Institute of Physics Publishing Journal of Physics: Conference Series **16** (2005) 471–475
doi:10.1088/1742-6596/16/1/064 SciDAC 2005

A TSTT integrated FronTier code and its applications in computational fluid physics

Brian Fix[1], **James Glimm**[1,2], **Xiaolin Li**[1,3], **Yuanhua Li**[1], **Xinfeng Liu**[1], **Roman Samulyak**[2], **Zhiliang Xu**[2]

Department of Applied Math and Stat
SUNY at Stony Brook
Stony Brook, NY 11794[1]

Computational Science Center
Brookhaven National Laboratory
Upton, NY 11793[2]

E-mail: `linli@ams.sunysb.edu`[3]

Abstract. We introduce the FronTier-Lite software package and its adaptation to the TSTT geometry and mesh entity data interface. This package is extracted from the original front tracking code for general purpose scientific and engineering applications. The package contains a static interface library and a dynamic front propagation library. It can be used in research of different scientific problems. We demonstrate the application of FronTier in the simulations of fuel injection jet, the fusion pellet injection and fluid mixing problems.

1. Introduction

A general purpose software package for the geometry and dynamics of a moving manifold has been extracted from the FronTier code and adapted to the TSTT mesh entity data interface for applications in different scientific and engineering problems. In this paper, we describe significant improvements to the front tracking package, including high order front propagation and 3D handling of topological bifurcations. We assess the performance of the package, in comparison with publicly distributed interface codes (the level set method) and with published performance results (VOF and other methods).

The code is integrated with the mesh entity data interface designed by the TSTT (Terascale Simulation Tools and Technology) Center. It is downloadable from the web. It is accompanied by a web based testing and evaluation site and extensive web based documentation. We name this newly extracted software package the FronTier-Lite code. We have put the front tracking package in the public domain:

`http://www.ams.sunysb.edu/FTdownload`

We have used this software package for simulations of several SciDAC problems including the fuel-injection jet, the study of Rayleigh-Taylor mixing and the fusion pellet injection into a tokamak.

2. High Order Front Propagation

We extracted from the original FronTier code the physics independent part to form the FronTier-Lite libraries. The FronTier-Lite package allows users to provide their own velocity functions for the front propagation. To do so, the user needs to create a data structure containing all necessary parameters for the velocity function. This data structure is casted into a pointer after it is initialized. It will be passed to the velocity function, which is also a pre-assigned anonymous pointer. Inside the velocity function, the pointer of velocity parameters will be cast back to point to the original data structure and the velocity function retrieves its parameters for the computation of velocity of each marker point.

There are two types of point propagation functions. The first one is for the velocity field which is a function of space coordinates and time only

$$\mathbf{v} = \mathbf{v}(\mathbf{x}, t). \tag{1}$$

In such a velocity field, the moving front is advanced through propagation of the discretized interface points via a simple ordinary equation

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}, t). \tag{2}$$

The package provides both first order Euler forward and fourth order Runge-Kutta solvers to solve for Equation 2.

The second type of front propagator is the hyper-surface propagator. This propagation function is needed when the velocity of the front is not a pure field function of space and time, but is dependent on the front geometry as well. Typical examples are the normal propagation of a front and curvature dependent front propagation,

$$\mathbf{v} = \mathbf{v}(\mathbf{x}, t, \mathbf{N}, \kappa). \tag{3}$$

Since as a front point propagates, so do its neighbors and thus $\mathbf{N}$ and κ change. For each sub-step in the Runge-Kutta method, to ensure a uniform order of truncation error, the geometry variables (normal and curvature) must be updated. Therefore, the surface must be advanced as a whole in each sub-step.

3. Locally Grid-Based Bifurcation

The grid-free (GF) tracking is a fully Lagrangian tracking method [3] which encountered difficulty in handling the interface tangling after the front propagation. To resolve the complexity in topological bifurcation, the front tracking code adopted the grid-based reconstruction method [4] similar to the marching cube method by Loreson and Cline [7]. However, recently it has been analyzed that the grid-based reconstruction suffers large truncation error when the curvature of the front is large [1].

To reduce the GB interface interpolation error, we introduce a new method, LGB, or the locally grid based tracking, which combines the advantages of both the GF and the GB methods. We use the GF method to propagate the interface to obtain an accurate solution of the interface position. Eulerian GB reconstruction of the interface is only used in small regions where topological bifurcation is detected. This method reduces the use of Eulerian reconstruction to its minimum.

4. Benchmark Tests of FronTier-Lite

As the first example, we present the benchmark test of the rotation of a slotted rigid disk. Our comparison is made between the FronTier code and the level set ToolBox package downloaded from the public website:

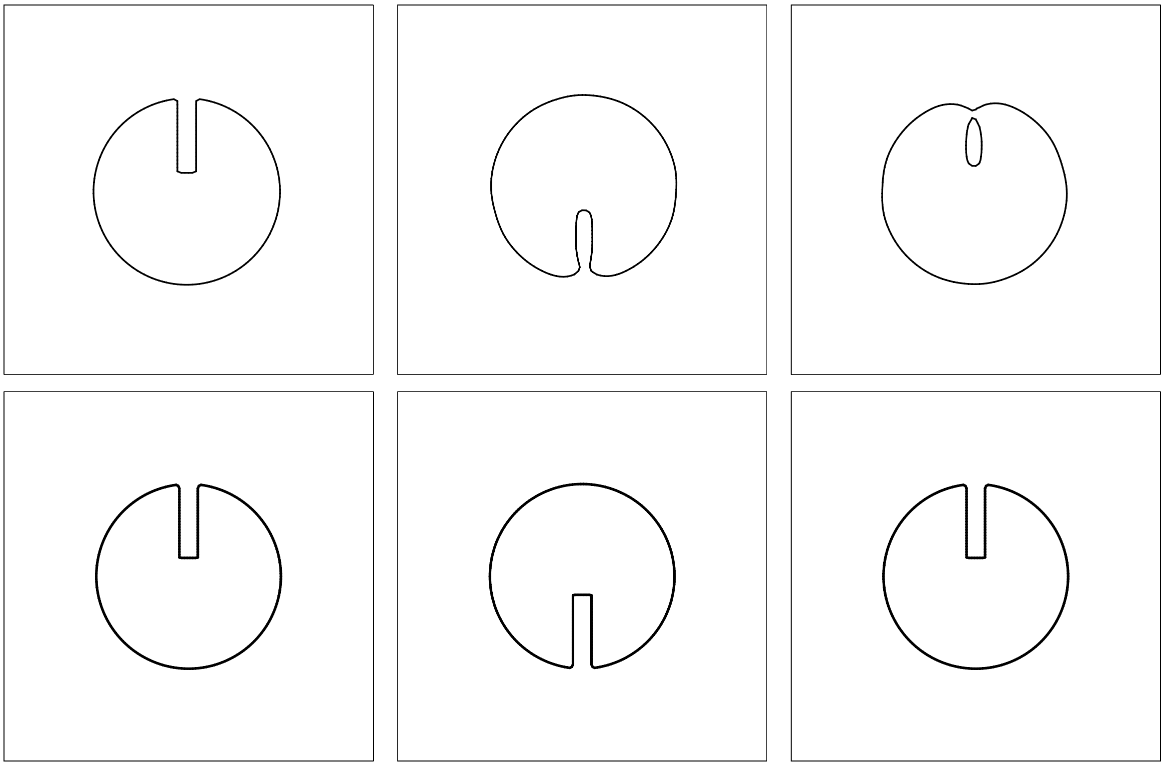

Figure 1. Comparison of slotted disk simulation using high order methods. The upper sequence shows the result of the level set method using the fifth order WENO scheme and the lower sequence shows the result of front tracking using the fourth order Runge-Kutta method.

`http://www.cs.ubc.ca/ mitchell/ToolboxLS`

In this comparison, we used the fifth order WENO scheme for the convection of the level set function, while for the front tracking code, we used the fourth order Runge-Kutta method for the point propagation. We continued the computation for 13 revolutions. The fourth order Runge-Kutta method appears to be extremely accurate in the front tracking simulation. Even after 13 rounds of rotation, the changes of both radius of the disk and the slot are invisible. The level set computation began to suffer edge smoothing after the second rotation. At the end of the 13th cycle, the slot is closed at the top resulting in a topologically incorrect bifurcation.

The second benchmark test deals with a three dimensional interface in a deformation velocity field. The velocity field in this experiment is described by the equations

$$u(x,y,z) = 2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z)\cos(\pi t/T) \quad (4)$$
$$v(x,y,z) = -\sin(2\pi x)\sin^2(\pi y)\sin(2\pi z)\cos(\pi t/T) \quad (5)$$
$$w(x,y,z) = -\sin(2\pi x)\sin(2\pi y)\sin^2(\pi z)\cos(\pi t/T)\ . \quad (6)$$

The interface evolves dynamically from an initial sphere of radius 0.15 centered at (0.35, 0.35, 0.35) to $t = 1.5$. The velocity field will then reverse its direction. At $t = T = 3.0$, the interface comes back to its initial state. The error comparison with the two PLIC methods in [6] is given in Table 1, and shows superior performance for LGB Front Tracking. More benchmark tests are documented in [1].

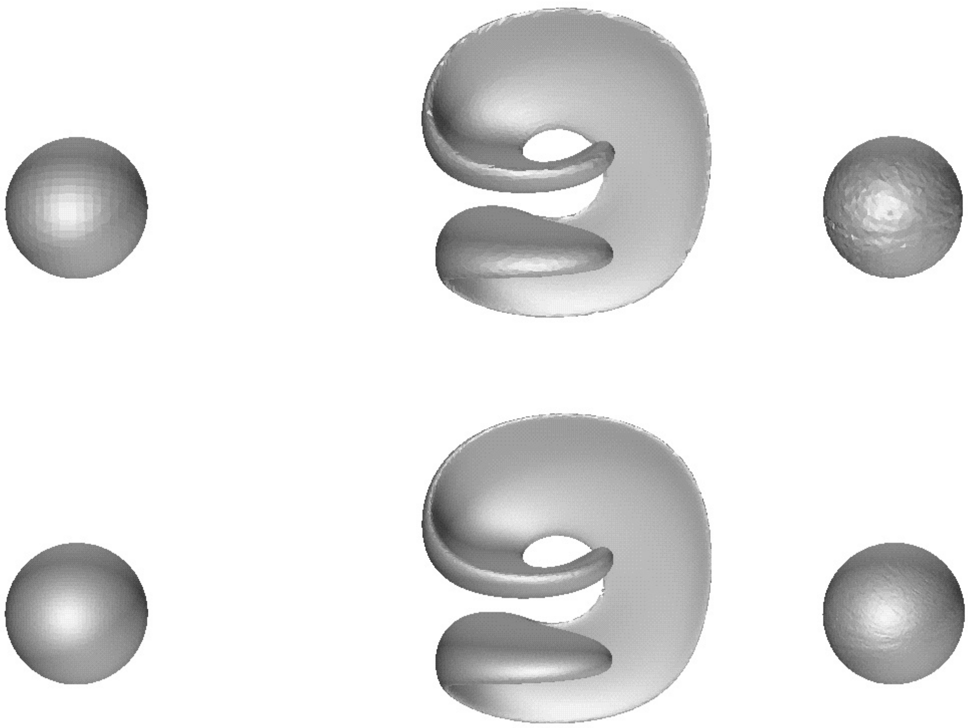

Figure 2. *Reversal test of a 3D interface in deformation velocity field with* $CFL = 0.5$*. The sequence above has the mesh of* 64^3*, and the sequence below has the mesh of* 128^3*. From left to right are* $t = 0,\ 1.5,\ 3$ *respectively.*

Table 1. L_1 norms for the error at $t = 3$ for the LGB method in the three dimensional deformation simulation compared to the two interface methods used in [6] with $CFL = 0.5$.

Mesh	LGB	Order	CVTNA	Youngs
32^3	5.72×10^{-3}	3.72	7.41×10^{-3}	7.71×10^{-3}
64^3	4.33×10^{-4}	1.82	1.99×10^{-3}	2.78×10^{-3}
128^3	1.23×10^{-4}	N/A	3.09×10^{-4}	7.58×10^{-4}

5. Scientific Applications of FronTier

The FronTier code has been used in several SciDAC applications including the simulation of diesel fuel injection jet, the study of Rayleigh-Taylor mixing and the MHD simulation of the fusion pellet injection.

5.1. Diesel Jet Injection

In this study [5], the vaporization of the fuel is simulated by the dynamic creation of the vapor bubbles in the fuel. The diesel fuel is treated as a viscous fluid. The thermal conductivity of the fuel is also considered. Because the thermal conductivity and the viscosity are both small, we are able to solve the Navier-Stokes equations with an explicit algorithm. On the liquid/vapor interface boundary, we solve a phase transition problem.

The discrete vapor bubble model predicted cavitating regions made of many microbubbles that led to jet breakup. It displays the influence of the disturbances brought by the finite size cavitation bubbles that causes atomization. This is the first direct numerical simulation of the impact of cavitation bubbles on atomization.

5.2. Fluid Mixing

Acceleration driven mixing has been the subject of intense investigation over the past 50 years. Idealized cases of steady acceleration (Rayleigh-Taylor or RT) and impulsive (Richtmyer-Meshkov or RM) mixing have been studied by theory, experiment and numerical simulations. Recently, we have identified surface tension as a significant contributor to the mixing rate. Using the FronTier code with the newly renovated tracking algorithms, We found [2] that the mixing rate α in the equation

$$h = \alpha A g t^2 \tag{7}$$

to be 0.062, within 5% of the experimental values $\alpha = 0.066$ [8].

5.3. MHD Simulation of Fusion Pellet Injection

Front tracking technologies have been used in the MHD code for free surface flows of conducting liquids and weakly ionized plasmas in the presence of phase transitions. The code is being developed at the Brookhaven National Laboratory and applied for the simulation of MHD processes in challenging DOE applications such as tokamak refueling technologies and liquid mercury targets for future advanced accelerators.

6. References

[1] J. Du, B. Fix, J. Glimm, X. Li, Y. Li, and L. Wu, *A simple package for front tracking*, J. Comp. Phys., (2005). Submitted. Stony Brook University preprint SUNYSB-AMS-05-02.

[2] E. George, J. Glimm, X. L. Li, Y. H. Li, and X. F. Liu, *The influence of scale breaking phenomena on turbulent mixing rates*, Phys. Rev. Lett., (2005). In press. Stony Brook University Preprint number SUNYSB-AMS-05-11.

[3] J. Glimm, J. W. Grove, X.-L. Li, K.-M. Shyue, Q. Zhang, and Y. Zeng, *Three dimensional front tracking*, SIAM J. Sci. Comp., 19 (1998), pp. 703–727.

[4] J. Glimm, J. W. Grove, X.-L. Li, and D. C. Tan, *Robust computational algorithms for dynamic interface tracking in three dimensions*, SIAM J. Sci. Comp., 21 (2000), pp. 2240–2256.

[5] J. Glimm, M.-N. Kim, X.-L. Li, R. Samulyak, and Z.-L. Xu, *Jet simulation in a diesel engine*, Elsevier Science, 2004. Accepted for publication in Proceedings of the third MIT Conference on Computational Fluid and Solid Mechanics.

[6] P. Liovic, M. Rudman, J.-L. Liow, D. Lakehal, and D. Kothe, *A 3d unsplit-advection volume tracking algorithm with planarity-preserving interface reconstruction*, Computers and Fluids, (2005). Submitted.

[7] W. E. Lorensen and H. E. Cline, *Marching cubes: A high resolution 3D surface construction algorithm*, Computer Graphics, 21 (1987), pp. 163–169.

[8] K. I. Read, *Experimental investigation of turbulent mixing by Rayleigh-Taylor instability*, Physica D, 12 (1984), pp. 45–58.