Proc. of 3rd KSV Conference
SNU, Seoul, Korea, Apr. 23, 2004

Fluids in Computer Animation

임인성 교수

서강대학교 공과대학 컴퓨터학과

최근 컴퓨터 애니메이션 분야에서는 계산 유체 역학 분야의 기법을 적용하여 다양한 유체 애니메이션 효과를 생성하려는 시도가 활발히 진행되고 있다. 본 강연에서는 연기와 가스, 그리고 물 등의 유체에 대하여 애니메이션 제작 분야에서 제안되고 적용되고 있는 유체 기법에 대하여 설명한 후, 본 서강대학교 컴퓨터 그래픽스 연구실에서 수행하고 있는 유체 애니메이션 관련 연구에 대하여 간략히 알아본다.

강연자 전자메일
ihm@sogang.ac.kr

Fluids in Computer Animation

2004. 4. 23

서강대학교 공과대학 컴퓨터학과
임 인 성

Physically Based Fluids in Computer Animation

?

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

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

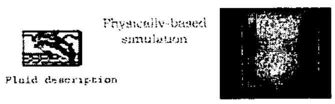
...

$$\phi_t + \mathbf{u} \cdot \nabla \phi = 0$$

$$\phi_t + sgn(\phi)(|P_{\nabla\psi} \cdot \nabla \phi| - 1) = 0$$

Two Major Tasks in Fluid Animation

- How are such natural phenomena as water, smoke, fire, flame, and explosion simulated numerically?



- How are the simulated results visualized realistically?

Important Issues in Fluid Animation

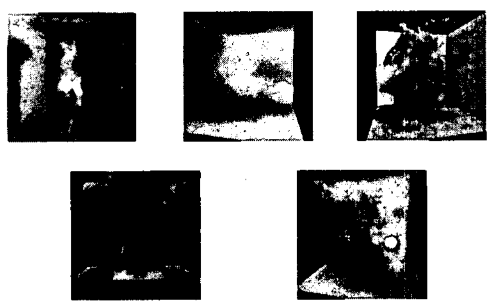
- Which one is more important in the physically based fluid animation?

  Preciseness ⟷ Cost & Complexity / Controllability

  - Preciseness and Realism
    - Physically-convincing motion must be generated.
  - Cost & Complexity
    - 5min x 60sec/min x 30fr/sec x 10min/fr = 90,000min = 62.5days
  - Controllability
    - Animation is all about control!
- Applicability to animation production is crucial.

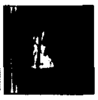
- Is it possible to have a fully-interactive real-time physically-based fluid animation?
  - The current programmable GPUs on PCs will be powerful and flexible enough to make this possible.

Physically Based Simulation of Smoke and Gas

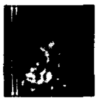


Recent Works on Gas and Smoke Simulations

- J. Stam, "Turbulent Wind Field for Gaseous Phenomena," *ACM SIGGRAPH '93*, pp. 369-376, 1993.



- J. Stam, and E. Fiume, "Depiction of Fire and Other Gaseous Phenomena Using Diffusion Process," *ACM SIGGRAPH '95*, pp. 129-136, 1995.



- N. Foster, and D. Metaxas, "Modeling the motion of a hot, turbulent gas," *ACM SIGGRAPH '97*, pp. 181-188, 1997.

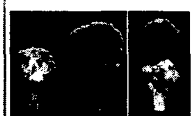
- J. Stam, "Stable Fluids," *ACM SIGGRAPH '99*, pp. 121-128, 1999.

- J. Stam, "Interacting with smoke and fire in real time," *CACM*, Vol. 43, pp. 76-83, 2000.

- R. Fedkiw, and J. Stam, and H. Jenson, "Visual Simulation of Smoke," *ACM SIGGRAPH '01*, pp. 15-22, 2001.

- J. Stam, "A simple fluid solver based on the FFT," Journal of Graphics Tools, Vol. 6, pp. 43-52, 2001.
- J. Stam, "Real-Time Fluid Dynamics for Games," Proceedings of Game Developer Conference, 2003.

- A. Treuille, A. McNamara, Z. Popovic, and J. Stam, "Keyframe Control of Smoke Animation," *ACM Transactions on Graphics (ACM SIGGRAPH '03)*, Vol. 22, No. 3, pp. 716-723, 2003.



- N. Rasmussen, D. Nguyen, W. Geiger, and R. Fedkiw, "Smoke simulation for large scale phenomena," *ACM Transactions on Graphics (ACM SIGGRAPH '03)*, Vol. 22, No. 3, pp. 703-707, 2003.

- X. Wei, Y. Zhao, Z. Fan, W. Li, S. Yoakum-Stover, and A. Kaufman "Blowing in the wind," *ACM Siggraph/Eurographics Symposium on Computer Animation '03*, pp. 75-85, 2003.

Basic Equations in [Stam99]

- The incompressible Navier-Stokes equations

For the velocity $\mathbf{u} = (u, v, w)$,

- Conservation of mass

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

- Conservation of momentum

$$\frac{\partial \mathbf{u}}{\partial t} = \underbrace{-(\mathbf{u} \cdot \nabla)\mathbf{u}}_{\text{advection}} + \underbrace{\nu \nabla \cdot (\nabla \mathbf{u})}_{\text{diffusion}} - \underbrace{\frac{1}{\rho}\nabla p}_{\text{pressure}} + \underbrace{\mathbf{f}}_{\text{external force}}$$

- Helmholtz-Hodge decomposition

  $\mathbf{w} = \mathbf{u} + \nabla q$, where $\nabla \cdot \mathbf{u} = 0$ and q is a scalar field.

  - "Any vector field is the sum of a mass conserving field and a gradient field."

- Projection operator $\mathbf{P}$

  $$\nabla \cdot \mathbf{w} = \nabla^2 q \longrightarrow \mathbf{u} = \mathbf{P}\mathbf{w} = \mathbf{w} - \nabla q$$

- The combined Navier-Stokes equations
  - Using the fact that $\mathbf{P}\mathbf{u} = \mathbf{u}$ and $\mathbf{P}\nabla p = 0$, the following equation is obtained:

  $$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{P}(-(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu \nabla \cdot (\nabla \mathbf{u}) + \mathbf{f})$$

Updating the Velocity Field u

- The general procedure

$u(x, t)$ $\qquad \frac{\partial \mathbf{u}}{\partial t} = \mathbf{P}(-(\mathbf{u}\cdot\nabla)\mathbf{u} + \nu\nabla\cdot(\nabla\mathbf{u}) + \mathbf{f})$ $\qquad u(x, t + \Delta t)$

$w_0(x) \xrightarrow{\text{add force}} w_1(x) \xrightarrow{\text{advect}} w_2(x) \xrightarrow{\text{diffuse}} w_3(x) \xrightarrow{\text{project}} w_4(x)$

1. The *add force* step: **f**

$$\mathbf{w}_1(\mathbf{x}) = \mathbf{w}_0(\mathbf{x}) + \Delta t\,\mathbf{f}(\mathbf{x}, t)$$

2. The *advect* step: $-(\mathbf{w}_1 \cdot \nabla)\,\mathbf{w}_1$
   - Use the method of characteristics for the effect of advection: a semi-Lagrangian scheme

$$\mathbf{w}_2(\mathbf{x}) = \mathbf{w}_1(\mathbf{p}(\mathbf{x}, -\Delta t))$$

where $\mathbf{p}(\mathbf{x}, t)$ is the characteristics of the vector filed.

   - Implementation:
     - Build a particle tracer and linear (or cubic) interpolator.

3. The *diffuse* step: $\nu\nabla \cdot (\nabla \mathbf{w}_2)$
   - Use an implicit method for the effect of viscosity.

$$(\mathbf{I} - \nu\Delta t\nabla^2)\mathbf{w}_3(\mathbf{x}) = \mathbf{w}_2(\mathbf{x}),$$

where **I** is the identity operator.

   - Implementation:
     - Use the linear solver POIS3D form FISHPAK after discretization.

4. The *project* step: $\mathbf{P}(\mathbf{w}_3)$
   - Apply the projection operator to make the velocity field divergent-free.

$$\nabla^2 q = \nabla\cdot\mathbf{w}_3 \longrightarrow \mathbf{w}_4 = \mathbf{w}_3 - \nabla q$$

   - Implementation:
     - Use the linear solver POIS3D form FISHPAK after discretization.

Moving Substances through the Fluid

- A non-reactive substance is advected by the fluid while diffusing at the same time.
- The following equation can be used to evolve density, temperature, texture coordinates, etc.

$$\frac{\partial a}{\partial t} = -\mathbf{u}\cdot\nabla a + \kappa_a\nabla^2 a - \alpha_a a + S_a$$

where κ_a is a diffusion constant, α_a is a dissipation rate, and S_a is a source term.

- Dissipation term

$$(1 + \Delta t\alpha_a)a(\mathbf{x}, t + \Delta t) = a(\mathbf{x}, t)$$

Summary of [Stam99]

- Based on the full Navier-Stokes equations
- Based on an 'unconditionally' stable computational model
  - Semi-Lagrangian integration scheme
- Easy to implement
- Appropriate for gas and smoke
- Suffer from too much 'numerical dissipation'
  - The flow tends to dampen too rapidly.
  - [FedkiwStamJensen01] attempted to solve this problem.

Basic Equations in [FedkiwStamJensen01]

- The incompressible Euler equations
  "Gases are modeled as inviscid, incompressible, constant density fluids."

$$\nabla \cdot \mathbf{u} = 0$$
$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} - \nabla p + \mathbf{f}$$

- The equations for the evolution of the temperature T and the smoke's density ρ

$$\frac{\partial T}{\partial t} = -(\mathbf{u} \cdot \nabla)T$$
$$\frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho$$

Updating the Velocity Field u

1. The *add force* step: **f**
   - Update the velocity field for the effect of forces.

$$\mathbf{f} = \mathbf{f}_{user} + \mathbf{f}_{buoy} + \mathbf{f}_{conf}$$

   - $\mathbf{f}_{user}$: user-defined force (for any purpose)
   - $\mathbf{f}_{buoy}$ gravity and buoyancy forces

$$\mathbf{f}_{buoy} = -\alpha\rho\mathbf{z} + \beta(T - T_{amb})\mathbf{z}, \text{ where } \mathbf{z} = (0,0,1)$$

   - $\mathbf{f}_{conf}$ vorticity confinement force
     - Use a vorticity confinement method by Steinhoff and Underhill.
     - Inject the energy lost due to numerical dissipation back into the fluid using a forcing term.
     - Reduce the numerical dissipation inherent in semi-Langrangian schemes.

$$\omega = \nabla \times \mathbf{u} \;\rightarrow\; \mathrm{N} = \frac{\eta}{|\eta|}\ (\eta = \nabla\,|\,\omega\,|) \;\rightarrow\; \mathbf{f}_{conf} = \epsilon h(N \times \omega)$$

   - Implementation: simple

2. The *advect* step: $-(\mathbf{u} \cdot \nabla)\mathbf{u}$
   - Use the method of characteristics for the effect of advection: a semi-Lagrangian scheme
   - Implementation:
     - Build a particle tracer and linear interpolator.
     - Same as [Stam99]

3. The *project* step: $P(\mathbf{w}_3)$
   - Apply the projection operator to make the velocity field divergent-free.
   - Same as [Stam99]

$$\nabla^2 p = \frac{1}{\Delta t}\nabla \cdot \mathbf{u}^*$$
$$\downarrow$$
$$\mathbf{u} = \mathbf{u}^* - \Delta t \nabla p$$

   - Implementation:
     - Impose free Neumann boundary conditions at the occupied voxels.
     - Use the conjugate gradient method with an incomplete Choleski pre-conditioner.

Moving Substances through the Fluid

- Use the semi-Lagrangian scheme.

$$\frac{\partial T}{\partial t} = -(\mathbf{u} \cdot \nabla)T$$
$$\frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho$$

Updating the Velocity Field in [stam03]

- The general procedure

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{P}(-(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu\nabla \cdot (\nabla \mathbf{u}) + \mathbf{f})$$

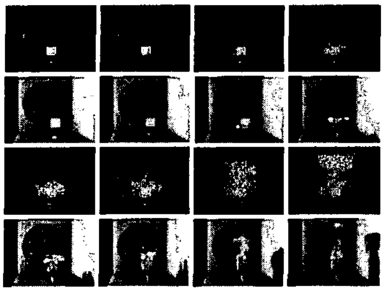
$\mathbf{u}(\mathbf{x}, t)$ … $\mathbf{u}(\mathbf{x}, t + \Delta t)$

$\mathbf{w}_0(\mathbf{x})$ *add force* $\mathbf{w}_1(\mathbf{x})$ *diffuse* $\mathbf{w}_2(\mathbf{x})$ *project* $\mathbf{w}_3(\mathbf{x})$ *advect* $\mathbf{w}_4(\mathbf{x})$ *project* $\mathbf{w}_5(\mathbf{x})$

- The basic integration method is same as in [stam99].
- The Gauss-Seidel method is used for solving the linear system instead of FFT.
  - 20 iterations are enough for the projection operator.

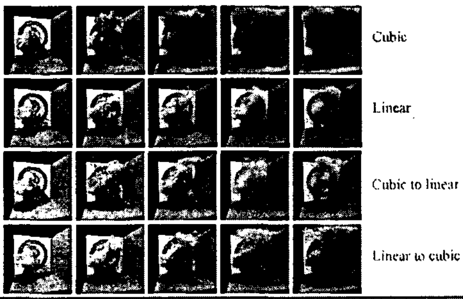
What are we doing with gas and smoke?

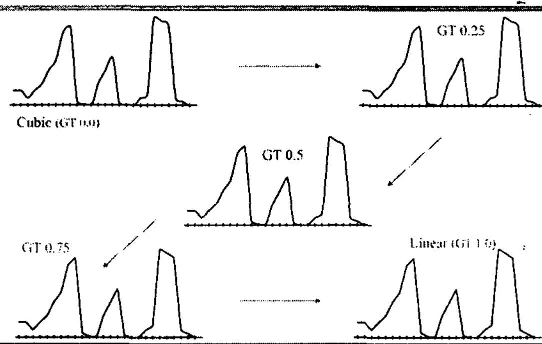
- Development of a stable simulation engine with intuitive user controls



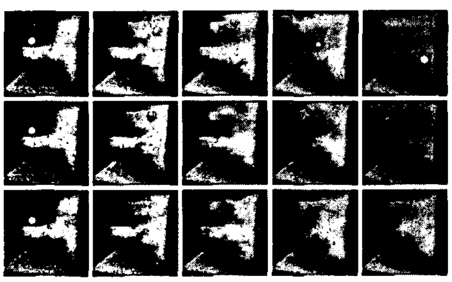
- Application to the animation production

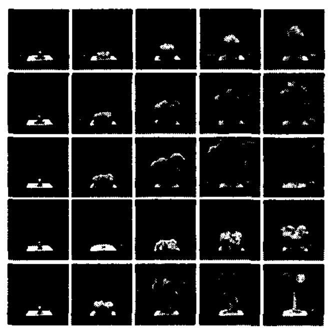
- Design of a controllable monotonic cubic interpolation filter



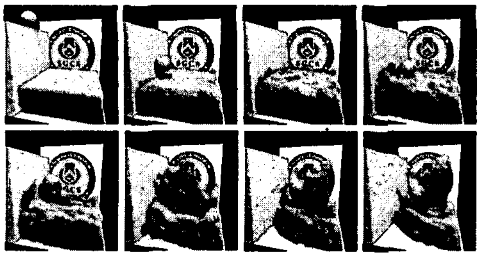


- Animation of chemically reactive fluids



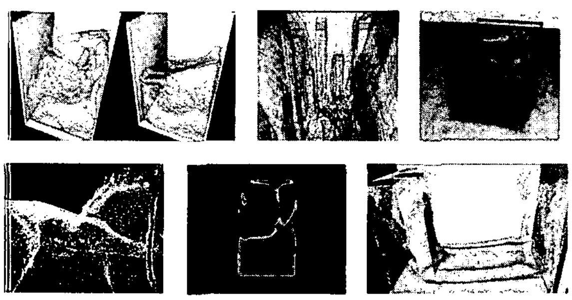


- Effective interaction of moving objects with fluids



- Adaptive and parallel computation of large scale simulations for HD-level animations
- Real-time simulation on programmable graphics processors

Physically Based Simulation of Water



Recent Works on Water Simulation

- N. Foster, and D. Metaxas, "Realistic animation of liquids," *Graphical Models and Image Processing* 58, pp. 471-483, 1996.

- N. Foster, and D. Metaxas, "Controlling fluid animation," *Computer Graphics International '97*, pp 178-188, 1997.

- N. Foster, and R. Fedkiw, "Practical animation of liquids," *ACM SIGGRAPH '01*, pp. 15-22, 2001.

- D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell, "A hybrid particle level set method for improved interface capturing," *J. Comp. Phys.* 183, pp. 83-116, 2002.

- D. Enright, S. Marschner, and R. Fedkiw, "Animation and rendering of complex water surfaces," *ACM Transactions on Graphics (ACM SIGGRAPH '02)*, Vol. 21, No. 3, pp. 736-744, 2002.

- C. Chiu, J. Chuang, C. Lin, and J. Yu, "Modeling Highly Deformable Fluid," http://cggmwww.csie.nctu.edu.tw/images/jwchiu/ModelingLiquid.pdf, 2002.

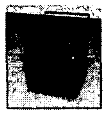
- R. Fedkiw, "Simulating Natural Phenomena for Computer Graphics." *Geometric Level Set Methods in Imaging, Vision and Graphics*, edited by S. Osher and N. Paragios, pp. 461-479, Springer Verlag, New York, 2003.

Cell Properties

- LIQUID
  - This cell is on that is fully filled with liquid. It is used for the calculation of pressure.
- EMPTY
  - This cell is one in which no liquid exists at all.
- SURFACE
  - This cell is one that exists between the LIQUID and the EMPTY cell.
- BOUNDARY
  - This cell is one that is contained in a solid object. It can be wall, obstacle, and other object that water can't flow in. Velocity of this cell is not governed by Navier-Stokes equation.

Basic Algorithm in [FosterFedkiw01]

```
For each time step {
    Update the velocity field by solving the Navier-
    Stokes equations.
    Update the position of the liquid volume using the
    new velocity field.
}
```



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

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

- How do we track the dynamically evolving surface of a volume of liquid?
  - Use a hybrid surface model based on particles and level set
  - Both the Lagrangian and the Eulerian approach

Representing the Water Surface

- Particle versus level set for dynamically evolving surfaces
  - Particle: a Lagrangian approach
    - Easy to move around in the velocity field.
    - Tricky to handle topological changes.
    - Hard to extract a smooth polygonal representation of the liquid surface.

$$\frac{dx_p}{dt} = \mathbf{u}_x$$

  - Level set: an Eulerian approach

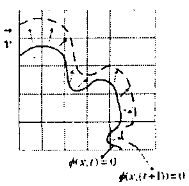
$$\phi_t + \mathbf{u} \cdot \nabla\phi = 0$$

    - Easily produce smooth surfaces.
    - Relatively easy to handle topological changes.
    - Suffer from severe volume loss in under-resolved regions.

Level Set Method

- Traces the interface of moving surfaces using a signed distance function.
- Signed distance function $\phi = \phi(x, y, z, t)$
  - The shortest distance from a point to the interface
  - The sign indicates whether a point is inside, on, or outside of the surfaces.
- The evolution of the interface is described by the following simple convection equation:

$$\phi_t + \mathbf{u} \cdot \nabla\phi = 0$$



- How do we define $\mathbf{u}$ in order to create desired visual effects?
- How do we discretize the equation temporarily and spatially in a stable manner?
  - It is convenient to make the signed distance function to the interface so that $|\nabla\phi| = 1$.
    - Reinitialize the function near the interface.

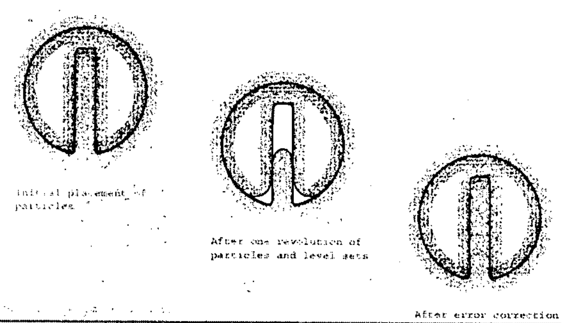
$$\phi_\tau + \frac{\phi}{\sqrt{\phi^2 + |\nabla\phi|^2 \Delta x}}(|\nabla\phi| - 1) = 0$$

where τ is the fictitious time and ϕ_τ is amount to adjust value ϕ.
  - Use a fifth-order accurate Hamilton-Jacobi WENO method to approximate the spatial derivatives.
  - Use a third-order accurate TVD Runge-Kutta method for the temporal integration.

Particle Level Set Method [Enright et al.02]

- A hybrid front tracking technique to model the surface interface that uses marker particles combined with a dynamic implicit surface level set.
- Based on the hybrid method proposed in [FosterFedkiw01].
- The particles are used to correct errors in moving the level set surfaces.

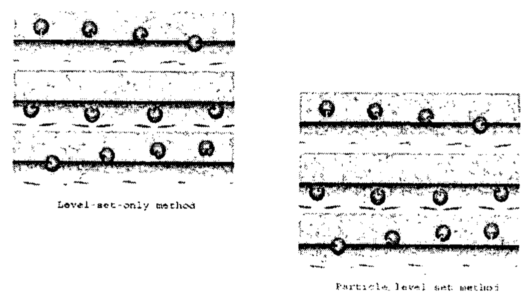
Example of Particle Level Set Method



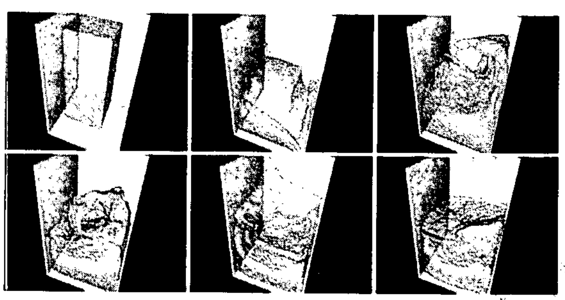
Algorithm of Particle Level Set Method

```
Place both positive and negative particles near the
   interface.
For each time step {
     1. Update the velocity field by solving the Navier-
        Stokes equations.
     2. Evolve both the particles and the level set
        functions forward in time.
     3. Correct errors in the level set function using
        particles.
     4. Apply re-initialization, correcting errors in the
        level set function using particles.
     5. Adjust the particle radii.
     6. Perform particle reseeding, if necessary.
}
```

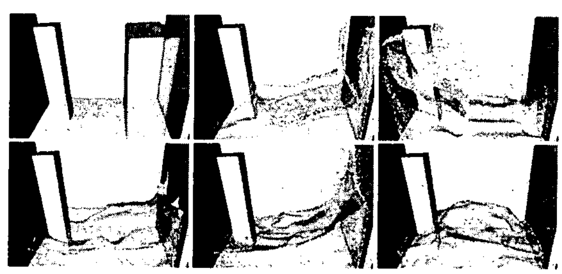

Comparison of the Old and the New Schemes



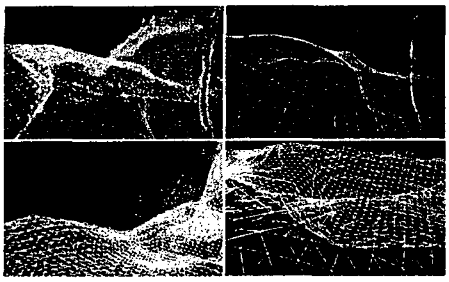
Implementation Results



Particle-only method without smoothing



Particle level set method with smoothing



Polygonized surface with particles

Slices showing positive and negative particles

What are we doing with water and level set?

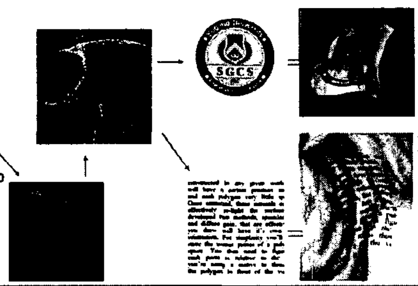
- Development of a stable simulation engine with intuitive user controls

- Local parameterization of polygonal objects using projection level set

$$\phi_t + C|P_{\nabla\psi}\nabla\phi| = 0$$

$$\phi_t + sgn(\phi)(|P_{\nabla\psi}\cdot\nabla\phi| - 1) = 0$$

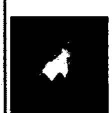
$$sgn(\phi) = \frac{\phi}{\sqrt{\phi^2+|P_{\nabla\psi}\nabla\phi|^2\Delta x}}$$



Physically Based Simulation of Fire, Flame, and Explosion

Recent Works on Fire and Flame Simulation

- P. Beaudoin, S. Paquet, and P. Poulin, "Realistic and controllable fire simulation," *Graphics Interface 2001*, pp. 159 - 166, 2001.



- X. Wei, W. Li, K. Mueller, and A. Kaufman, "Simulating fire with texture splats", *IEEE Visualization 2002*, pp. 227-234, 2002.

- D. Nguyen, R. Fedkiw, and H. Jensen, "Physically based modeling and animation of fire," *ACM Transactions on Graphics (ACM SIGGRAPH '02)*, Vol. 21, No. 3, pp. 721-728, 2002.

- A. Lamorlette, and N. Foster, "Structural modeling of flames for a production environment," *ACM Transactions on Graphics (ACM SIGGRAPH '02)*, Vol. 21, No. 3, pp. 729-735, 2002.

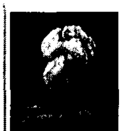
- S. Hasinoff, and K. Kutulakos, "Photo-consistent 3D fire by flame-sheet decomposition", *IEEE International Conf. on Computer Vision*, pp. 1184-1191, 2003.

Recent Works on Explosion Simulation

- M. Neff, and E. Fiume, "A visual model for blast waves and fracture," *Graphics Interface '99*, pp. 193-202, 1999.
- O. Mazarak, C. Martins, and J. Amanatides, "Animating exploding objects," *Graphics Interface '99*, pp. 211-218, 1999.

- G. Yngve, J. O'Brien, and J. Hodgins, "Animating explosions," *ACM SIGGRAPH '00*, pp. 29-36, 2000.

- B. Bashforth, and Y.-H. Yang, "Physics-based explosion modeling," *Graphical Models*, Vol. 63, No. 1, pp. 21-44, 2001.



- B. Feldman, J. O'Brien, and O. Arikan, "Animating suspended particle explosions," *ACM Transactions on Graphics (ACM SIGGRAPH '03)*, Vol. 22, No. 3, pp. 708-715, 2003.

- N. Rasmussen, D. Nguyen, W. Geiger, and R. Fedkiw, "Smoke simulation for large scale phenomena," *ACM Transactions on Graphics (ACM SIGGRAPH '03)*, Vol. 22, No. 3, pp. 703-707, 2003.