Multifluid Finite Volume Navier-Stokes Solutions for Realistic Fluid Animation

John A. Turner   Andre C. Mazzone

Blue Sky Studios, Inc.

1 Introduction

It is commonly believed that solution of the full three-dimensional (3D) Navier-Stokes equations is too computationally intensive for computer graphics applications. Previous approaches have typically used either the simplified shallow-water approximations [3] or, most recently, two-stage approaches involving a low resolution 3D Navier-Stokes solution followed by a height-field solution [2]. However, recent advances in incompressible free surface flow algorithms and methods for the solution of linear systems of equations make high-resolution solution of the full 3D Navier-Stokes equations possible.

2 Mizu: A Casting Simulation Tool

A computational tool, referred to herein as Mizu[1], has been developed at Los Alamos National Laboratory for the simulation of casting processes, *i.e.* the filling, cooling, and solidification of molten fluid in molds with complex geometry[2]. Such simulation involves modeling physical phenomena such as unsteady, incompressible (or slightly compressible) flow of multiple, immiscible fluids, interface physics (*e.g.* surface tension), convective, diffusive, and radiative heat transfer, solidification of multi-component alloy systems, microstructural physics (*e.g.* nucleation, dendrite growth), and material response effects (*e.g.* stress, distortion, shrinkage, plastic flow). Accurate simulation of these processes can improve the ability to control the microstructural properties during the casting process so as to minimize defects in the final cast part, saving time and money.

Obviously, not all of these processes are necessary for computer graphics applications, so we will concentrate on the flow simulation. Further details of the other aspects can be found elsewhere [6].

Mizu solves the 3D, incompressible, variable-density Navier-Stokes equations on generalized-connectivity unstructured (GU) meshes. The use of GU meshes, which can contain hexahedra, tetrahedra, prisms, and pyramids, enables simulation of arbitrarily complex geometry, crucial for both mold filling and computer graphics applications. Note that these are volumetric meshes, so the availability of tools for generating high-quality GU meshes from surface meshes created by CAD software and modeling tools such as Maya and Softimage is essential.

The flow algorithm is a 3D extension of recent advances in projection methods [1, 8, 5], and is based on a colocated, cell-centered, finite volume formulation that is 2^{nd}-order in both time and space[3] [7]. The algorithm is implicit, and hence requires the solution of linear systems of equations at each timestep. The solver library used [9] makes use of recent advances in iterative solution techniques, specifically preconditioned Krylov subspace methods.

Note that energy is included in the equation set, so flows driven by density variations, *i.e.* natural circulation, can be simulated. An example would be a lava lamp.

Since the flow involves multiple materials, a critical aspect of the simulation involves tracking the interface between materials. Note that the flow of wine into a glass is a multimaterial problem, the two materials being wine and air, and that material properties such as density vary by several orders of magnitude over an extremely small distance at the interface. In addition, the interface itself is topologically complex, and physical properties such as surface tension play a significant role in the behavior of the flow. Hence, realistic fluid simulation requires accurate modeling of these interfaces.

There are a number of approaches to interface tracking, including moving-mesh (Lagrangian) methods, front-tracking methods, particle-based methods, boundary integral methods, *etc*[4]. Mizu uses a volume tracking, or volume of fluid (VOF), approach. Interfaces are tracked on 3D generalized hexahedral cells and localized over one cell width at each timestep. They are assumed to be planar within each cell, yielding a globally piecewise planar approximation to the actual interface.

3 Rendering

Once the flow conditions are computed, the results are rendered as spherical metaballs (blobs), with blob radius determined by the volume fraction of the appropriate fluid. While this smears some of the fine detail of the solution, and would likely not be appropriate for scientific visualization, it results in motion realistic enough for film and commercial applications.

The results were rendered using Blue Sky Studios, Inc.'s proprietary ray-tracing renderer, CGI Studio.

As an initial test, we simulated a fluid being poured into an glass box eight centimeters on each side. Since the geometry of this situation is simple, a 32x32x32 orthogonal mesh was used, resulting in 32,768 computational cells of size 0.25 cm on each side. The calculation thus requires solution of linear systems involving this number of unknowns at each timestep. While this mesh is somewhat coarse, it nevertheless yields fairly realistic results.

The simulation was performed primarily using 250 MHz R10000 processors on an SGI Origin 2000 (although Mizu provides for parallel execution in either SMP or distributed modes *via* MPI, these calculations were performed in serial

[1] Japanese for *water*.

[2] It must be noted that Mizu has not been released publicly, and is in a state of development. It was only available to us due to the fact that one of us (Turner) was involved in its development while at LANL. More information on Mizu can be found at http://www.zephyr-group.com/mizu/.

[3] That is, if the mesh spacing is decreased by a factor of two, accuracy improves by a factor of four.

[4] A thorough discussion of these can be found in [4].

mode). CPU time requirements were significant, but not unreasonable. Early in the simulation, when the flow is complex and difficult to compute, roughly an hour of CPU time was required for each frame of animation. Near the end of the simulation, when the flow is nearing steady-state, only a few minutes per frame were required.

The resulting animation sequences are quite realistic, particularly the degree to which the complex topology of the interface is captured. Two selected frames are shown in Figures 1 and 2.

4 Future Improvements

Currently, blobs are placed at the centroid of each cell. More realistic results could be achieved if they were placed at the centroid of the region of the cell occupied by the fluid being rendered. Since a planar interface is reconstructed in the course of the flow calculation, this information is available.

In addition, blobs need not be spherical. That is, ellipsoids which “fit” the cell region in question, using cell geometry and the reconstructed fluid interface, can be used. This would further enhance the realism of the final animation.

5 Conclusions

We feel that while simplified approaches are sufficient for many computer graphics applications, truly realistic fluid animation requires high-resolution solutions of the Navier-Stokes equations. We have presented initial results toward that end, demonstrating that while such simulations are computationally intensive, they are possible and yield promising results.

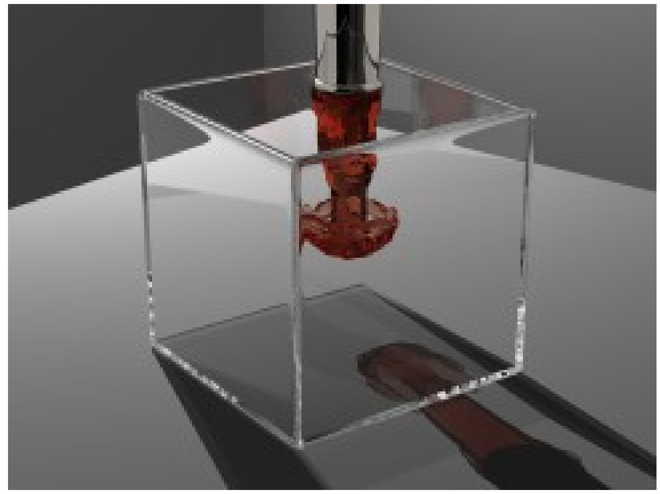

Figure 1: Simulation time: 0.12 seconds.

Figure 2: Simulation time: 0.77 seconds.

References

[1] T. J. Barth. Aspects of unstructured grids and finite-volume solvers for Euler and Navier-Stokes equations, 1995. VKI/NASA/AGARD Special Course on Unstructured Grid Methods for Advection Dominated Flows, AGARD Publication R-787.

[2] N. Foster and D. Metaxas. Realistic animation of liquids. In *Proceedings GI*, pages 204–212, 1996.

[3] M. Kass and G. Miller. Rapid, stable fluid dynamics for computer graphics. In *Proceedings of SIGGRAPH '90, in Computer Graphics*, volume 24, pages 49–57, 1990.

[4] D. B. Kothe. Perspective on Eulerian finite volume methods for incompressible interfacial flows. In H. Kuhlmann and H. Rath, editors, *Free Surface Flows*, New York, NY, 1998. Springer-Verlag.

[5] E. G. Puckett, A. S. Almgren, J. B. Bell, D. L. Marcus, and W. J. Rider. A second-order projection method for tracking fluid interfaces in variable density incompressible flows. *Journal of Computational Physics*, 130:269–282, 1997.

[6] A. V. Reddy, D. B. Kothe, C. Beckermann, R. C. Ferrell, and K. L. Lam. High resolution finite volume parallel simulations of mould filling and binary solidification on unstructured 3D meshes. In *Solidification Processing 1997: Proceedings of the Fourth Decennial International Conference on Solidification Processing*, pages 83–87, The University of Sheffield, Sheffield, UK, July 7–10, 1997.

[7] W. J. Rider, D. B. Kothe, S. J. Mosso, J. H. Cerruti, and J. I. Hochstein. Accurate solution algorithms for incompressible multiphase fluid flows. Technical Report AIAA 95–0699, AIAA, 1995. Presented at the 33rd Aerospace Sciences Meeting and Exhibit.

[8] W. J. Rider, D. B. Kothe, E. G. Puckett, and I. D. Aleinov. Accurate and robust methods for variable density incompressible flows with discontinuities. In V. Venkatakrishnan, M. D. Salas, and S. R. Chakravarthy, editors, *Workshop on Barriers and Challenges in Computational Fluid Dynamics*, pages 213–230, Boston, MA, 1998. Kluwer Academic Publishers.

[9] J. A. Turner, R. C. Ferrell, and D. B. Kothe. JTpack90: A parallel, object-based Fortran 90 linear algebra package. In *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing*, Minneapolis, MN, March 14–17, 1997.