Fluids and Solids on Octree Structure

Kupka Milan*
supervised by Roman Durikovic

Faculty of Mathematics Physics and Informatics Comenius University

Abstract

Our fluid simulation runs on an unrestricted octree data structure which uses mesh refinement techniques to enable higher level of detail and solves Navier Stokes equations for multiple fluids. We also show our solution for floating objects and unmoving obstacles by setting up the boundry conditions and solving the velocity of objects using a simple rigid body solver. We propose technique for discretizing the Poisson equation on this octree grid. The resulting linear system is symmetric positive definite enabling the use of fast solution methods. The standard approximation to the Poisson equation on an octree grid results in a non-symmetric linear system which is more computationally challenging. For solving advection velocity of fluid we use the semi-Lagrangian characteristic tracing technique. To track the fluid surface we use Volume of fluid (VOF) method.

Keywords: VOF, Octree, Navier-Stokes, Floating objects, Obstacles

1 Introduction

Interaction between fluid and solid is very common in real world and in computer graphics animation. It can be various objects falling into water, floating on the surface or acting as unmoving obstacle for the flow of fluid. This is also the basic dividing of interactions between fluid and solid objects.

First type of interaction is one-way solid-to-fluid coupling where the motion of solid is predetermined and is not influenced by the velocity of fluid, but the solid influence the velocity of the fluid so it can splash the water as it falls into water or it can be an unmoving obstacle.

Second type of interaction one-way fluid-to-solid coupling where the fluid moves the solid without the solid affecting the fluid. This is why the size of the solid can be from tiny to big object without affecting the motion of the fluid.

Most interesting in the way of simulation and visual effect is the two-way coupling of solids and fluid. It is the 'real world' way of interaction where also the properties of solid, like density, take count in the computation. This type of coupling is a difficult problem. In our solution we combine the velocity of solid that we handle specific when solving N-S equations.

*kupka.milan@gmail.com

2 Previous work

First one who solved three dimensional Navier-Stokes equations was Kass and Miller 1990 [6], and Chen and Lobo 1995 [1] solved the two dimensional Navier-Stokes equations converting to 3D by using height field based on the pressure.They demonstrated both types of one-way coupling. This kind of coupling became common in many other researches. The full three dimensional Navier-Stokes equations were solved in Foster and Metaxas 1996, Foster and Metaxas 1997 and Foster and Metaxas 1997 Siggraph [3] [7] [4] for both water and smoke. Big steps in efficiency were made introducing the use of semi-Lagrangian numerical techniques by Stam 1999 [8]. Another improvemnet of fluid-solid interaction was the tangential movement of fluid along the obstacles presented by Foster and Fedkiw 2001 [2] .

Foster and Metaxas 1996 [3] demonstrate one-way fluid-to-solid coupling where solids are treated as massless particles that move freely on the fluids surface.

Yngve et al. 2000 [11] demonstrated two-way coupling of breaking objects and compressible fluids in explosions, however their technique does not apply to incompressible fluids like water.

Two-way coupling on regural grid was first presented by Takahashi et al. 2002 [10] . To approximate solid-to-fluid coupling they set zero Neumann boundary conditions for the pressure at these boundaries. Later they presented another variation at Takahashi et al. 2003 [9] where they used rigid body solver and fluid solver to achive solid-fluid coupling. Drawback of this techniques is neglecting the dynamic forces and torques of solid objects.

3 Simulation algorithm

Our article follows the simulation steps in our implementations. It consists of

1. Setting the time step size Δt
2. Computing rigid dynamics of solids, Eqs.1 6
3. Setting boundary conditions between solid objects and fluid, see Section 5

4. Solving Navier - Stokes equations, Eqs.7, 8

5. Computation of new VOF values, Eqs.17, 18

4 Rigid solids dynamics

In our simulation we simplify the movement of solid objects to translation an rotation of center of mass CM, which position and rotation matrix is saved in global variables. The solid is represented in the enviroment as another fluid, but with restrictions in later steps of simulation, when solving Navier-Stokes equations. These restrictions will be explained later in this section. First, we have to calculate forces acting on our solid. We compute them from velocities of the fluid, from previus time step, in the fluid cells neighbouring to cells with 'solid fluid'. It is easy to find the vector from point of force activity to CM, because we save the CM position. We sum the forces to find the total force and torque:

$$F^n = \sum_i F_i, \tau^n = \sum_i r_i \times F_i, \tag{1}$$

where F_i is force acting on solid and r_i is a vector from point of force activity to CM. This values can be used to integrate the position, velocity, orientation and angular velocity. We do this by solving the following equations:

$$r_{CM}^{n+1} = r_{CM}^n + \Delta t v_{CM}^n, \tag{2}$$

where Δt is the timestep, v_{CM}^n is the velocity of CM and r_{CM}^{n+1} is a new position.

$$v_{CM}^{n+1} = v_{CM}^n + \Delta t \frac{F^n}{M} \tag{3}$$

M is the mass of solid calculated from volume of 'solid fluid' and density.

$$A^{n+1} = A^n + \Delta t \omega^n A^N \tag{4}$$

A^{n+1} is the orientation of the solid needed later for visualisation of solid. The visualisation process is explained in Results section. Last integration equation is the computation of angular momentum L_{CM}^{n+1}.

$$L_{CM}^{n+1} = L_{CM}^n + \Delta t \tau^n \tag{5}$$

We compute the angular velocity equation.

$$\omega^{n+1} = I L_{CM}^{n+1} \tag{6}$$

where I is inertia of solid. After sovling the dynamics of solid object we set the computed velocities as 'solid fluid' velocities. Now the simulation of fluid can continue, but the 'solid fluid' velocities aren't changed until the next simulation step and recomputation of solid dynamics again.

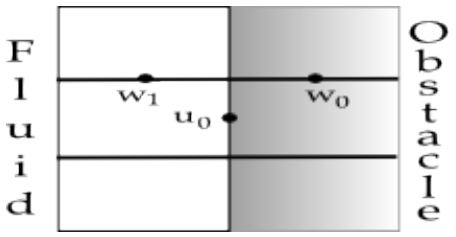

Figure 1: Boundary conditions

5 Boundary conditions

To prevent fluid to flow into the obstacle, specific boundary conditions must be set. This process contains manualy seting up the velocity on faces between fluid and obstacle. The situation is outlined on figure 1.

First the normal velocity on boundary face is set to zero $u_0 = 0$. The pressure of obstacle cell is set as the adjacent fluid cell. This is called Neumann boundary conditions.

Later in the process we also set the tangential velocity in obstacle w_0. Depending on the type of slip conditions we have two opportunities. Free-slip boundary $w_0 = w_1$ and no-slip boundary $w_0 = -w_1$.

In many simulations we have to set also surface boundarys to preserve mass. In our simulation we use another fluid to simulate air. Due to this fact empty cells are eliminated and surface cells are cells containig both fluids so there is no need for surface boundarys.

6 Navier-Stokes equations

The Navier-Stokes equations for incompressible fluids are

$$\frac{\partial u}{\partial t} = -(u.\nabla)u + \nabla(\upsilon \nabla u) - \frac{1}{\rho}\nabla p + \frac{f}{\rho} \tag{7}$$

$$\nabla.u = 0 \tag{8}$$

where u is velocity vector consisting of (u,v,w), ρ is density constant, υ is viscosity constant, f is the external force (f,g,h), mostly it is only the gravitation force $\left(f = h = 0; g = 9.8ms^2\right)$, p is the pressure and ∇ is vector of partial derivation $\left(\frac{\partial}{\partial x}; \frac{\partial}{\partial y}; \frac{\partial}{\partial z}\right)$. Equation 7 is the law of momentum and equation 8 represent the conservation of mass. On left side of equation 7 is the change of velocity of the fluid and on the right side are the acting forces on fluids like advection, diffusion, pressure and external forces.

Some motion elements, turbulence and surface tension, are not included in these two equations, but we assume their effects are dominated by the above velocity and forces.

In our simulation we solve this equations discreeted on MAC grid, so we are able to use finite difference method. In the gird cell the velocities are defined on faces of the cell and pressure, viscosity and density in the middle.

To solve N-S equations we use Helmholtz - Hodgeo decomposition of vector field so we get the potencial and gradient component.

$$\tilde{u} = u + \nabla q \tag{9}$$

We get the gradient component by solving Poisson equation

$$\nabla . u = \nabla^2 q \tag{10}$$

To do so we define operator P that project vector field into his potencial component $u = P(\tilde{u}) = \tilde{u} - \nabla q$. After application of P operator on first N-S equation we get

$$u^{nova} = P\left[u + \Delta t\{-(u.\nabla)u + \upsilon\nabla^2 u + \frac{f}{\rho}\}\right] \tag{11}$$

where the ∇ operator on MAC grid cell with index i,j,k is sum of

$$\frac{u_{i+1,j,k} - u_{i,j,k}}{\Delta x} + \frac{v_{i,j+1,k} - v_{i,j,k}}{\Delta y} + \frac{w_{i,j,k+1} - w_{i,j,k}}{\Delta z} \tag{12}$$

Solving this equation consists of two steps. First we compute *guess velocities* by the equation

$$\tilde{u} = u + \Delta t\{-(u.\nabla)u + \upsilon\nabla^2 u + \frac{f}{\rho}\} \tag{13}$$

Now we have velocities of fluid by the advection, diffusion and external forces. The second step is *pressure projection*. This means that after second step would $\nabla . u = 0$.

$$\nabla . u^{nova} = \nabla . \left(\tilde{u} - \frac{\Delta t}{\rho}\nabla p\right) = \nabla . \tilde{u} - \frac{\Delta t}{\rho}\nabla^2 p = 0 \tag{14}$$

and after solving the poisson equation

$$\nabla^2 . p = \frac{\rho}{\Delta t}\nabla . \tilde{u} \tag{15}$$

we get new velocity

$$u^{nova} = \tilde{u} - \frac{\Delta t}{\rho}\nabla p \tag{16}$$

7 VOF method

As a method of surface tracking we choosed VOF. It is a method based on jumping function F. This function designates volume of fluid of cell where 1 means cell is full of fluid and 0 means empty.

It was presented in Numata,Durikovic [5]. The difference of our method is the use of this method for the blend of fluids.

F is changed by the velocity of fluid, to compute it we use donor-acceptor schema. The aproximation by finite diferences would "blur" the surface of fluid so the sharp profile of surface is lost.

We outline the computing of VOF. Volume of stream that flows from donor to aceptor is $|V_i| = u_i \Delta t \Delta S_{ss}$ where u is normal velocity and number sign defines which cell is donor and which is acceptor. The ΔS_{ss} is the cubic capacity of the smaller of the two faces. Volume of F that flows trough face of the cell in time step Δt is

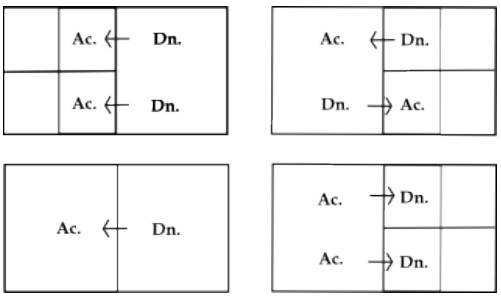

Figure 2: Donor-Acceptor situations

$$|V_n| = min\left(F_{i_n}^{AD}|V_i| + CF_{i_n}, F_{i_n}^{D}|V_D|\right) \tag{17}$$

where

$$CF_{i_n} = max\left(\left(1.0 - F_{i_n}^{AD}\right)|V_i| - \left(\sum_i F_i^D - F_{i_n}^D\right)|V_D|, 0.0\right) \tag{18}$$

Indexis D,A denote donor and acceptor while double index AD denotes donor or acceptor by the orientation of interface in respect of direction of the flow.

The minimum in equation 17 restrain the cell to give away more fluid it has and the maximum in equation 18 assures additive flow CF if volume of air that should be transferd is larger then accessible.

Few examples of donor acceptor situations are shown in figure 2.

IF an error occurs and some cell has the F value > 0 we proportionaly distribute excess fluid back to donor cells.

8 Rewriting the methods on octree structure

In our simulation we use octree data structure. Advantages of this method are less computational demands at preservation of level of details. We use the ability of AMR (AMR - adaptive mesh refinement) in visualy pleasent sections. Disadvantage is the necessity of recomputing the velocity, density and pressure and VOF values when changing the level of octree.

8.1 Velocity recomputation

We show the method of velocity recomputation on a 2D example in figure 3. Let us explain how to find velocity C5 from figure 3.

First, we compute nodal velocities on C5 which are v1...v4. Nodal velocity is average of face velocities the node is a part of. Than we compute edge velocities v12, v13, v24, v34 as average of adjacent nodal velocities. Average of this edge velocities gives us central velocity vc1

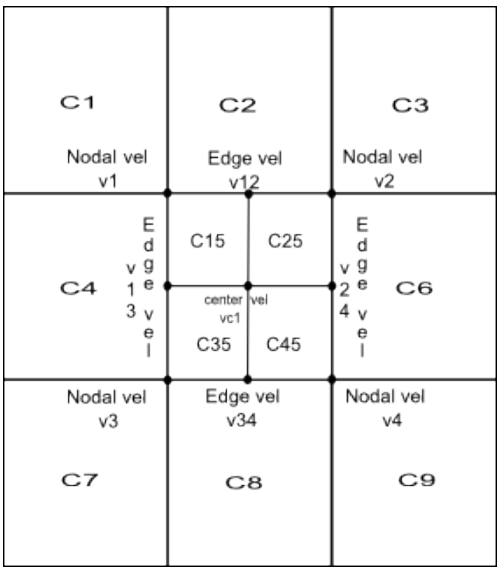

Figure 3: Velocity recomputation

The final velocities of new subfaces of face C5 are average of their nodal velocitys.

$$c15 = \frac{v1 + v12 + vc1 + v13}{4} \tag{19}$$

The other three subface velocities are computed similarly.

Other way around when combining 4 faces to a new face we just average their velocities.

8.2 Density and pressure recomputation

We deal with pressure and density similar way as with velocities. When dividing cells we just set the values of child pressure and density the same as parent cell. When combining we set the value as average of child cells values.

8.3 VOF recomputation

When combining child cells to a parent cell we do the weighted average of child VOF values

$$F_n = \frac{\sum_{i=1}^{8} F_{i_n} |V_i|}{|V|} \tag{20}$$

Dividing of parent cell is a little more complex problem because we need to be sure there won't be any holes in the fluid after dividing. The value of VOF of the child cell is

$$F_{i_n} = min\left(1.0, 8F_n \frac{\sum_{neigh(i)} F_{s_n}}{\sum_{j=1}^{8}\left(\sum_{neigh(j)} F_{s_n}\right)}\right), \tag{21}$$

where n is the index of fluid, i is the index of child cell and s is the index of neigbour of child cell. Neighbour of cell is every cell that has common face with child cell i and one neighbour cell with commnon node. None of these neighbours is another child cell.

9 Results

The results of our simulation are stored in series of xml files. One xml file for each simulation step and one PovRay [5] file, where the positions of obstacles are defined. All files are combined to form the scene file that is rendered by PovRay raytracing program. The bubble picture shows interaction of two fluids air and water with static obstacle. Second pictures show water floating down the stairs. Following tables shows properties of scenes.

Bubble		
# of cells	**Start of scene**	**End of scene**
total	512	512
air fluid (bubble)	91(27)	91(3)
water fluid	389	389
obstacles	32	32
# of timesteps	50	
time of computation	8 [sec]	

Steps		
# of cells	**Start of scene**	**End of scene**
total	512	512
air fluid	48	48
water fluid	374	374
obstacles	90	90
# of timesteps	70	
time of computation	10 [sec]	

Number of cells occupied by air and water fluid stays the same at the start and the end, that shows conservation of volume in this case. For both scenes fluids with properties shown in following table were used.

Fluids			
Fluid	**Density**	**Viscosity**	**Surface tension**
air fluid	0.1 $[kg/m^3]$	1 $[Pa \cdot s]$	0 $[N/m]$
water fluid	10 $[kg/m^3]$	0.1 $[Pa \cdot s]$	35 $[N/m]$

References

[1] J. X. CHEN and N. DA VITORIA LOBO. Toward interactive-rate simulation of fluids with moving obstacles using navier-stokes equations. *Graph. Models Image Process.*, 1995.

[2] N. FOSTER and R. FEDKIW. Practical animation of liquids. *In Proceedings of ACM SIGGRAPH 2001, Computer Graphics Proceedings, Annual Conference Series*, 2001.

[3] N. FOSTER and D. METAXAS. Realistic animation of liquids. *Graph. Models Image Process.*, 1996.

[4] N. FOSTER and D. METAXAS. Modeling the motion of a hot, turbulent gas. *In Proc. of SIGGRAPH 97*, 1997.

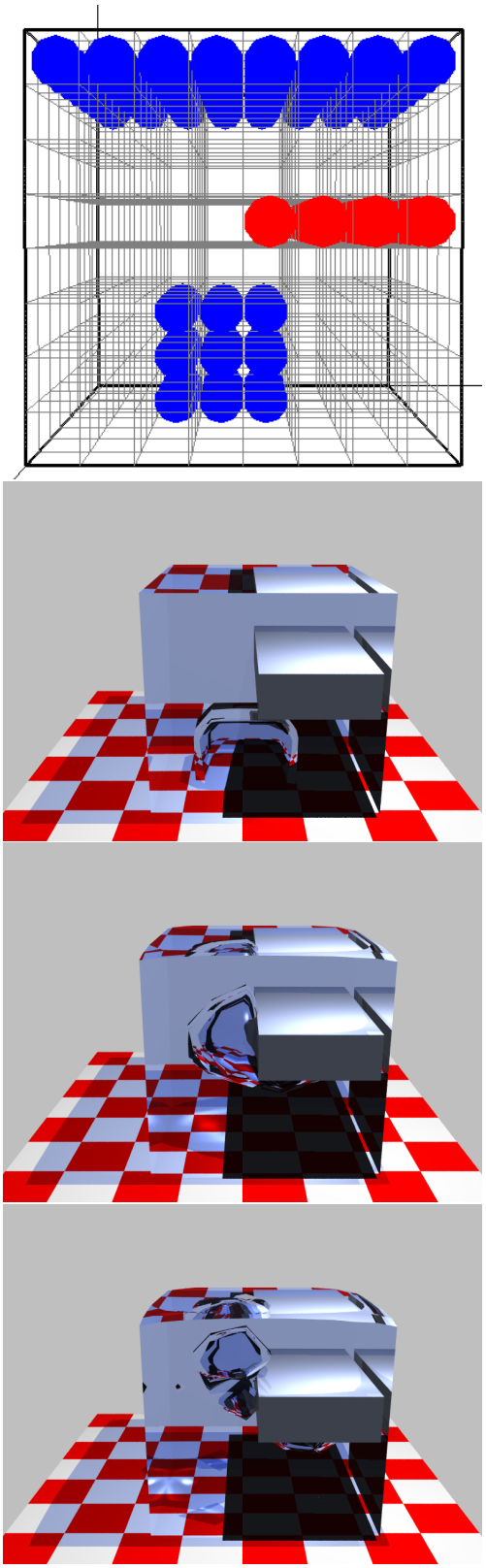

Figure 4: Bubble

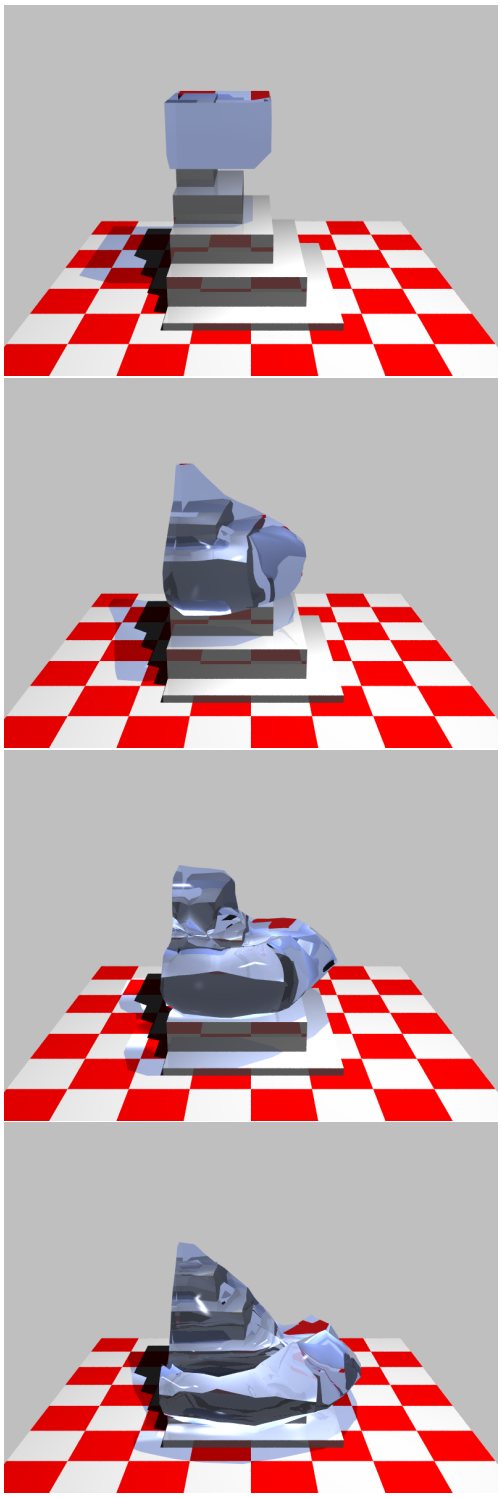

Figure 5: Steps

[5] Numata K. Animation of fluids using multiphase flow approach. Master's thesis, University of Aizu, 2006.

[6] M. Kass and G. Miller. Rapid, stable fluid dynamics for computer graphics. *ACM SIGGRAPH*, 1990.

[7] N. and D. METAXAS. Controlling fluid animation. in cgi. *Proc. of the 1997 Conference on Computer Graphics International*, 1997.

[8] J. STAM. Stable fluids. *In Proceedings of SIGGRAPH 99, Computer Graphics Proceedings, Annual Conference Series*, 1999.

[9] FUJII H. KUNIMATSU A. HIWADA K. SAITO T. TANAKA K. TAKAHASHI, T. and H. UEKI. Realistic animation of fluid with splash and foam. *Computer Graphics Forum 22*, 2003.

[10] HEIHACHI U. TAKAHASHI, T. and A. KUNIMATSU. The simulation of fluid-rigid body interaction. *In SIGGRAPH 2002: Sketches Applications*, 2000.

[11] OBRIEN J. F. YNGVE, G. D. and J. K. HODGINS. Animating explosions. *In Proceedings of SIGGRAPH 2000, Computer Graphics Proceedings, Annual Conference*, 2000.