Implementation of Level Set Methods in Cactus Framework

By Carbunescu Razvan Corneliu
Introduction

- Computational Fluid Dynamics is used as a valuable engineering simulation tool to understand and design complex systems involving fluid flow, heat transfer, mixing and multiphase flows.

- A simple example of multiphase fluid flow would be a drop of water falling into a glass.

- These problems involve complex boundary and interface conditions evolving in time and therefore level set methods can be a powerful tool to handle such problems computationally.
Level Set Methods

- are a general and powerful technique to represent an object's boundary by the means of an implicit function that has a specific value on the boundary

- allows for the easy modeling of interfaces between different fluids that interact by the use of partial differential equations to represent the interactions

- are used extensively in Computational Fluid Dynamics (CFD) because of these abilities
Cactus Framework

- “is a framework for developing portable, modular applications, in particular, although not exclusively, high-performance simulation code”

- provides great flexibility for the written code allowing for it to be easily modified to meet new needs

- is an open source code so also allows for further development by any other interested parties
Zalesak’s problem

- problem discusses the movement of a notched disc that is rotating in a given velocity field
- Zalesak’s problem has often been used in papers to test the implementation of level set methods
- the disc should maintain constant volume after one full rotation with the use of level set methods although the shape is not preserved with simple level set methods
Implicit functions

- a function which has a value of 0 on the boundary or interface of the object, negative inside, and positive outside

- these functions are easily manipulated by the use of partial differential equations

- the best choice for level set methods is the Signed Distance Function (SDF)
Signed Distance Function

- the value of the function is the distance from the interface (d) with a negative or positive sign assigned depending on it’s location inside or outside the object

\[
SDF(x) = \begin{cases} 
-d & \text{if } x \text{ is inside the object} \\
0 & \text{if } x \text{ is on the interface} \\
+d & \text{if } x \text{ is outside the object}
\end{cases}
\]
Equation of movement

• the evolution of an object represented by an implicit function in an external velocity field is given by the following partial differential equation:

\[
\frac{\partial \Phi}{\partial t} + \nabla \cdot \vec{V} = 0
\]

Where:

\[
\vec{V} = (u, v, w)
\]

and

\[
\nabla \Phi = \left( \frac{\partial \Phi}{\partial x}, \frac{\partial \Phi}{\partial y}, \frac{\partial \Phi}{\partial z} \right)
\]
Discretization

- partial derivatives are represented by discretized approximations
- this schemes used are upwind differencing and are first order accurate

\[
\frac{\partial \Phi}{\partial t} \approx \frac{\Phi_{i,i,k}^{n+1} - \Phi_{i,i,k}^n}{\Delta t} \\
\frac{\partial \Phi}{\partial x} \approx \frac{\Phi_{i,i,k}^n - \Phi_{i-1,i,k}^n}{\Delta x} \\
\frac{\partial \Phi}{\partial y} \approx \frac{\Phi_{i,i,k}^n - \Phi_{i,i-1,k}^n}{\Delta y} \\
\frac{\partial \Phi}{\partial z} \approx \frac{\Phi_{i,i,k}^n - \Phi_{i,i,k-1}^n}{\Delta z}
\]
Stability concerns

- the schemes are not stable for all timesteps and velocities
- a scheme’s stability can be determined by analysis of a factor known as the Courant number
Reinitialization importance

- the equation presented before changes the nature of the function after a number of time steps
- constant reinitialization of the function to the signed distance function is needed to preserve it’s qualities
- the reinitialization technique chosen for the example is a slight variant to the Fast Marching Method algorithm
- the FMM algorithm first adds the values closest to the interface to a list and then continues to expand that band
Reinitialization algorithm

- uses only the neighbors of any point and their sources on the interface to determine the value at that point

where:

- points on the grid
- points near the interface
- currently selected point in the list
- currently selected neighbor
- other points around the neighbor
- points at which the neighbor looks for determining the S.D.F.
Reinitialization results

- The following results are on a grid size of 100x100 and with a timestep of \( dt = 0.0005 \)
Current status and future work

- The implementation has been partly moved to parallel but there are still a few problems to work out with the parallel reinitialization algorithm.
- The next move will be to move the thorns in Cactus from 2D to 3D and implement higher order accurate schemes (2\textsuperscript{nd} order has been implemented but still needs testing).
- After that will be accomplished more advanced problems like multiphase flow and curvature driven flows will be implemented.
Acknowledgements

• The Center for Computation and Technology at LSU and their excellent technical, academic and organizational personnel which helped me with the research.

• I would especially like to thank Dr. Gabrielle Allen, Mr. Yaakoub el Khamra, Dr. Mayank Tyagi and Ms. Kathy Traxler

• The Department of Computer Science, the ACM and the Student Government organizations at LSU for helping fund the presentation.
Thank you!