Numerical Simulation of Fluids – NuSiF

„Free Boundary Value Problems“

Daniel Brinkers, Florian Forster, Tobias Preclik

Friedrich Alexander Universität Erlangen-Nürnberg

Numerical Simulation of Fluids, 2006-07-24

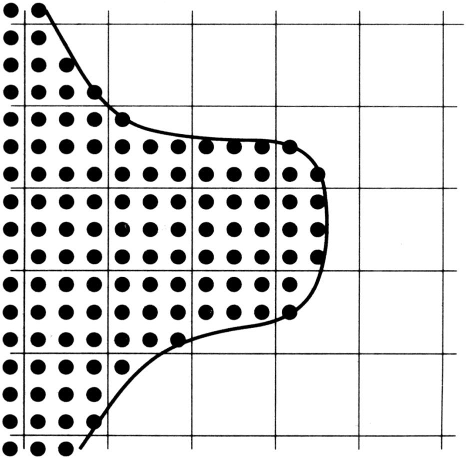
`http://faui2k3.org/svn/nusif/`

Outline

What are „Free Boundary Value Problems“

- Domain Ω has a shared boundary with another fluid
- This domain may change in time
- Many applications fall into this category

Illustration of a „Free Boundary“



Outline

Devision of the domain

- The domain $\mathcal{G}$ is devided into three parts:
  - The time-dependend fluid domain Ω_t
  - The obstacle domain $\mathcal{H}$
  - The empty domain $\mathcal{G} \backslash \{\mathcal{H} \cup \Omega_t\}$
- At each timestep, compute the shape of the fluid domain Ω_t
- Cells which touch the empty domain are considered to be „Surface Cells“ and are handled separately

Tracing the domain

- Need to track the shape of the fluid domain Ω_t.
- Done using (many!) particles: About 3×3 or 4×4 particles per cell.
- The only extension to the basic solver is the handling of the surface cells

Cells with one empty neighbor

- Velocity on the free boundary calculated using the continuity equation, e.g. on a north boundary:

$$v_{i,j} = v_{i,j-1} - \frac{\delta x}{\delta y}(u_{i,j} - u_{i-1,j})$$

- Pressure approximated by:

$$p_{i,j} = \frac{2}{\mathrm{Re}} \frac{v_{i,j} - v_{i,j-1}}{\delta x}$$

- Many values in the empty domain have to be calculated, too

Other cell combinations

- Corner-cells can be handled okay
- Other cells cannot satisfy the boundary conditions
- Such cells are only moved by the „body forces“, e.g. gravity

Outline

Working parts

- The „easy“ surface cells are all calculated correctly, so in general it works
- Arbitrary geometries can be handled

Current/unsolved problems

- „Flying-Super-Cells“[1]: Sometimes a NES- and a NSW-cell fly horizontally through the domain
- Visualization troublesome

[1] I like that term ;)

Outline

Description and motivation

Theory of operation

Our implementation

Visualization techniques

- Image based flow visualization
- Raytracing
- Polygon extraction
- Povray and „blob“s

LIC (line integral convolution)

- Visualize vector field
- Use random texture (white noise)
- Sum up colors along streamlines
- High correlation between two neighbour pixels on a streamline
- Downside: Many calculations

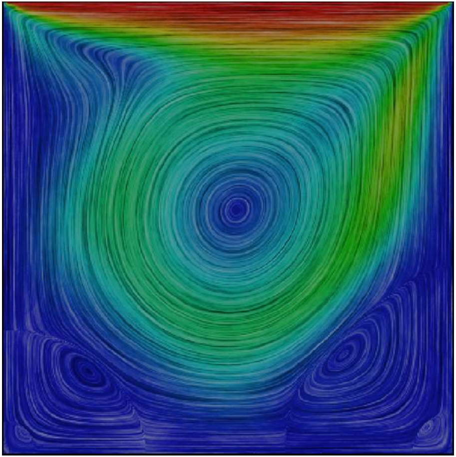
Ibfv (image based flow visualization)

- Uses graphic hardware (OpenGL)
- Random texture is drawn on a plane
- Movement is done by shaping the plane
- New random texture is blended in every frame
- Result is animated
- `http://www.win.tue.nl/~vanwijk/ibfv/`

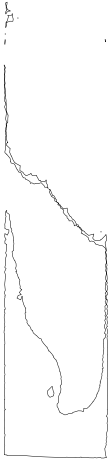
freesurface flows

- Obstacle cells are colored black
- Empty cells are colored white
- Interior cells are moved and blended
- Surface cells are just moved

Illustration of LIC/IBFV



Surface Reconstruction



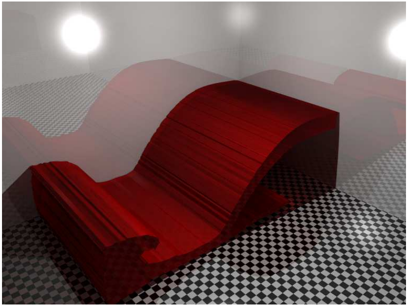
- Iterate along fluid boundaries
- Use fluid particles as polygon vertices
- Eliminate degenerations and tiny polygons

Triangle Mesh Extraction

- Extrude fluid polygons
- Triangulate fluid polygons for front and back face
- Export as a povray scene description

Final Rendering

- Raytrace each frame
- Sleep a night or two
- Wait staring at screen
- Encode video



Using Povray and „blob“s

- Add one sphere per fluid cell and let Povray[2] merge the „drops“
- Very easy to implement
- Fast
- Quite coarse

[2] http://www.povray.org/