모바일 기기를 위한 실시간 유체 시뮬레이션 엔진

Interactive Fluid Simulation Method for Mobile Device

김도엽, Doyub Kim*, 송오영, Oh-young Song**, 고형석, Hyeong-Seok Ko***

요약 본 논문은 데스크탑 PC 에서만 구현 가능하였던 기존의 유체 시뮬레이션 기술을 모바일 환경으로 확장하는 방법론을 제시한다. 유체 시뮬레이션은 나비어-스토크스 (Navier-Stokes) 방정식의 수치적 해를 구하는 것이며, 기존의 방법론은 수치적 해의 안정성과 [1] 사실성 [2]에 그 초점을 맞추고 있다. 하지만 이는 모바일 기기에서 기대하기 힘든 충분한 연산 자원을 가정한 것이다. 한편, 모바일 환경에서의 물리기반 기술은 현재 강체 시뮬레이션 모듈이 주로 활용되고 있으며 [3], 유체 시뮬레이션은 높이장 (Height field) 기반의 단순한 모델만이 제시되어있다 [4]. 이를 극복하기 위해 본 연구에서는 이러한 한계를 극복한 수정된 비압축유동의 시뮬레이션 기법을 소개하며, 또한 모바일 상에서 유체의 가시화 기술을 제안한다.

Abstract This paper proposes a method for extending simulating fluid on mobile device, which was only possible on desktop PC. Fluid simulation is done by solving Navier-Stokes equation numerically, and previous research were mainly focused on numerical stability [1], and realism [2]. However, such methods assume rich computational resources, which is not available on mobile devices. On the other hand, rigid-body solver is the mostly used physically-based technique [3], and only simple height field-based method is released for fluid simulation [4]. To overcome these problems, we proposes a modified incompressible fluid dynamics solver for the mobile device, and also we propose a technique for visualizing fluids on the mobile device.

핵심어: Fluid simulation, fluid animation, computational fluid dynamics, mobile device

본 논문은 2008 년 BK21 학술 연구비 지원에 의하여 연구되었음.

*주저자: 서울대학교 전기공학부 박사과정 e-mail:kim@ graphics.snu.ac.kr

**공동저자 : 세종대학교 교수 e-mail: oysong@sejong.ac.kr

***교신저자 : 서울대학교 전기공학부 교수; e-mail: ko@graphics.snu.ac.kr

1. 서론

유체 현상은 연기, 불, 액체 등의 다양한 현상들을 포함하며, 이러한 현상들을 영화나 광고, 게임 등에서 재현하기 위한 연구들이 현재까지 활발히 진행되어왔다. 하지만 유체 현상을 표현하는 나비어-스토크스 (Navier-Stokes) 방정식은 일반해가 존재하지 않으며, 따라서 근사된 수치 방정식만을 통하여 시뮬레이션이 가능하다.

유체 방정식을 안정화된 수치 방정식으로 근사하는 방법은 [1]에서 소개된 바 있다. 또한, 이러한 근사과정에서 발생하는 수치적 소산과 같은 오차들을 해결하기 위한 방법 또한 [6]에서 소개되었다. 이와 유사한 방법들은 [1]가 발표된 이후, 비교적 활발히 연구, 개발되었으며 다양한 영화, 혹은 게임에 적용된 바 있다.

하지만, 위의 방법들은 충분한 메모리와 충분한 계산 시간을 가정하고 있다. 최근 Graphics Processing Unit (GPU)을 활용한 방법론이 제시된 바 있으나, 이 역시 고사양의 그래픽카드를 염두 한 기술이다. 따라서 상대적으로 저사양 중앙처리장치(CPU)를 탑재한 모바일 기기에서는 기존의 방법론을 그대로 적용하기에 무리가 있다. 또한 모바일 기기들은 저성능의 GPU 를 내장하고 있으므로, 유체 시뮬레이션 결과의 가시화 또한 문제점이다.

본 논문에서는 우선 계산량의 문제점을 개선하기 위해, 비압축유동의 해석에 있어서 가장 병목에 해당하는 압력 포아슨 방정식(Pressure Poisson equation, PPE)의 계산을 예측-수정 방법을 통해 개선하였다. 이는 순환법 기반의 선형 시스템 해석기를 통하여 달성되었으며, 대류항으로 예측된 압력으로 보다 빠른 시간에 해를 구해낼 수 있게 되었다. 또한 사용자가 터치 등의 입력으로 유체를 조종할 수 있도록 연기 밀도의 대류항을 추가로 구현하였다. 한편, 이를 가시화하기 위한 방법으로 OpenGL|ES [5] 텍스쳐 기반의 기법을 제시한다.

2. 관련 연구

Foster 와 Metaxas [6, 7]가 3 차원 나비어-스토크스 방정식을 해석하는 유체 애니메이션 방법을 소개한 이래로, 유체 애니메이션은 컴퓨터 그래픽스 분야에 널리 연구되어 왔다. 1999 년에 Stam [1]은 Stable Fluids 로 알려진 안정적인 유체 시뮬레이터를 소개했다. 이 시뮬레이터에서는 큰 시뮬레이션 시간 간격을 사용해도 안정적으로 동작할 수 있도록 대류항을 세미-라그랑지안(semi-Lagrangian)방법을 통해 해석한다. 그 이후, 이 방법을 기초로 빠르게 동작하는 유체 시뮬레이터를 개발하기 위해 활발히 연구가 진행되어 왔다.

액체를 시뮬레이션하기 위해서는 안정적인 시뮬레이터 뿐만 아니라, 액체의 표면을 추적하는 모델도 필요하다. 이를 위해 Foster 와 Fedkiw [8]는 레벨 셋(level set)법을 제시하였으며 이는 Enright 등 [9]에 의해 입자 레벨 셋(particle level set)법으로 발전하게 된다.

또한 컴퓨터 시뮬레이션에서 시각적 사실성 및 물리적 정확성을 떨어뜨리는 가장 큰 요인인 수치적 소산을 해결하기 위해, Fedkiw 등 [10]은 3 차함수를 이용한 보간법을 소개하였다. 또한 [10]에서는 와류 강조법(vorticity confinement)를 도입하여 유속의 회전 성분을 인위적으로 증폭하였다. 또한 와류의 소산을 보다 잘 막기 위해 와류 입자법(vortex particle method)등이 도입되었다 [11]. 또한 최근에는 고차의 안정화된 보간법을 사용한 대류 해석 기법이 발표된 바 있다 [2, 12].

효과적인 유체의 시뮬레이션을 위해 적응적 격자 기반의 방법들이 [13-15]에서 제시된 바 있다. Losasso 등[13]은 옥트리(octree)기반의 적응적 격자를 사용하여 연기 및 액체의 시뮬레이션을 보여준 바 있다. 유사한 방법으로 semi-Lagrangian contouring 기법이 [14]에 소개되었다. Houston 등[15]은 계층적 RLE 격자를 사용하여 보다 적은 메모리로 고해상도의 시뮬레이션을 재현한 바 있다. 연산 시간을 높이기 위해 CPU 가 아닌 GPU 를 활용한 기법 또한 소개된 바 있다[17].

3. 유체의 수치적 해석

비압축유동 해석을 위한 지배 방정식인 나비어-스토크스 방정식은 다음과 같다.

$$u_t = -u \cdot \nabla u - \nabla p + f \quad (1)$$

$$\nabla \cdot u = 0 \quad (2)$$

식 (1)은 운동량 보존식이며, 식 (2)는 질량 보존식이다. 이때 u 는 유속을 의미하며, p 는 압력, f 는 중력과 같은 외력을 의미한다. (1)번 식 우변의 첫 번째 항은 대류로 인한 관성력을 의미하며, 두 번째 항은 압력 경도로 인한 유체 가속을 의미한다. 위 식은 나비어-스토크스 방정식에서 점성항이 제외된 식으로서, 오일러 (Euler) 방정식이라고도 일컫는다. 즉, 위 식은 정확히 말하자면 비점성비압축유동의 지배 방정식이라고 할 수 있다. 점성항을 제외한 이유는 비선형 항인 대류항(advection)을 계산할 때 발생하는 수치적 소산효과가 바로 점성항의 역할을 해주기 때문이다. 매우 점도가 높은 유체를 재현하기 위해서는 어렵지 않게 점성항을 추가할 수 있다 [16].

본 논문에서는 유체 현상의 하나로 연기 시뮬레이션을 가정하였다. 연기의 밀도 ρ 는 전 계산 영역에 분포해 있으며 이는 부력의 계산과 연기의 가시화에 사용된다. 연기의 밀도를 통한 부력의 계산은 다음과 같이 한다.

$$f_{buoy} = -\alpha \ (\rho - \rho_{avg})\mathbf{z} \qquad (3)$$

위 식에서 α 는 사용자가 조절하는 파라미터이며, ρ_{avg} 는 전 영역에 대한 밀도의 평균값이다. 즉, 위 식은 전체의 밀도값에 비해 어떤 위치의 밀도가 낮다면, 해당 위치에서는 상위 방향으로 힘을 받는 것을 의미한다. 위 식은 [10]의 부력 식을 단순화 한 것으로, 보다 빠른 연산을 위해 도입되었다. 기존의 식은 밀도와 더불어 연기의 온도에 대한 고려가 포함되어 있지만, 위 식은 밀도만으로 부력을 표현하였다. 이렇게 계산된 부력은 외력의 형태로 식 (1)에 가한다.

앞서 언급한 식들을 수치적으로 해석하기 위해서 본 논문에서는 격자 기반의 수치해석법을 사용하였다. 식 (1)의 대류항을 수치적으로 해석하기 위해서는 [1]에서 소개된 세미-라그랑지안 기법이 사용되었다. 이 기법은 식 (1)을 시간에 대해 적분할 때 기존의 유한차분법이 가지는 조건적 안정성을 해결한 방법으로, 이론적으로는 시간격과 상관없이 무조건적 안정성을 보장한다 (물론 수치적 오차는 시간격이 커질수록 늘어난다).

식 (1)의 압력항과 식 (2)의 질량 보존 조건은 밀접하게 연관되어있다. 유체가 질량 보존을 어기는 방향으로, 즉 특정 위치에 뭉치거나 흩어지는 방향으로 움직일 경우, 이를 즉시 평형 상태로 되돌리는 힘이 압력에 의한 힘이다. 따라서 압력에 의한 평형을 가정하여 식 (1)과 (2)를 결합하면 다음과 같은 식으로 전개된다.

$$\nabla \cdot \nabla p = \nabla \cdot u / dt \qquad (4)$$

위 식은 포아슨 방정식이며, 이는 Ax=b 형태의 선형 시스템으로 재정리 할 수 있다. 위 선형 시스템의 해인 p를 구하면

$$u = u^{*} - \nabla p \qquad (5)$$

위와 같이 속도장에 압력의 경도를 가해줄 수 있다.

선형 시스템의 해를 구하는 방법은 매우 다양하나 일반적으로 데스크탑 기반의 유체 엔진에서는 Preconditioned Conjugate Gradient (PCG)법을 주로 사용한다. PCG 법은 정확한 해로 수렴하는 시간이 다른 선형 시스템 해석 기법들에 비해 짧다는 장점이 있다. 구체적인 알고리즘은 Saad [18]에 기술되어있다. 하지만 절대적인 계산량은 대류항이나 외력항을 해석하는데 비해 매우 높으며, 따라서 PCG 법은 실시간 시뮬레이션에는 적합하지 않다. 또한, PCG 법은 rougher 계열의 시스템 해석기로서, 연산 도중 수렴하지 않은 상태에서는 현재 구한 임시 해 x 가 실제 해와 시각적으로 유사하다고 보기

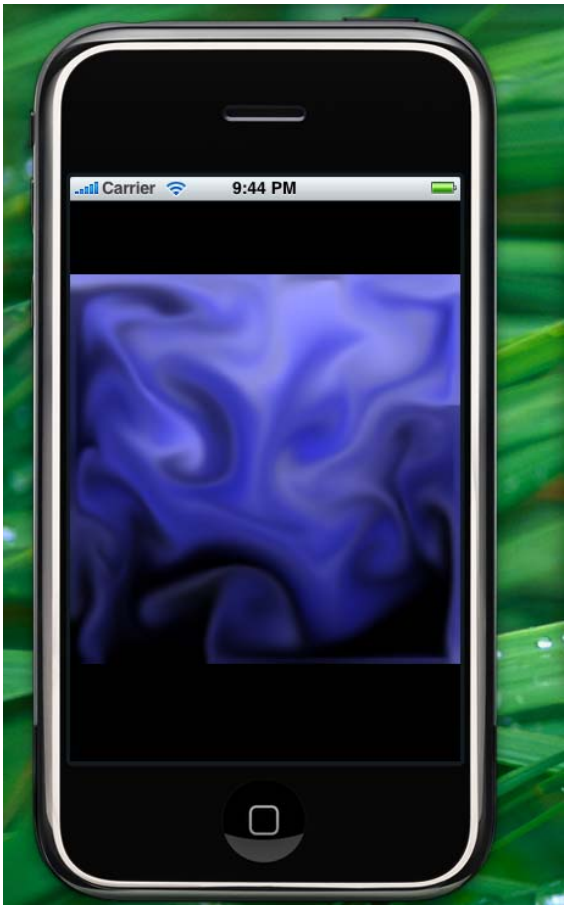

그림 1 – 유체 방정식의 해석을 통한 연기 시뮬레이션 모습. 연기의 밀도와 유속은 본문에서 설명한 semi-Lagrangian 기법을 통해 해석되고 있다. 또한 질량 보존식이 만족되어 소용돌이와 같은 현상들이 재현되고 있다.

그림 2 – 연기 시뮬레이션의 연속 이미지. 초기 이미지를 텍스쳐 형태로 받으면 이를 연기 밀도의 초기조건으로 사용하여 (왼쪽) 시뮬레이션을 전개해 나간다.

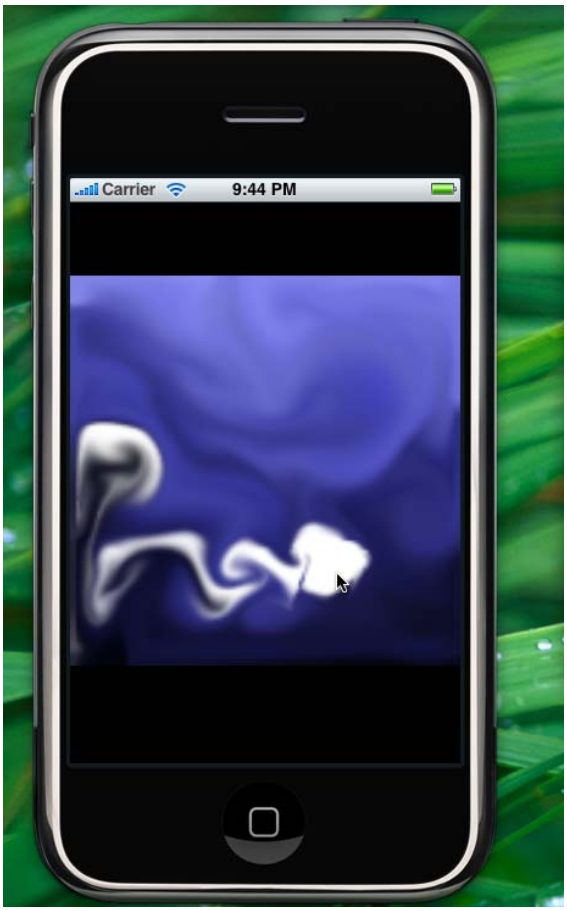

그림 3 – 사용자에 반응하여 연기가 추가되는 모습. 배경의 유속에 의해 밀도가 높아지는 순간 주변으로 흩어짐을 볼 수 있다.

힙들다 [18]. 즉, 적당한 수렴치에서 연산을 멈추는 것이 힘들고, 따라서 단위 시간 당 프레임 비율(FPS)이 중요한 실시간 엔진에 활용하기에는 무리가 있다. 특정 시간격에서는 계산이 빨리 종료되어 FPS 가 유지가 되지만, 수렴이 더딘 상황에서는 목표 FPS 를 초과할 가능성이 높다.

이를 극복하기 위하여 [17]에서는 순환(iterative)법 중 하나인 자코비(Jacobi)법을 사용하였다. 자코비법은 PCG 법에 비해 수렴 속도 자체는 느리나, 연산이 간단하고 병렬화가 용이하며, 연기와 같이 질량보존의 오차가 크게 눈에 띄지 않는 경우에는 순환의 횟수를 제한하여 속도를 크게 향상시킬 수 있다 [18]. 또한 자코비법은 smoother 계열의 순환법으로서, 연산을 도중에 멈추더라도 시각적으로 실제 구하고자 하는 해와 유사하게 보인다 (물론 이는 수치적 오차와는 거리가 있다). 하지만 자코비법은 순환법 중 같은 순환 횟수에 따른 수치적 오차가 가장 큰 방법으로, 속도는 향상될 수 있으나 순환 횟수를 제한하면 수치적 오차가 눈에 잘 띄게 된다.

이를 해결하기 위해 본 논문에서는 이전 시간격에서 해로 구한 압력값 p 를 대류방정식을 통하여 다음 시간격에서의 p 로 예측하고, 이를 시점으로 포아슨 방정식을 해석하는 방법을 택하였다. 이는 시간에 따른 압력의 변화가 부드러울 것이라는 가정을 하고 있으며, 압력 p 에 대해 대류항을 해석하는 과정이 필요하다. 예측된 압력 p 를 사용하여 현재 시간격에서의 압력을 계산하기 위해, 본 논문에서는 가우스-사이델(Gauss-Seidel) 방법을 사용하였다 [18]. 이는 자코비법에 비해 병렬화는 용이하지 않으나, 해로 수렵하는 속도가 더 빠르며, 따라서 같은 횟수의 순환을 거친 뒤에는 자코비법에 비해 수치적 오차가 훨씬 적다 (동일한 오차범위에 수렴하기 위해 자코비법은 가우스-사이델법에 비해, 3 배 이상의 순한이 필요하다). 또한 가우스-사이델법은 자코비법과 비교하여, 한번의 순환에 요구되는 연상량이 거의 동일하므로, 속도의 저하를 가지고 오지 않는다. 가우스-사이델법 대신 PCG 법을 사용할 수도 있으나, PCG 법은 rougher 계열의 순환법으로서 smoother 계열의 가우스-사이델법과 같이 이전 시간격의 압력으로 현재를 예측하는 것이 보다 빠른 수렴성을 보장하지 않는다. 본 논문에서 구현한 예측 기반의 가우스-사이델법은 자코비법에 비해 동일한 수렴 오차에서 4 배 가량의 속도 향상을 얻을 수 있었으며, 일반적인 가우스-사이델법에 비해 20%가량의 연산량을 줄일 수 있었다 (순환 횟수 기준). 본 방법을 적용한 예는 그림 1 에 나타나있다.

4. 사용자 반응 및 유체 가시화

사용자의 입력에 반응하는 유체 엔진을 위해 본 논문에서는 손가락 입력을 통해 화면을 건드리면 연기를 추가로 불어넣을 수 있는 방식을 가정하고 Apple iPhone 의 멀티 터치 센서를 활용하여 이를 구현하였다.

이를 위해 유체의 유속이 아닌 연기 매질을 표현하는 방정식을 추가로 기술하면 다음과 같다.

$$\rho_t = -u \cdot \nabla \rho + S_{touch} \quad (6)$$

위 식은 연기 매질의 대류에 의한 힘과 사용자 입력으로 인한 매질의 추가를 의미한다. 여기서 ρ 는 연기의 밀도를 의미한다. 일반적으로 ρ 는 스칼라(Scalar)값을 취하게 되나, 본 논문에서는 RGB 의 3 차원 벡터 값을 취하도록 하였다.

우변 첫 번째 항은 식 (1)과 마찬가지로 대류를 의미하는데, 이때 사용되는 속도 u 는 식 (1)과 (2)를 연산한 결과이다. 속도 u 를 연산할 때와 마찬가지로, 이때 사용되는 수치해석 기법은 세미-라그랑지안이다. 식 (6)에는 식 (1)과 마찬가지로 매질의 확산과 관련된 항이 존재하지 않는데, 이는 앞에서 밝힌 바와 같이 세미-라그랑지안으로 인해 발생하는 수치적 소산이 확산의 효과를 부여하게

된다. 식 (6)의 우변 마지막 항은 사용자의 입력(Input)이 검출 될 시에 입력 좌표 주변 격자 점에 부여하는 상수 값의 밀도 입력(Source)항이다.

최종적으로 계산된 유체를 실시간으로 가시화 하기 위해, 기존의 연구들은 ray-marching 기법을 사용하였다 [17]. 하지만 저사양의 GPU 에서 이러한 기법을 사용하는 것은 무리가 있다. 본 연구에서는 모바일 기기 사용자의 시점이 고정이라는 가정하에, 2 차원 텍스쳐를 알파 블렌딩을 통한 선형 결합하는 형태로 구현하였다. 이는 OpenGL|ES 상으로도 손쉽게 구현 가능하며 적은 량의 계산으로도 가시화 가능하다.

4. 실험 결과

본 논문의 구현은 Apple 의 iPhone SDK 2.0 [19] 기반으로 이루어졌으며, 실험은 iPhone Simulator 를 통해 수행되었다. 그림 1-3 은 그 결과를 보여주고 있다. 연기는 RGB 의 3 색을 가지도록 구현되었으며, 초기 조건은 이미지 파일을 로딩하도록 하였다. 시뮬레이션이 시작되면 부력으로 인해 사용자 반응이 없어도 연기가 자연스럽게 흩어지는 모습을 관찰할 수 있다. 또한, 사용자가 터치를 통해 입력을 가하게 되면 식 (6)를 통해 입력 받은 위치 주의에 연기의 밀도가 높아짐을 볼 수 있다. 시뮬레이션은 1 초당 30 프레임을 진행하였고, 수치 해석을 위한 격자 해상도는 64^2 이다.

5. 결론

본 논문에서는 저사양의 모바일 기기에서 복잡한 유체 방적식을 해석하는 기법을 제시하였다. 이는 기존의 높이장과 같은 단순한 물리 모델이 아닌, 역학적 요소들을 모두 고려한 stable fluids 모델을 기반으로 구현되었다. 하지만 stable fluids 모델은 높이장 모델에 비해 연산량이 상대적으로 높으며, 가장 큰 병목은 포아슨 방정식의 해석 과정에 있다. 이를 위해 본 연구에서는 순환법 기반의 선형 시스템 해석기와 대류항을 통한 예측 모델을 결합함으로써 보다 적은 연산으로 사실적인 수치해석을 구현 가능케 하였다. 또한 본 연구에서는 연기의 대류 방정식의 간단한 변형으로 실시간 사용자 인터랙션이 가미된 기체 시뮬레이션을 구현할 수 있었다.

본 연구는 위의 방법들을 개발함으로써 기존에 존재하는 타 모바일용 물리기반 엔진들의 표현 한계를 좀 더 확장할 수 있게 되었다. 비록 본 연구는 기체에 제한된 결과만을 보여주고 있으나, 추후에는 액체, 혹은 불에 대한 연구를 진행할 계획이다. 또한 가속도 센서와 같은 모바일 기기의 다양한 사용자 인터페이스를 적극 활용하여 유체 역학에 적용하는 것 또한 추후 연구 과제이다.

참고문헌

[1] Jos Stam. Stable Fluids, Computer Graphics (Proc. ACM SIGGRAPH ' 99) 33, Annual Conference Series, 121 128, 1999.

[2] Doyub Kim, Oh-young Song and Hyeong-Seok Ko, Computer Graphics Forum (Proc. Eurographics), Vol. 27, No. 2, 467 475, 2008.

[3] SIO2 Interactive Game Engine, www.sio2interactive.com.

[4] Koi Pond for iPhone, www.theblimppilots.com/The_Blimp_Pilots/Koi_Pond.html.

[5] OpenGL|ES, www.khronos.org/opengles.

[6] Foster, N., and Metaxas, D. Realistic animation of liquids. Graphical models and image processing: GMIP 58, 5, 471 483, 1996.

[7] Foster, N., and Metaxas, D. 1997. Controlling fluid animation. In Computer Graphics International 97, 178 188, 1997.

[8] Foster, N., and Fedkiw, R. Practical animation of liquids. Computer Graphics (Proc. ACM SIGGRAPH 2001) 35, 23 30, 2001.

[9] Enright, D., Marschner, S., and Fedkiw R. Animation and rendering of complex water surfaces. ACM Transactions on Graphics 21, 3, 736 744, 2002.

[10] Fedkiw, R., Stam, J., and Jensen, H. W. Visual simulation of smoke. Computer Graphics (Proc. ACM SIGGRAPH 2001) 35, 15 22, 2001.

[11] Selle, A., Rasmussen, N., and Fedkiw, R. A vortex particle method for smoke, water and explosions. ACM Transactions on Graphics 24, 3, 910 914, 2005.

[12] Song, O.-Y., Shin, H., and Ko, H.-S. Stable but non-dissipative water. ACM Transactions on Graphics 24, 1, 81 97, 2005.

[13] Losasso F., Gibou, F., and Fedkiw, R. Simulating water and smoke with an octree data structure. ACM Trans. Graph. 27, 3, 1 6, 2004.

[14] Bargteil, A. W., Goktekin, T. G., O' brien, J. F., and Strain, J. A. 2006. A semi-lagrangian contouring method for fluid simulation. ACM Trans. Graph. 25, 1, 19 38, 2006.

[15] Houston, B., Nielsen, M. B., Batty, C., Nilsson, O., and Museth, K. Hierarchical rle level set: A compact and versatile deformable surface

representation. ACM Trans. Graph. 25, 1, 151 175, 2006.

[16] Goktekin, T. G., Bargteil, A. W., O'Brien, J. F., "A Method for Animating Viscoelastic Fluids." ACM Transactions on Graphics (Proc. of ACM SIGGRAPH 2004), vol. 23, pp. 463-467, 2004.

[17] Hubert Nguyen, GPU Gems 3, Addison Wesley, 2007.

[18] Saad, Y., Iterative methods for sparse linear systems (2^{nd} edtion), Society for Industrial and Applied Mathematics, 2003.

[19] Apple iPhone SDK, www.apple.com/developer.