

본 논문에서는 비점성 유체역학 방정식과 와도 제한을 이용하여 역동적인 연기의 움직임을 구현하고자 했다. 좀 더 사실적인 연기 시뮬레이션을 위해 연기의 이류과정에서 격자 주변의 임계영역을 설정해서 임계영역내에 있는 격자들 중 현재의 위치로 이류할 속도를 가진 격자를 찾아내어, 그 격자의 속도를 이류항으로 사용하는 새로운 방법을 제안했다. 기존의 방법은 시간이 지남에 따라 밀도와 소규모의 와도가 너무 빨리 사라진다. 하지만 본 논문의 방법은 밀도의 소실량도 적으며 작은 크기의 소용돌이가 보존되어 역동적인 연기를 세밀하게 표현할 수 있다. 또한 GPU 상에서 구현함으로써 다소 추가된 계산 부분이 있을지라도 종래와 같은 속도를 유지하였다.

참고 문헌

[1] N. Foster and D. Metaxas, "Modeling the Motion of Hot, Turbulent Gas," in *Proceedings of ACM SIGGRAPH 1997*, pp.181-188, 1997

[2] Stam J. Stable fluids. In *Proceedings of ACM SIGGRAPH 1999*: pp.121-128,1999

[3] Fedkiw R, Stam J, Wann Jensen H. Visual simulation of smoke. In *Proceedings of ACM SIGGRAPH 2001*: pp.15-22, 2001

[4] Hong J-K, Kim J-H. Animating smoke with dynamic balance. *Computer Animation and Virtual Worlds*; 16:405-414, 2005

[5] Mark J. Harris. *Fast Fluid Dynamics Simulation on the GPU*. GPU Gems, 2004

[6] J. Steinhoff and D. Underhill. Modification of the Euler equations for "vorticity confinement": Application to the computation of interacting vortex rings. *Physics of Fluids*, 6(8):2738-2744,1994