속도의 초기값 추정을 사용한 Navier-Stokes 방정식 풀이 기법

(Solver for the Navier-Stokes Equations by using Initial Guess Velocity)

김 영 희 † 이 성 기 ††
(Younghee Kim) (Sungkee Lee)

요 약 본 논문은 물리적인 힘을 기반으로 유체의 흐름을 실시간으로 시뮬레이션하기 위하여 유체 의 흐름을 지배하는 Navier-Stokes 방정식에 대한 빠르고 정확한 풀이 기법을 제안한다. 본 논문에서는 Navier-Stokes 방정식에 있는 비선형 항의 속도에 대한 초기값을 Stokes 방정식의 해로써 추정한다. 주어진 비선형 미분방정식의 해에 근사하게 초기값을 추정함으로써 정확하고 안정적인 풀이 기법을 만들 수 있었다. 또한 유한차분법(finite difference method)의 암시적(implicit) 방법 중에서 방대한 계산량을 피할 수 있는 ADI(Alternating Direction Implicit) 방법을 사용함으로써 큰 시간 간격(time-step)에 대해서 시스템이 안정적이며 계산속도 또한 빠르다. 실험 결과들은 특히 연기, 구름과 같이 큰 레이놀드 수(Reynolds number)를 가지는 유체에 대해서 탁월한 성능을 보여주었다.

키워드 : 물리 기반 유체 시뮬레이션, Navier-Stokes 방정식, Stokes 방정식, ADI(Alternating Direction Implicit) 방법, 유한차분법

Abstract We propose a fast and accurate fluid solver of the Navier-Stokes equations for the physics-based fluid simulations. Our method utilizes the solution of the Stokes equation as an initial guess for the velocity of the nonlinear term in the Navier-Stokes equations. By guessing the initial velocity close to the exact solution of the given nonlinear differential equations, we can develop remarkably accurate and stable fluid solver. Our solver is based on the implicit scheme of finite difference methods, that makes it work well for large time steps. Since we employ the ADI method, our solver is also fast and has a uniform computation time. The experimental results show that our solver is excellent for fluids with high Reynolds numbers such as smoke and clouds.

Key words : Physics-based fluid simulation, Navier-Stokes Equations, Stokes Equation, ADI(Alternating Direction Implicit) method, Finite difference method

1. 서 론

최근까지도 물리 기반 유체 시뮬레이션은 계산량이 방대하여 실제적인 사용이 어려웠으나, 컴퓨터 하드웨어와 컴퓨터 그래픽스 기술의 급속한 발전에 힘입어 유체 시뮬레이션의 실시간 처리가 가능해지고 있다. 이에 따라 게임, 컴퓨터 애니메이션, 가상현실과 같은 그래픽 환경에서 유체 시뮬레이션에 대한 관심이 고조되고 있다.

† 정 회 원 : 경북대학교 컴퓨터과학과
alive_kim@hanmail.net
†† 종신회원 : 경북대학교 컴퓨터과학과 교수
sklee@knu.ac.kr
논문접수 : 2005년 3월 23일
심사완료 : 2005년 9월 5일

일반적인 물리 기반 유체 시뮬레이션에 대한 토대를 제공하기 위해서는 유체의 흐름을 지배하는 Navier-Stokes 방정식을 수치적으로 계산하여야 한다. 그러나 Navier-Stokes 방정식은 밀도 및 속도, 압력과 같은 기본 요소들의 관계가 복잡하고 비선형 방정식이기 때문에 풀이가 어려운 것으로 알려져 있다. 1950년대에 Harlow와 Welch는 유한차분법의 명시적(explicit) 방법에 기반하는 MAC(Marker-And-Cell) 방법을 제안하여 비압축성 유체를 위한 수많은 방법들에 영향을 미쳤다[1]. Kass와 Miller는 얕은 물 시뮬레이션을 위한 빠른 알고리즘을 제안하였다[2]. 이 방법은 유체 표면을 높이 요소만을 가지는 단일변수 함수로 가정하여 수식을 단순화하였다. 이러한 가정은 상호작용이 가능한 응용들을 만들어 낼 수 있었지만 방법의 활용 범위를 제한하였다.

Chen과 Lobo는 Navier-Stokes 방정식의 단순화된 형태를 2차원에서 풀이함으로써 Kass와 Miller의 방법보다 개선된 방법을 제안하였다[3]. 이 방법 역시 유체는 단일한 높이를 가지는 것으로 가정하고, 유체 표면의 높이를 단지 압력만으로 계산함으로써 활용 범위가 제한적이다. Foster와 Metaxas는 복셀(voxel) 기반 환경에서의 Navier-Stokes 방정식의 풀이에 대한 가능성을 보여주었다[4]. 그러나 풀이 방법의 안정성을 보장하기 위해서는 작은 시간 간격을 취해야 하는 제약 때문에 실시간 유체 시뮬레이션에 적합하지 않다. 1999년에 Stam은 현재 컴퓨터 그래픽스 분야에서 가장 널리 사용되고 있는 풀이 기법을 제안하였다[5]. 그의 방법은 상대적으로 빠르고 항상 시스템의 안정성을 보장할 수 있어서 액체 또는 기체에 이 방법을 적용했던 많은 연구들이 좋은 결과를 보여주었다[6,7]. 그러나 Stam 방법은 Semi-Lagrangian 기법을 사용하기 때문에 시뮬레이션 결과들이 실제 유체보다 빠르게 소멸하고, 때때로 심각한 수치적 손실 문제를 야기 시킬 수 있다. 최근에 Fedkiw와 Stam은 수치적 손실을 보완하기 위하여 와류도 제한(vorticity confinement) 항을 추가하여 소실된 유체의 특성을 되살리는 방법을 제안하였다[8].

본 연구에서는 실시간 물리 기반 유체 시뮬레이션을 위한 Navier-Stokes 방정식의 풀이에 초점을 맞추었다. 실시간 물리 기반 시뮬레이션을 위해서는 두 가지 중요한 요소가 고려되어야 한다. 첫째, 시뮬레이션의 매 시간 단위에서의 계산이 빨라야 하고 둘째, 정확성과 안정성을 보장하면서 큰 시뮬레이션 시간 간격(이하 "시간 간격"이라 한다)을 취할 수 있어야 한다. 본 논문에서는 이러한 요구 조건들을 만족하는 매우 효율적인 풀이 기법을 제안한다. 본 논문의 방법은 유한차분법(finite difference method)의 암시적(implicit) 방법을 기반으로 하며, 선형 시스템 풀이의 방대한 계산을 피하기 위하여 ADI(Alternating Direction Implicit) 방법을 사용하였다. 이 방법은 각 시간 단위에서의 계산이 빠르고 시뮬레이션 변수들에 상관없이 계산 시간이 일정하다. 본 논문의 주요 공헌은 Navier-Stokes 방정식에 있는 비선형 항의 속도에 대한 초기값을 Navier-Stokes 방정식에서 비선형 항을 포함하지 않는 Stokes 방정식의 해로 추정하는 것이다. 주어진 비선형 미분 방정식의 해에 근사하게 초기값을 추정함으로써 매우 정확하고 안정적인 유체 풀이 기법을 만들 수 있었다. 특히 본 논문의 풀이 기법은 연기, 구름과 같이 큰 레이놀드 수를 가지는 유체의 시뮬레이션에서 뛰어난 결과를 보여주었다. 본 논문의 풀이 기법은 이론적으로는 어떠한 시간 간격에 대해서도 시스템이 항상 안정적임에도 불구하고 실험적으로는 제한을 가지며 이러한 제한은 경계 조건과 관련이 있을 것으로 예상된다. 본 논문에서 아직까지는 안정성을 위한 기준을 제공하지 못하지만 본 논문의 풀이 기법은 큰 시간 간격에 대해서 매우 안정적이다.

본 논문의 구성은 다음과 같다. 제2절에서 유체 역학에서의 지배 방정식과 ADI 방법에 대해 설명한다. 제3절에서는 이산화 방법과 본 논문에서 제안하는 풀이 기법을 유도 과정과 함께 설명한다. 제4절에서는 시뮬레이션 결과를 통하여 본 논문에서 제안한 풀이 기법의 유용성을 설명한다. 제5절에서는 본 논문의 결론을 내리고 향후 연구 과제에 대하여 설명한다.

2. 관련 연구

2.1 Navier-Stokes 방정식과 Stokes 방정식

유체의 흐름은 유체 역학의 기본 방정식인 연속방정식(equation of continuity)과 Navier -Stokes 방정식에 의해 지배된다[9]. 연속방정식은 질량 보존의 법칙(mass conservation law)으로부터, Navier-Stokes 방정식은 운동량 보존의 법칙(momentum conservation law)으로부터 유도된다. 연속 방정식을 다음과 같이 벡터 형식으로 쓸 수 있다.

$$\frac{1}{\rho}\frac{D\rho}{Dt}+(\nabla\cdot\mathbf{v})=0$$

여기서 $\mathbf{v}=(u,v,w)$는 유체의 속도를 나타내고 ρ는 유체의 밀도를 나타낸다. 비압축성 유체에서 밀도는 일정하므로 다음 식이 성립한다.

$$\frac{1}{\rho}\frac{D\rho}{Dt}=0$$

따라서 비압축성 유체에 대한 연속 방정식은 다음과 같다.

$$\nabla\cdot\mathbf{v}=0$$

외부에서 유체에 가해지는 힘을 고려한 Navier-Stokes 방정식은 다음과 같다.

$$\frac{\partial\mathbf{v}}{\partial t}=-(\mathbf{v}\cdot\nabla)\mathbf{v}+\frac{1}{Re}\Delta\mathbf{v}-\nabla p+\mathbf{f}$$

여기서 p는 유체의 압력을, Re는 레이놀드 수를, $\mathbf{f}$는 외부 힘을 나타낸다. 레이놀드 수는 유체의 점성도를 나타내는 매개변수로서 이 값이 작을수록 점성이 강한 유체를 나타낸다.

일정한 밀도를 가지는 유체가 일반적인 유체들의 흐름을 잘 근사함은 알려져 있다. 본 논문에서는 다음과 같이 정의되는 비압축성 Navier-Stokes 방정식을 사용하여 유체의 흐름을 근사한다.

$$\frac{\partial\mathbf{v}}{\partial t}=-(\mathbf{v}\cdot\nabla)\mathbf{v}+\frac{1}{Re}\Delta\mathbf{v}-\nabla p+\mathbf{f} \qquad (1)$$

$$\nabla \cdot \mathbf{v} = 0 \tag{2}$$

이 식에는 벡터(vector) 함수 $\mathbf{v}$와 스칼라(scalar) 함수 p가 미지수이며 나블라(nabla) 연산자의 세 가지 형태가 존재한다.

◦ 기울기(gradient) : $\nabla \mathbf{v} = \left(\frac{\partial \mathbf{v}}{\partial x}, \frac{\partial \mathbf{v}}{\partial y}, \frac{\partial \mathbf{v}}{\partial z} \right)$

◦ 확산(divergence) : $\nabla \cdot \mathbf{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}$

◦ 라플라시안(Laplacian) : $\Delta \mathbf{v} = \nabla \cdot \nabla \mathbf{v}$

Stokes 방정식은 식 (3)과 같이 Navier-Stokes 방정식에서 비선형 항을 포함하지 않는 방정식이다. Stokes 방정식은 Navier-Stokes 방정식과 매우 유사한 형태를 가지기 때문에 이 방정식의 해는 Navier-Stokes 방정식에서 초기 추정치로 사용하기에 적당하다.

$$\frac{\partial \mathbf{v}}{\partial t} = \frac{1}{Re} \Delta \mathbf{v} - \nabla p + \mathbf{f} \tag{3}$$

2.2 ADI 방법

ADI방법은 1950년대에 Peaceman과 Rachford에 의해서 포물선 방정식의 풀이를 위해 제안된 이래로 다양한 분야에서 널리 사용되고 있다[10]. 이 방법은 병렬계산이 가능한 3중 대각(tri-diagonal) 행렬들의 풀이로 구성되며, 선형 시스템 풀이에 요구되는 방대한 계산량을 줄일 수 있다. 그러므로 이 방법은 계산이 빠르면서도 행렬의 계수에 관계없이 일정한 계산 시간을 가진다. 다음과 같은 2차원 대류-확산(advection-diffusion) 방정식을 살펴보자.

$$\frac{\partial \mathbf{F}}{\partial t} + \left(\alpha \frac{\partial \mathbf{F}}{\partial x} + \beta \frac{\partial \mathbf{F}}{\partial y} \right) - \nu \Delta \mathbf{F} = \mathbf{0}$$

여기서 α, β, ν는 상수로 가정한다. ADI 방법은 다음과 같이 두 단계로 구성된다.

$$\frac{2}{\Delta t}\left(\tilde{\mathbf{F}} - \mathbf{F}^n \right) + \alpha \Delta_x^0 \tilde{\mathbf{F}} + \beta \Delta_y^0 \mathbf{F}^n - \nu \left(\Delta_{xx} \tilde{\mathbf{F}} + \Delta_{yy} \mathbf{F}^n \right) = \mathbf{0}$$

$$\frac{2}{\Delta t}\left(\mathbf{F}^{n+1} - \tilde{\mathbf{F}} \right) + \alpha \Delta_x^0 \tilde{\mathbf{F}} + \beta \Delta_y^0 \mathbf{F}^{n+1} - \nu \left(\Delta_{xx} \tilde{\mathbf{F}} + \Delta_{yy} \mathbf{F}^{n+1} \right) = \mathbf{0}$$

보조 값 $\tilde{\mathbf{F}}$는 중간 타임 $(n+1/2)\Delta t$에서의 근사해이고 미분 연산자는 다음과 같다.

$$\Delta_x^0 \mathbf{F}_{i,j} = \frac{2}{\Delta x}\left(\mathbf{F}_{i+1,j} - \mathbf{F}_{i-1,j} \right), \quad \Delta_y^0 \mathbf{F}_{i,j} = \frac{2}{\Delta y}\left(\mathbf{F}_{i,j+1} - \mathbf{F}_{i,j-1} \right)$$

$$\Delta_{xx} \mathbf{F}_{i,j} = \frac{\left(\mathbf{F}_{i+1,j} - 2\mathbf{F}_{i,j} + \mathbf{F}_{i-1,j} \right)}{2\Delta x^2}, \quad \Delta_{yy} \mathbf{F}_{i,j} = \frac{\left(\mathbf{F}_{i,j+1} - 2\mathbf{F}_{i,j} + \mathbf{F}_{i,j-1} \right)}{2\Delta y^2}$$

3. 제안하는 방법

이 절에서는 본 논문에서 제안하는 Navier-Stokes 방정식에 대한 수치적 풀이 방법을 유도 과정과 함께 소개한다.

시간 간격(Δt)에 제한을 가하지 않기 위하여 유한차분법의 암시적 방법을 사용한다. Navier-Stokes 방정식을 시간에 대해 이산화(discretization)하면 다음과 같다.

$$\frac{\mathbf{v}^{n+1} - \mathbf{v}^n}{\Delta t} = -\left(\mathbf{v}^{n+1} \cdot \nabla \right) \mathbf{v}^{n+1} + \frac{1}{Re} \Delta \mathbf{v}^{n+1} - \nabla p^{n+1} + \mathbf{f},$$

$$\nabla \cdot \mathbf{v}^{n+1} = 0$$

위 식으로부터 비선형 항을 선형화 함으로써 비압축성 Oseen형 방정식을 얻는다. 비선형 미분 방정식에 대한 초기값이 해에 근사하게 취해진다면 2차 수렴(quadratic convergence)이 보장된다는 것은 수학 분야에서 잘 알려진 사실이다[11]. Stokes 방정식은 Navier-Stokes 방정식에서 비선형 항만을 포함하지 않기 때문에 두 방정식의 해는 매우 유사하다. 따라서 다음 수식과 같이 Stokes 방정식의 해를 Navier-Stokes 방정식의 비선형 항의 속도에 대한 초기 추정치로 사용한다.

$$\frac{\hat{\mathbf{v}} - \mathbf{v}^n}{\Delta t} = \frac{1}{Re} \Delta \hat{\mathbf{v}} - \nabla p^n + \mathbf{f} \tag{4}$$

$$\frac{\mathbf{v}^{n+1} - \mathbf{v}^n}{\Delta t} = -\left(\hat{\mathbf{v}} \cdot \nabla \right) \mathbf{v}^{n+1} + \frac{1}{Re} \Delta \mathbf{v}^{n+1} - \nabla p^{n+1} + \mathbf{f} \tag{5}$$

$$\nabla \cdot \mathbf{v}^{n+1} = 0 \tag{6}$$

식 (5)와 (6)에서 연결되어 있는 미지수 $(\mathbf{v}, p)$를 분리하기 위하여, Chorin과 Temam에 의해 제안된 사영법(projection method or fractional step method)을 사용한다[12,13]. 사영법을 이용한 많은 풀이 기법들이 제안되고 있으며 오차 분석에 대한 연구 또한 활발히 이루어지고 있다[14]. 식 (5)와 (6)에 사영법을 적용한 암시적 방법은 다음과 같다.

$$\frac{\tilde{\mathbf{v}}^{n+1} - \mathbf{v}^n}{\Delta t} = -\left(\hat{\mathbf{v}} \cdot \nabla \right) \tilde{\mathbf{v}}^{n+1} + \frac{1}{Re} \Delta \tilde{\mathbf{v}}^{n+1} + \mathbf{f}$$

$$\frac{\mathbf{v}^{n+1} - \tilde{\mathbf{v}}^{n+1}}{\Delta t} = -\nabla p^{n+1}, \quad \nabla \cdot \mathbf{v}^{n+1} = 0$$

본 논문에서 제안하는 풀이 기법은 세 단계로 구성되며, 다음과 같이 요약될 수 있다.

◦ 단계 1 : Stokes 방정식의 해로 초기 속도 추정

$$\frac{\hat{\mathbf{v}} - \mathbf{v}^n}{\Delta t} = \frac{1}{Re} \Delta \hat{\mathbf{v}} - \nabla p^n + \mathbf{f} \tag{7}$$

◦ 단계 2 : ADI 방법에 의한 중간 단계의 속도 계산

$$\frac{\tilde{\mathbf{v}}^{n+1} - \mathbf{v}^n}{\Delta t} = -\left(\hat{\mathbf{v}} \cdot \nabla \right) \tilde{\mathbf{v}}^{n+1} + \frac{1}{Re} \Delta \tilde{\mathbf{v}}^{n+1} + \mathbf{f} \tag{8}$$

◦ 단계 3 : 중간 단계의 속도를 무발산계(divergence free field)로 사영

$$\frac{\mathbf{v}^{n+1}-\tilde{\mathbf{v}}^{n+1}}{\Delta t}=-\nabla p^{n+1},\quad \nabla\cdot\mathbf{v}^{n+1}=0 \tag{9}$$

본 논문에서는 설명을 단순화하기 위하여 계산 영역을 2차원으로 제한한다. 계산 영역을 직각 격자(rectangular grid)로 나누고 엇갈린 격자(staggered grid) 방법을 사용한다. 시간 $n\Delta t$에서 압력 $p^n_{I,J}$은 셀(I,J)의 중심에서 정의되고 속도 $\mathbf{v}^n_{I,J}=\left(u^n_{i,J},v^n_{I,j}\right)$의 각 성분들은 셀(cell)의 경계에서 정의된다. 공간에 대한 이산화를 단순하게 하기 위하여 x축과 y축 방향의 격자 간격이 동일하다고 가정한다. 즉, $\Delta x=\Delta y=h$ 이다(그림 1 참조).

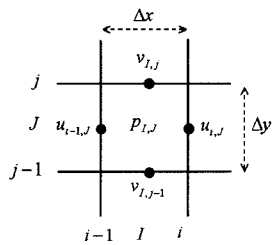


그림 1 엇갈린 격자

3.1 단계 1 : Stokes 방정식의 해로 초기 속도 추정

식 (7)은 다음과 같이 공간 미분으로 이산화 된다.

$$\frac{\hat{u}_{i,J}-u^n_{i,J}}{\Delta t}=\frac{1}{Re}\left(\frac{\hat{u}_{i+1,J}-2\hat{u}_{i,J}+\hat{u}_{i-1,J}}{h^2}+\frac{\hat{u}_{i,J+1}-2\hat{u}_{i,J}+\hat{u}_{i,J-1}}{h^2}\right)-\frac{p^n_{I,J}-p^n_{I-1,J}}{h}+f_{I,J} \tag{10}$$

$$\frac{\hat{v}_{I,j}-v^n_{I,j}}{\Delta t}=\frac{1}{Re}\left(\frac{\hat{v}_{I+1,j}-2\hat{v}_{I,j}+\hat{v}_{I-1,j}}{h^2}+\frac{\hat{v}_{I,j+1}-2\hat{v}_{I,j}+\hat{v}_{I,j-1}}{h^2}\right)-\frac{p^n_{I,J}-p^n_{I,J-1}}{h}+f_{I,J} \tag{11}$$

이 식은 미지수 $\hat{u}_{i,J}$와 $\hat{v}_{I,j}$에 대한 선형시스템이다. 이 행렬들은 희박 행렬(sparse matrix)이기 때문에 반복법인 SOR(Successive OverRelaxation) 방법으로 쉽게 풀 수 있다. 이 단계에서 속도 $\hat{\mathbf{v}}=(\hat{u},\hat{v})$는 단지 초기치 추정을 위한 속도일 뿐 시간적으로 중간 단계의 속도가 아니기 때문에 Stokes 방정식의 해를 정확히 구할 필요가 없다. 더욱 흥미로운 것은 반복법에서 반복횟수가 일정 횟수를 넘으면 결과에 큰 영향을 미치지 않는다는 것이다. 본 논문에서 수행한 시뮬레이션에서는 반복횟수를 3~10 범위에서 고정하여 사용하였고 이는 좋은 결과를 얻기에 충분하였다.

3.2 단계 2 : ADI 방법에 의한 중간 단계의 속도 계산

ADI 방법을 Navier-Stokes 방정식 풀이에 적용하였던 방법은 1980년대에 Peyret에 의해 제안되었다[15]. 본 논문에서는 비선형 항의 암시적 부분에 풍상(upwind) 미분 연산자를 사용하여 3중 대각 행렬을 만드는 Peyret의 아이디어를 응용하여 식 (8)에 ADI 방법을 적용한다. 그러나 Peyret 방법에서 사용하는 인공안정(artificial stabilizer) 단계는 포함하지 않는다. ADI 방법에 의해서 식 (8)이 다음과 같이 두 단계로 나누어진다.

$$\frac{2}{\Delta t}\left(\tilde{\mathbf{v}}_{I,J}-\mathbf{v}^n_{I,J}\right)=-\hat{u}\Delta^*_x\tilde{\mathbf{v}}_{I,J}-\hat{v}\Delta^0_y\mathbf{v}^n_{I,J}+\frac{1}{Re}(\Delta_{xx}\tilde{\mathbf{v}}_{I,J}+\Delta_{yy}\mathbf{v}^n_{I,J})+\mathbf{f}_{I,J} \tag{12}$$

$$\frac{2}{\Delta t}\left(\tilde{\mathbf{v}}^{n+1}_{I,J}-\tilde{\mathbf{v}}_{I,J}\right)=-\hat{u}\Delta^0_x\tilde{\mathbf{v}}_{I,J}-\hat{v}\Delta^*_y\tilde{\mathbf{v}}^{n+1}_{I,J}+\frac{1}{Re}(\Delta_{xx}\tilde{\mathbf{v}}_{I,J}+\Delta_{yy}\tilde{\mathbf{v}}^{n+1}_{I,J})+\mathbf{f}_{I,J} \tag{13}$$

여기서 $\hat{\mathbf{v}}=(\hat{u},\hat{v})$은 단계 1로부터 주어지고 풍상 미분 연산자는 다음과 같이 정의된다.

$$\Delta^*_x=\frac{1}{2}\left[\left(1-\varepsilon_{\hat{u}}\right)\Delta^+_x+\left(1+\varepsilon_{\hat{u}}\right)\Delta^-_x\right],\quad \varepsilon_{\hat{u}}=sign(\hat{u})$$

$$\Delta^*_y=\frac{1}{2}\left[\left(1-\varepsilon_{\hat{v}}\right)\Delta^+_y+\left(1+\varepsilon_{\hat{v}}\right)\Delta^-_y\right],\quad \varepsilon_{\hat{v}}=sign(\hat{v})$$

Δ^+와 Δ^-는 각각 전진(forward) 연산자와 후진(backward) 연산자이다. 미분 연산자를 적용하면 속도 u 성분의 방정식은 다음과 같다.

$$\frac{2}{\Delta t}\left(\tilde{u}_{i,J}-u^n_{i,J}\right)=-\frac{\hat{u}_{i,J}}{2h}\left[\left(1-\varepsilon_{\hat{u}}\right)\left(\tilde{u}_{i+1,J}-\tilde{u}_{i,J}\right)+\left(1+\varepsilon_{\hat{u}}\right)\left(\tilde{u}_{i,J}-\tilde{u}_{i-1,J}\right)\right]-\frac{\hat{v}_{i,J}}{2h}\left(u^n_{i,J+1}-u^n_{i,J-1}\right)+\frac{1}{Re}\left(\frac{\tilde{u}_{i+1,J}-2\tilde{u}_{i,J}+\tilde{u}_{i-1,J}}{h^2}+\frac{u^n_{i,J+1}-2u^n_{i,J}+u^n_{i,J-1}}{h^2}\right)+f_{I,J} \tag{14}$$

$$\frac{2}{\Delta t}\left(\tilde{u}^{n+1}_{i,J}-\tilde{u}_{i,J}\right)=-\frac{\hat{u}_{i,J}}{2h}\left(\tilde{u}_{i+1,J}-\tilde{u}_{i-1,J}\right)-\frac{\hat{v}_{i,J}}{2h}\left[\left(1-\varepsilon_{\hat{v}}\right)\left(\tilde{u}^{n+1}_{i,J+1}-\tilde{u}^{n+1}_{i,J}\right)+\left(1+\varepsilon_{\hat{v}}\right)\left(\tilde{u}^{n+1}_{i,J}-\tilde{u}^{n+1}_{i,J-1}\right)\right]+\frac{1}{Re}\left(\frac{\tilde{u}_{i+1,J}-2\tilde{u}_{i,J}+\tilde{u}_{i-1,J}}{h^2}+\frac{\tilde{u}^{n+1}_{i,J+1}-2\tilde{u}^{n+1}_{i,J}+\tilde{u}^{n+1}_{i,J-1}}{h^2}\right)+f_{I,J} \tag{15}$$

여기서 $\hat{v}_{i,J}$ 은 다음과 같다.

$$\hat{v}_{i,J}=\frac{\hat{v}_{I,j}+\hat{v}_{I,j-1}+\hat{v}_{I+1,j}+\hat{v}_{I+1,j-1}}{4}$$

식 (14)는 $(I_{\max}-1)\times J_{\max}$ 개의 미지수 $\tilde{u}_{i,J}$ 를 포함하는 방정식으로 다음과 같은 선형 시스템으로 표현된다.

$$a_{i,J}\tilde{u}_{i-1,J}+b_{i,J}\tilde{u}_{i,J}+c_{i,J}\tilde{u}_{i+1,J}=d_{i,J},$$

$$i=1,\cdots,I_{max}-1,\quad J=1,\cdots,J_{max}$$

여기서 $a_{i,J}$, $b_{i,J}$, $c_{i,J}$ 는 다음과 같이 계산되고 $d_{i,J}$ 는 이미 알고 있는 $u^n_{i,J}$, $\hat{v}_{i,J}$, $f_{I,J}$ 에 의해 결정된다.

$$a_{i,J}=\frac{-(1+\varepsilon_{\hat{u}})\hat{u}_{i,J}}{2h}-\frac{1}{Reh^2},$$

$$b_{i,J}=\frac{2}{\Delta t}+\frac{\varepsilon_{\hat{u}}\hat{u}_{i,J}}{h}+\frac{2}{Reh^2},$$

$$c_{i,J}=\frac{(1-\varepsilon_{\hat{u}})\hat{u}_{i,J}}{2h}-\frac{1}{Reh^2}$$

행렬을 행이나 열을 기준으로 나누면 다음과 같은 행렬들을 얻는다.

$$J=1,\ldots,J_{\max},$$

$$\begin{bmatrix} b_{1,J} & c_{1,J} & & \\ a_{2,J} & b_{2,J} & c_{2,J} & \\ & \ddots & \ddots & \ddots \\ & & a_{I_{\max}-1,J} & b_{I_{\max}-1,J} \end{bmatrix}\begin{bmatrix} \tilde{u}_{1,J} \\ \tilde{u}_{2,J} \\ \vdots \\ \tilde{u}_{I_{\max}-1,J} \end{bmatrix}=\begin{bmatrix} d_{1,J} \\ d_{2,J} \\ \vdots \\ d_{I_{\max}-1,J} \end{bmatrix} \tag{16}$$

위의 행렬은 3중 대각 행렬이고, 식 (17)에서 보여주는 것처럼 엄격히 대각 성분 우세한 (strictly diagonal dominant) 성질을 가진다(부록 참조).

$$|b_{i,J}|\geq|a_{i,J}|+|c_{i,J}| \tag{17}$$

따라서 어떠한 크기의 시간 간격도 취할 수 있고 3중 대각 행렬을 위해 제안된 계산 복잡도가 $O(N)$을 가지는 Thomas 방법으로 쉽게 풀 수 있다[16].

풍상 미분 연산자의 이산화 오차는 1차의 정확도 (first-order accuracy)를 가진다. 1차의 정확도는 공간 영역에서 격자 간격을 반으로 줄이면 오차가 1/2로 감소함을 의미하고, 2차의 정확도는 오차가 1/4로 감소함을 의미한다. Peyret의 방법은 정확도를 높이기 위해서 다음과 같이 정의되는 또 다른 1차의 정확도를 가지는 풍하(downwind) 미분 연산자를 명시적 부분에 사용하였다.

$$\Delta_x^{**}=\frac{1}{2}\left[(1+\varepsilon_{\hat{u}})\Delta_x^{+}+(1-\varepsilon_{\hat{u}})\Delta_x^{-}\right]$$

반면에 본 논문에서는 명시적 부분에 2차의 정확도를 가지는 중앙(central) 미분 연산자를 적용하였다. 풍상 미분 연산자와 중앙 미분 연산자를 사용하는 것이 1차의 정확도를 가지는 두 개의 미분 연산자를 사용하는 것보다 오차를 줄이면서도 선형 시스템의 안정성(stability)을 상당히 높일 수 있음을 관찰하였다.

지금까지 속도 $\tilde{u}_{i,J}$ 에 대한 계산 과정만을 설명하였다. 식 (15)의 속도 $\tilde{u}^{n+1}_{i,J}$ 에 대한 방정식들은 식 (16)과 유사하며 y축 속도 성분인 $\tilde{v}_{I,j}$ 와 $\tilde{v}^{n+1}_{I,j}$ 를 위한 방정식들은 속도 $\tilde{u}_{i,J}$, $\tilde{u}^{n+1}_{i,J}$ 의 방정식과 동일한 형식을 가진다.

3.3 단계 3 : 중간 단계의 속도를 무발산계로 사상

이 단계는 압력에 대한 Poisson 방정식을 계산하고 중간 단계의 속도 $\tilde{v}^{n+1}$를 무발산 조건을 만족하도록 갱신하는 두 단계로 구성된다. 식 (9)에 발산 연산을 적용하면 다음과 같이 압력에 대한 Poisson 방정식이 된다.

$$\Delta p^{n+1}=\frac{1}{\Delta t}\left(\nabla\cdot\tilde{\mathbf{v}}^{n+1}\right)$$

이 선형 시스템 또한 희박 행렬(sparse matrix)이므로 SOR 반복법을 사용하였다. 반복법에서 반복 계산은 수렴 조건이 만족될 때까지 연산이 수행된다. 즉, 오차(residual)가 특정 허용치 (tolerance) eps 보다 작아졌을 때 종료하는 것으로, 본 논문에서는 압력의 크기를 포함하는 상대적 허용치(relative tolerance) $eps\times\|p\|$ 을 사용하였다. 시간 $(n+1)\Delta t$ 에서의 압력이 계산되면 중간 단계의 속도는 다음의 수식에 따라 갱신되어 무발산 조건을 만족하게 된다.

$$\mathbf{v}^{n+1}=\tilde{\mathbf{v}}^{n+1}-\Delta t\Delta p^{n+1}$$

최종적으로 계산된 속도 $\mathbf{v}^{n+1}$와 압력 $\mathbf{p}^{n+1}$이 Navier-Stokes 방정식의 해이다.

4. 실험 결과 및 분석

컴퓨터 그래픽스의 유체 시뮬레이션 분야에서 Navier-Stokes 방정식의 풀이를 위해 가장 많이 사용되는 방법은 MAC 방법과 Stam 방법이다. Harlow와 Welch에 의해 제안된 MAC 방법은 명시적 방법이므로, 이 방법의 결과가 정확하다 할지라도 실시간 시뮬레이션에는 적당하지 않다. 서론에서 언급하였듯이 Stam 방법은 실시간 유체 시뮬레이션을 위해서 가장 널리 사용되고 있는 방법이다. 따라서 상당히 정확한 것으로 알려진 MAC 방법의 결과는 단지 본 논문의 방법과 Stam 방법의 정확도를 비교하기 위해서 사용하고, 본 논문의 방법은 Stam 방법과 비교한다.

정확도 테스트를 위해서 전산유체역학(computational fluid dynamics) 분야의 전통적인 문제인 공동내 흐름(cavity flow) 문제를 시뮬레이션 하였다. 시뮬레이션 환경은 그림 2와 같이 유체로 채워진 사각형 공동과 no-slip 경계 조건으로 이루어진다. 그림 3~6은 MAC 방법, Stam 방법, 본 논문의 방법, 이 세 가지 방법으로 시뮬레이션을 수행한 결과의 최종 정상 상태(final steady state)를 유선(streamline)으로 보여준다. 유선은 유체 역학에서 가장 보편적인 시각적 개념으로써 속도 벡터에 접하는 선을 의미한다. 결과 영상들은 속도의 크기가 결과 영상에 곱해진 LIC(Line Integral Convolution) 기법에 의해 표현되었다[17]. 그림에서 보여지듯이 $\Delta t = 0.005$ 의 작은 시간 간격을 가지고 실행된 결과에서는 본 논문의 방법과 Stam 방법 모두 MAC 방법의 결과와 비교할 때 상당히 정확함을 알 수 있다. $\Delta t = 0.2 \sim \Delta t = 0.7$ 등의 큰 시간 간격을 가지고 실행된

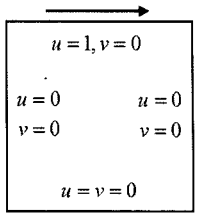
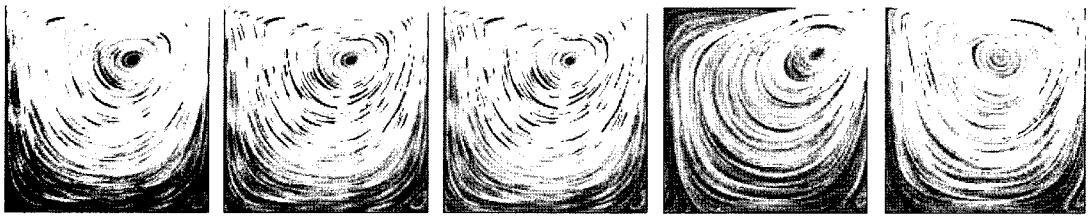


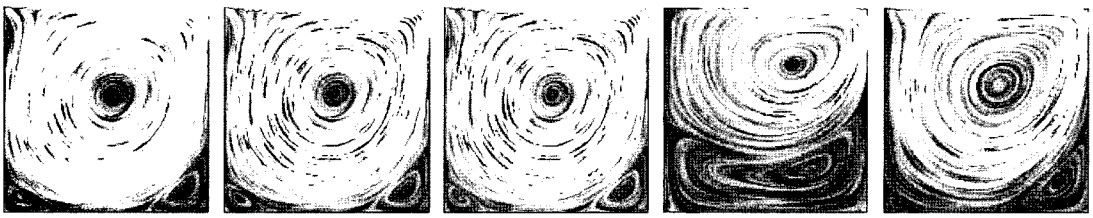
그림 2 공동내 흐름(cavity flow) 시뮬레이션 환경(높이 1, 너비 1)

결과에서는 Stam 방법은 실제 유체의 흐름보다 매우 빨리 속도가 감소하고 때때로 심각한 소실 문제를 발생시킨다. 이러한 현상은 레이놀드 수가 커질수록 더욱 심해짐을 볼 수 있다. 그림 7은 본 논문의 방법과 Stam 방법에 대해서 $\Delta t = 0.1$ 의 시간 간격을 가지고 $Re = 10000$ 인 유체의 시뮬레이션이 시간에 따라 진행되는 과정을 보여준다. Stam 방법에 의한 유체 흐름은 정상 상태의



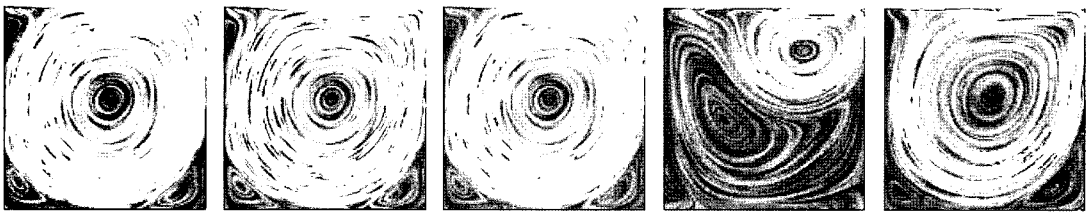
(a) MAC 방법 $\Delta t = 0.001$ (b) Stam 방법 $\Delta t = 0.005$ (c) 논문 방법 $\Delta t = 0.005$ (d) Stam 방법 $\Delta t = 0.7$ (e) 논문 방법 $\Delta t = 0.7$

그림 3 공동내 흐름 시뮬레이션에서 $Re = 100$ 에 대한 최종 정상 상태 유선의 비교



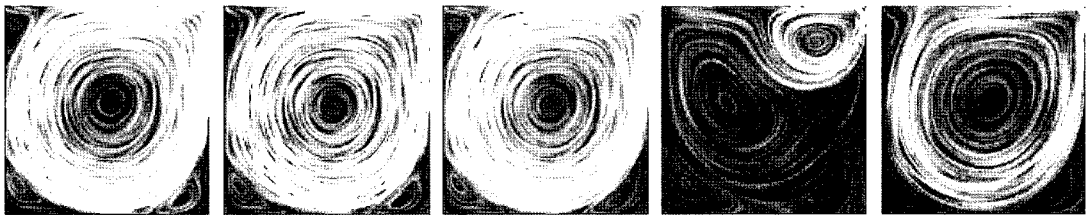
(a) MAC 방법 $\Delta t = 0.001$ (b) Stam 방법 $\Delta t = 0.005$ (c) 논문 방법 $\Delta t = 0.005$ (d) Stam 방법 $\Delta t = 0.3$ (e) 논문 방법 $\Delta t = 0.3$

그림 4 공동내 흐름 시뮬레이션에서 $Re = 1000$ 에 대한 최종 정상 상태 유선의 비교



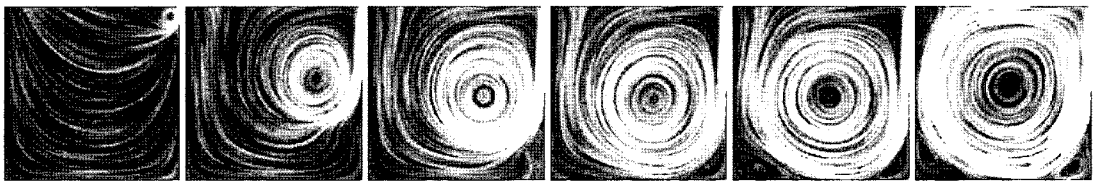
(a) MAC 방법 $\Delta t = 0.001$ (b) Stam 방법 $\Delta t = 0.005$ (c) 논문 방법 $\Delta t = 0.005$ (d) Stam 방법 $\Delta t = 0.2$ (e) 논문 방법 $\Delta t = 0.2$

그림 5 공동내 흐름 시뮬레이션에서 $Re = 5000$ 에 대한 최종 정상 상태 유선의 비교

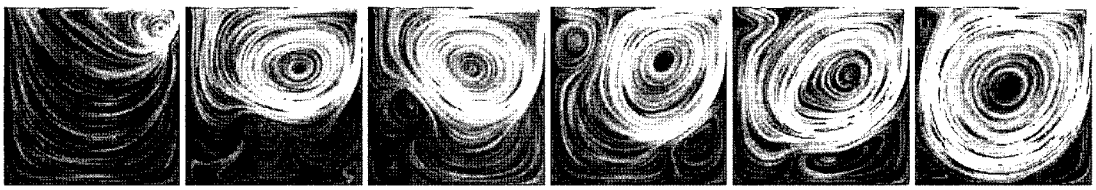


(a) MAC 방법 $\Delta t = 0.001$ (b) Stam 방법 $\Delta t = 0.005$ (c) 논문 방법 $\Delta t = 0.005$ (d) Stam 방법 $\Delta t = 0.2$ (e) 논문 방법 $\Delta t = 0.2$

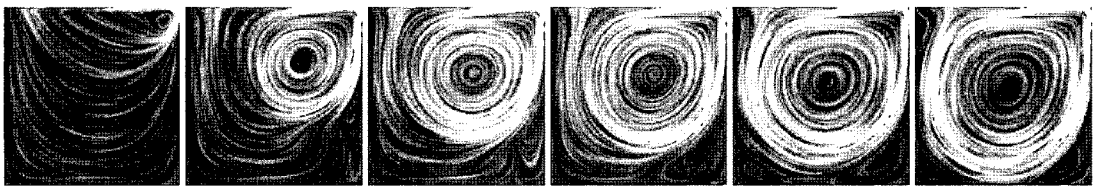
그림 6 공동내 흐름 시뮬레이션에서 $Re = 10000$ 에 대한 최종 정상 상태 유선의 비교



(a) MAC 방법



(b) Stam 방법, $\Delta t = 0.1$



(c) 논문의 방법, $\Delta t = 0.1$

그림 7 $Re = 10000$ 인 유체 시뮬레이션의 시간에 따른 진행 과정

유선이 정확한 모양을 나타낸 경우에도 가끔씩 중간 과정에서 실제 흐름과 다른 모습을 보여준다. 이에 비해 본 논문의 방법은 큰 시간 간격과 큰 레이놀드 수에 대해서 이러한 문제를 발생시키지 않는다.

소실 정도를 테스트하기 위해서 높이 2, 너비 1인 공동내 유체의 흐름을 시뮬레이션 하였다. 그림 8과 같이 공동내의 상단과 하단에서의 속도를 다르게 설정하고 $Re = 1000$ 인 유체가 들어있다고 가정하였다. 그림 9는 각각 다른 시간 간격을 가진 시뮬레이션의 정상 상태의 유선을 속도의 크기에 따라 다른 색깔로 보여준다. Stam 방법에 의한 시뮬레이션은 시간 간격이 커짐에 따라 속도가 빠른 공동의 상단에서 더 많은 소실이 발생한다. 결국은 시간 간격이 큰 경우에 오히려 상단의

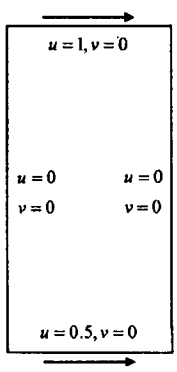


그림 8 높이 2, 너비 1의 공동 : 상단 속도 $\mathbf{v} = (1,0)$, 하단 속도 $\mathbf{v} = (0.5,0)$

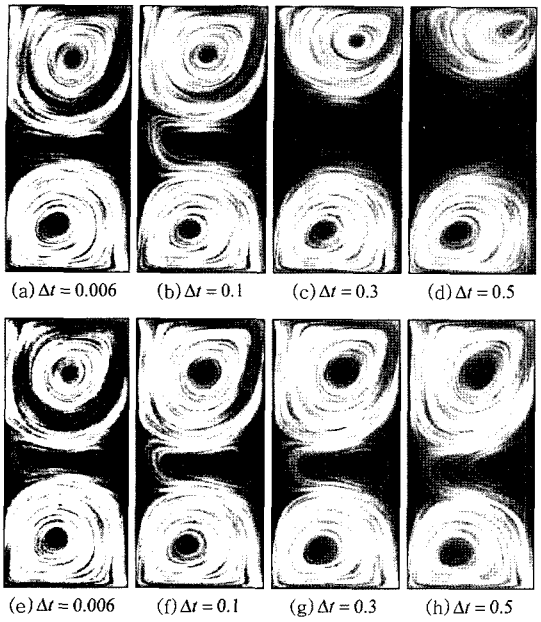


그림 9 다양한 시간 간격에서 $Re = 1000$ 인 유체 시뮬레이션 결과, 속도 크기에 따라 다른 색으로 그려진 정상 상태 유선, (a-d) Stam 방법, (e-h) 논문의 방법

속도가 하단의 속도보다 작아지게 된다. 반면에 본 논문의 방법에 의한 시뮬레이션은 상대적으로 매우 작은 소실을 보여준다.

수치적 오차 분석을 위하여 엄밀해(exact solution)와 본 논문의 방법에 의한 근사해(approximate solution)의 오차를 분석하였다. 계산 영역 $\Omega = (-1,1)^2$ 에서 Navier-Stokes 방정식에 대한 엄밀해는 다음과 같이 설정하였다.

$$\mathbf{v}(x,y,t) = \pi \log(1+t)(\sin 2\pi y \sin^2 \pi x, -\sin 2\pi x \sin^2 \pi y)$$

$$p(x,y,t) = \log(1+t)\sin \pi x \sin \pi y$$

그러면 외부에서 가해지는 힘은 Navier-Stokes 방정식으로부터 식 (17)에 의해 주어지고, 모든 계산은 초기 시간 $t_0 = 0$ 에서 초기 속도 $\mathbf{v} = 0$ 에서 시작한다.

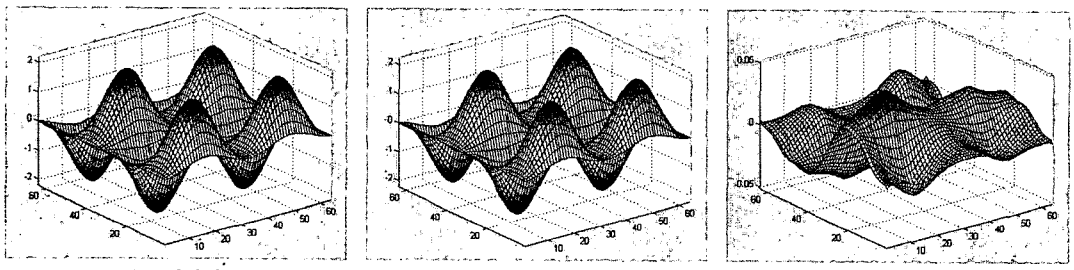
$$\mathbf{f} = \frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} - \frac{1}{Re}\Delta \mathbf{v} + \nabla p \tag{17}$$

속도의 근사해에 대한 에러는 $l^\infty - norm$ 을 사용하여 다음 식 (18)을 측정하고 수렴 비율은 식 (19)로 계산하였다.

$$err_u(t,\Delta t) = \frac{\max_{i,j=0,\ldots,64} \left| \mathbf{v}(x_i,y_j,t) - \mathbf{v}^{\frac{t}{\Delta t}}(x_i,y_j) \right|}{\max_{i,j=0,\ldots,64} \left| \mathbf{v}(x_i,y_j,t) \right|} \tag{18}$$

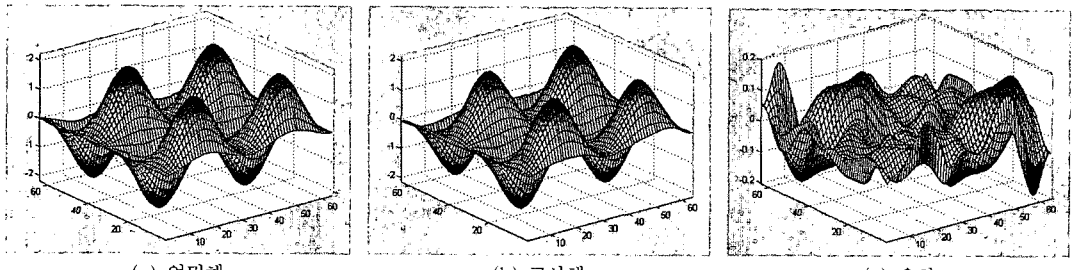
$$rate_u = \frac{err_u(t,\Delta t)}{err_u(t,\Delta t/2)} \tag{19}$$

표 1은 $t = 1$ 에서의 속도 u 성분의 오차 및 수렴 비율을 보여주고 그림 10과 그림 11은 $t = 1$ 에서 엄밀해, 근



(a) 엄밀해 (b) 근사해 (c) 오차

그림 10 $t = 1$ 에서 엄밀해와 근사해의 오차 ($\mathrm{Re} = 1$)



(a) 엄밀해 (b) 근사해 (c) 오차

그림 11 $t = 1$ 에서 엄밀해와 근사해의 오차 ($\mathrm{Re} = 5000$)

표 1 속도 u 성분의 오차 및 수렴 비율

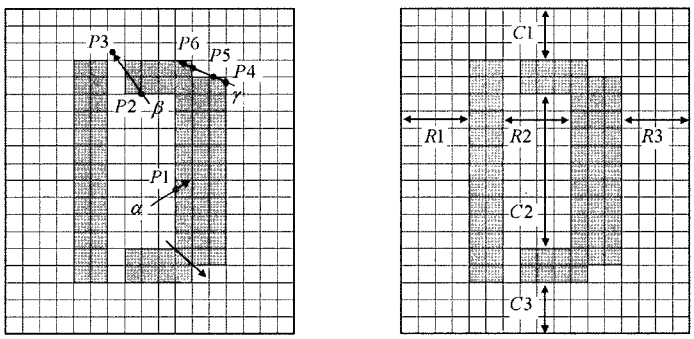
Δt	Re = 1		Re = 100		Re = 5000	
	err_u	$rate_u$	err_u	$rate_u$	err_u	$rate_u$
0.25	0.130	2.05	0.194	2.09	0.438	2.11
0.125	0.063	2.04	0.093	1.92	0.207	1.95
0.0625	0.031		0.048		0.106	

사해 그리고 두 해의 오차를 보여준다. 본 논문의 방법은 유한 차분법을 사용하므로 공간 이산화로 발생하는 오차를 포함하고 있지만 시간에 의해서 발생하는 오차에 비하면 작기 때문에 큰 영향을 주지 않는다.

계산 영역 내부에 경계가 있는 경우에 식 (16)의 행렬을 경계가 나타날 때마다 작은 행렬들로 나눈다. 여기서 경계라 함은 유체와 유체가 아닌 물체가 접하는 영역을 의미한다. 그림 12는 그림 14의 내부 경계들 중에서 일부만을 포함하는 간단한 계산 영역을 나타낸다. 이 계산 영역에 대해서 식 (16)의 행렬은 R1, R2, R3가 있는 행에 대해서 세 개의 작은 행렬들로 나누어 계산한다. 이처럼 본 논문의 방법은 계산 영역 내부에 복잡한 경계가 있는 시뮬레이션에 적용이 용이하다. Stam 방법의 경우 Semi-Lagrangian 기법으로 알려진 대류 단계를 포함한다. 이 기법은 시간에 대해 역방향으로 추적하여 속도의 이전 위치를 찾고, 그 위치의 속도를 현재 속도로 설정하는 기법이다. 그림 12에서 α 의 경우 P1 위치의 속도를 새로운 속도로서 설정하면 되지만 β, γ 의 경우는 P2, P3, P4, P5, P6중에서 어떤 위치를 선택해야 할지 기준이 모호하다.

세 방법 모두 압력에 대한 Poisson 방정식 풀이는 동일하기 때문에 무발산계로 사영되지 않은 속도 계산에 대해서만 계산 성능을 측정하였다. MAC 방법이 암시적 방법이기 때문에 가장 빠르다. Stam 방법은 확산 단계라는 선형시스템 풀이 단계를 포함하고 있어서 시간 간격, 레이놀드 수, 외부에서 가해지는 힘과 같은 조건들에 따라 계산 시간이 유동적이어서 계산 시간을 정확하게 측정하기가 어렵다. 본 논문의 방법의 계산 시간은 Stam 방법이 확산 단계의 반복법에서 10~15회 반복하는 경우와 비슷하다(그림 13 참조). Stam 방법은 고정된 반복 횟수가 예기치 않은 나쁜 결과를 야기할 수 있기 때문에 본 논문에서 Stam 방법에 의한 시뮬레이션은 수렴 조건이 만족될 때까지 반복 계산이 수행되었으며 평균적으로 20회 이상의 반복을 필요로 하였다.

시각적 효과를 높이기 위하여 유체의 속도장에 속도에 의해 움직이는 유체 입자와 연기를 삽입하였다. 그림 14는 복잡한 내부 경계들을 가진 유체 시뮬레이션을 보



(a) Stam 방법 (b) 논문의 방법

그림 12 그림 14의 내부 경계중 하나의 경계를 가진 계산 영역

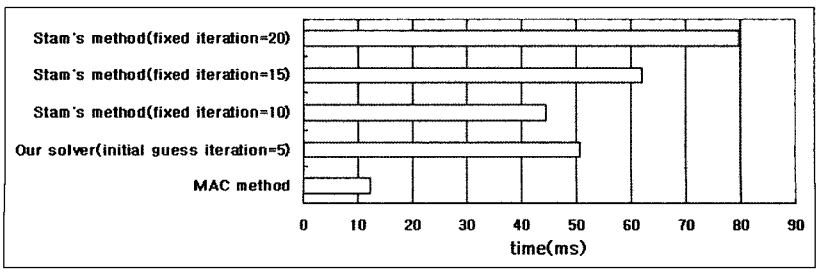
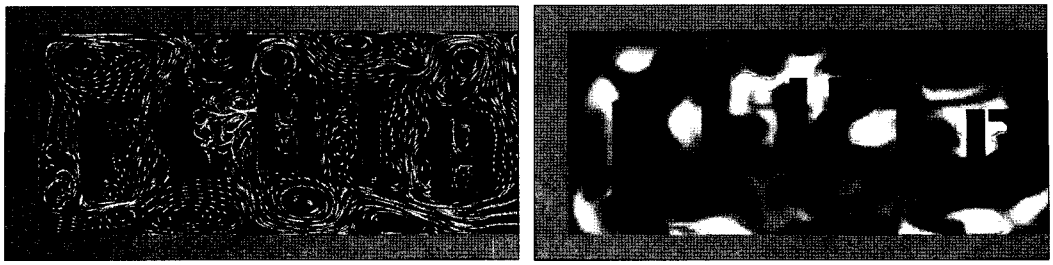


그림 13 CPU 계산 시간 비교, 격자 수 : 256×256



(a) 유체 입자의 움직임 (b) 연기의 움직임

그림 14 복잡한 내부 경계가 있는 계산 영역에서의 유체 입자와 연기의 시뮬레이션

여준다. 계산 영역 내에 균일하게 분포된 유체 입자들은 자신을 둘러싼 속도에 의해서 움직이며 이전 시간 간격 10단계의 자취를 가진다. 연기의 움직임을 시뮬레이션하기 위해서 다음과 같은 이류-확산 방정식을 도입하였다.

$$\frac{\partial d}{\partial t} = -(\mathbf{v}\cdot\nabla)d + k\Delta d + d_s$$

여기서 d는 연기의 밀도, k는 확산 계수, d_s는 연기의 유입량을 나타낸다. 속도 계산에 사용된 ADI 방법을 적용하면 수식은 다음과 같다.

$$\frac{2}{\Delta t}\left(\tilde{d} - d^n\right) = -u\Delta_x^{dc}\tilde{d} - v\Delta_y^{dc}d^n + k(\Delta_{xx}\tilde{d} + \Delta_{yy}d^n) + d_s$$

$$\frac{2}{\Delta t}\left(d^{n+1} - \tilde{d}\right) = -u\Delta_x^{dc}\tilde{d} - v\Delta_y^{dc}d^{n+1} + k(\Delta_{xx}\tilde{d} + \Delta_{yy}d^{n+1}) + d_s$$

여기서 Δ_x^{dc}, Δ_y^{dc}는 다음과 같이 정의되는 donor-cell 미분 연산자이다.

$$\Delta_x^{dc} = \frac{\partial(ud)}{\partial x} = \frac{u_i d_r - u_{i-1} d_l}{\Delta x},$$

$$d_l = \begin{cases} d_{i-1} & u_{i-1} > 0 \\ d_i & u_{i-1} < 0 \end{cases}, \quad d_r = \begin{cases} d_i & u_i > 0 \\ d_{i+1} & u_i < 0 \end{cases}$$

그림 15는 다른 속도로 두 번 내뿜어지는 연기의 3차원 시뮬레이션 결과 영상이다. 이 시뮬레이션의 기본 아이디어는 "The Virtual Album of Fluid Motion"에서 발췌되었다[18]. 연기는 그래픽 하드웨어 가속기를 사용한 볼륨 렌더링(volume rendering)[19] 기법으로 256MB의 비디오 메모리를 가진 ATI Radeon 9800 그래픽 카드를 가지고 OpenGL API와 OpenGL Extensions을 사용하여 가시화하였다.

5. 결론 및 향후 연구 과제

본 논문은 실시간 물리 기반 유체 시뮬레이션을 위해서 Navier-Stokes 방정식에 대한 빠르고 정확한 풀이 기법을 제안하였다. 본 논문의 방법은 Poisson 방정식 계산과 시간 복잡도 $O(N)$을 가지는 ADI 방법에 의한 3중 대각 행렬의 풀이로 구성된다. 따라서 계산이 빠른 것으로 알려진 Stam 방법보다 일반적으로 계산이 빠르며 계산 시간의 변동 또한 적다.

시뮬레이션 결과에서 보여 지듯이 유체 시뮬레이션 방법들은 작은 시간 간격에 대해서는 상당히 정확하였

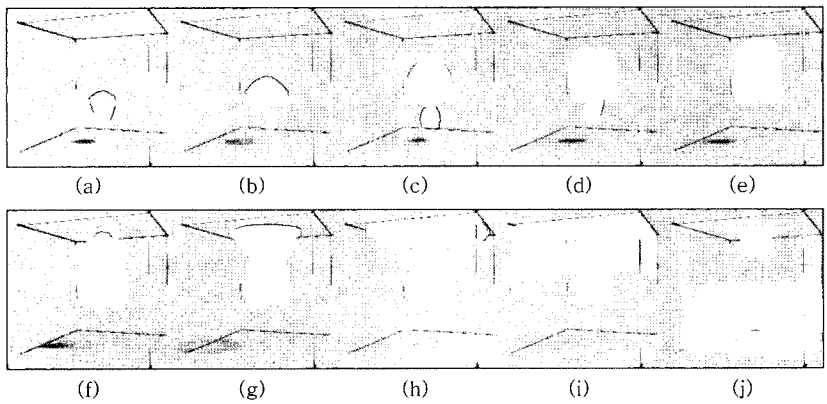


그림 15 다른 속도로 두 번 내뿜어지는 연기의 3차원 시뮬레이션, 그림(a)~그림(j)는 시간의 진행에 따른 변화

다. 그러나 Stam 방법의 경우 시간 간격이 증가함에 따라서 정확성을 잃게 되고 때때로 예기치 않은 유체의 흐름을 야기 하였다. 본 논문에서 제안한 방법은 큰 시간 간격과 큰 레이놀드 수에 대해서 상당히 정확한 결과를 나타내었다. 이것은 Stokes 방정식의 해를 이용해 속도의 초기값을 실제 해와 매우 근사하게 추정함으로써 얻어진 결과이다.

본 논문의 방법은 큰 시간 간격, 큰 레이놀드 수, 강한 외부 힘에 대해서 시스템이 매우 안정적이다. 그러나 어떠한 시간 간격에서도 시스템의 안정성을 보장하는 이론적 근거에도 불구하고 실제 시뮬레이션에서는 시간 간격에 제한이 있었다. 안정성 기준이나 시뮬레이션 변수들과의 상관관계를 찾는 것은 향후 연구과제일 것이다.

최근에 관심이 모아지고 있는 GPU(Graphics Processing Units)를 이용한 범용 계산에 관한 연구들에서 GPU에서의 행렬 계산과 편미분 방정식의 풀이로 이루어진 물리 기반 시뮬레이션 구현이 상당한 성능 향상을 보여주었다[20-23]. 본 논문의 방법은 행렬 계산만으로 이루어져 있고 더욱이 ADI 방법은 병렬처리가 가능한 행렬들의 계산으로 이루어져 있다. 그러므로 GPU를 이용하면 상당한 속도 향상을 기대할 수 있을 것으로 예상한다.

참 고 문 헌

[1] Harlow, F.H. and Welch, J.E., "Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with a Free Surface," The Physics of Fluids, 8, pp. 2182-2189, 1965.

[2] Kass, M. and Miller, G., "Rapid, stable fluid dynamics for computer graphics," ACM Computer Graphics, 24, pp. 49-57, 1990.

[3] Chen, J. and Lobo, N., "Toward Interactive-Rate Simulation of Fluids with Moving Obstacles Using the Navier-Stokes Equations," Graphical Models and Image Processing, 57, pp. 107-116, 1994.

[4] Foster, N. and Metaxas, D., "Realistic Animation of Liquids," Graphical Models and Image Processing, 58, pp. 471-483, 1996.

[5] Stam, J., "Stable Fluids," ACM SIGGRAPH 99, pp. 121-128, 1999.

[6] Foster, N. and Fedkiw, R., "Practical Animation of Liquids," ACM SIGGRAPH 01, pp. 23-30, 2001.

[7] Treuille, A., McNamara, A., Popovic, Z, and Stam, J., "Keyframe Control of Smoke Simulations," ACM SIGGRAPH 03, pp. 716-723, 2003.

[8] Fedkiw, R., Stam, J., and Jensen, H., "Visual Simulation of Smoke," ACM SIGGRAPH 01, pp. 15-22, 2001.

[9] Streeter, V. L., Wylie, E. B., and Bedford, K. W., Fluid Mechanics, McGraw-Hill, 1998.

[10] Peaceman, D. W., and H. H. Rachford, J., "Numerical solution of parabolic and elliptic differential equations," J. Soc. Indust. Appl. Math., 3, pp. 28-41, 1955.

[11] Girault, V. and Raviart, P. A., Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms, Springer-Verlag, New York, 1986.

[12] Chorin, A. J., "On the convergence of discrete approximations to the Navier-Stokes equations," Math. Comput., 22, pp. 745-762, 1968.

[13] Temam, R., "Une methode d'approximation de la solution des equations de Navier-Stokes," Bull. Soc. Math. de France, 98, pp. 115-152, 1968.

[14] Shen, J., "On Error Estimates of the Projection Methods for the Navier-Stokes Equations: Second-Order Schemes," Math. Comput., 65, pp. 1039-1065, 1996.

[15] Peyret, R. and Taylor, T. D., Computational Methods for Fluid Flow, Springer-Verlag, New York, 1983.

[16] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., Numerical recipes in C. The Art of scientific Computing. Cambridge University Press, 1992.

[17] Brian, C., and Leith, L. C., "Imaging vector fields using line integral convolution," SIGGRAPH 93, ACM Press, pp. 263-270, 1993.

[18] Kilian, S.,and Turek, S. The Virtual Album of Fluid Motion: An Interactive Exploration via Numerical Simulation, DVD, Springer, 2002.

[19] Rudiger, W., and Thomas, E. "Efficiently using graphics hardware in volume rendering applications," SIGGRAPH 98, ACM Press, pp. 169-177, 1998.

[20] Bolz, J., Farmer, I., Grinspun, E., and Schroder, P., "Sparse matrix solvers on the GPU: conjugate gradients and multigrid," ACM Trans. Graph. 22, 3, pp. 917-924, 2003.

[21] Krueger, J., and Westermann, R., "Linear algebra operators for GPU implementation of numerical algorithms," ACM Trans. Graph. 22, 3, pp. 908-916, 2003.

[22] Harris, M. J., "Real-Time Cloud Simulation and Rendering," PhD thesis, University of North Carolina, 2003.

[23] Zeller, C., "Cloth Simulation on the GPU," SIGGRAPH 2005 Conference Abstracta and Applications, 2005.

부 록

엄격히 대각 성분 우세(strictly diagonal dominance) 증명 $\hat{u}_{i,J} > 0$ 이고 $\varepsilon_{\hat{u}} = 1$ 인 경우에 다음과 같다.

$$a_{i,J} = \frac{-\hat{u}_{i,J}}{h} - \frac{1}{Reh^2}, \quad b_{i,J} = \frac{2}{\Delta t} + \frac{\hat{u}_{i,J}}{h} + \frac{2}{Reh^2},$$

$$c_{i,J} = -\frac{1}{Reh^2}$$

$$|a_{i,J}| = \frac{\hat{u}_{i,J}}{h} + \frac{1}{Reh^2}, \quad |b_{i,J}| = \frac{2}{\Delta t} + \frac{\hat{u}_{i,J}}{h} + \frac{2}{Reh^2},$$

$$|c_{i,J}| = \frac{1}{Reh^2}$$

변수들이 모두 양수이고 $\hat{u}_{i,J} > 0$ 이기 때문에 다음이 성립한다.

$$|b_{i,J}| = \frac{2}{\Delta t} + \frac{\hat{u}_{i,J}}{h} + \frac{2}{Reh^2} \geq \frac{\hat{u}_{i,J}}{h} + \frac{1}{Reh^2} + \frac{1}{Reh^2} = |a_{i,J}| + |c_{i,J}|$$

유사하게 $\hat{u}_{i,J} < 0$ 이고 $\varepsilon_{\hat{u}} = -1$ 인 경우는 다음과 같다.

$$a_{i,J} = -\frac{1}{Reh^2}, \quad b_{i,J} = \frac{2}{\Delta t} - \frac{\hat{u}_{i,J}}{h} + \frac{2}{Reh^2},$$

$$c_{i,J} = \frac{\hat{u}_{i,J}}{h} + \frac{1}{Reh^2}$$

$$|a_{i,J}| = \frac{1}{Reh^2}, \quad |b_{i,J}| = \frac{2}{\Delta t} - \frac{\hat{u}_{i,J}}{h} + \frac{2}{Reh^2},$$

$$|c_{i,J}| = -\frac{\hat{u}_{i,J}}{h} + \frac{1}{Reh^2}$$

변수들이 모두 양수이고 $\hat{u}_{i,J} < 0$ 이기 때문에 다음이 성립한다.

$$|b_{i,J}| = \frac{2}{\Delta t} - \frac{\hat{u}_{i,J}}{h} + \frac{2}{Reh^2} \geq \frac{1}{Reh^2} - \frac{\hat{u}_{i,J}}{h} + \frac{1}{Reh^2} = |a_{i,J}| + |c_{i,J}|$$

김 영 희

1995년 부산대학교 자연과학대학 수학과(학사). 2001년 경북대학교 컴퓨터과학과(석사). 2001년 3월~현재 경북대학교 컴퓨터과학과 박사과정. 관심분야는 컴퓨터 그래픽스 및 애니메이션, 물리 기반 시뮬레이션

이 성 기

1979년 서울대학교 공과대학 전기공학과(학사). 1981년 서울대학교 대학원 전기공학과(석사). 1990년 University of Utah, Department of Computer Science(박사). 1982년~1984년 울산대학교 컴퓨터공학과 교수. 1990년~현재 경북대학교 컴퓨터과학과 교수. 관심분야는 영상처리, 컴퓨터그래픽스, 의료정보