Eurographics/ACM SIGGRAPH Symposium on Computer Animation (2005)
K. Anjyo, P. Faloutsos (Editors)

Oriented Particle Level Set for Fluid Simulation

Frederic Pighin Sanjit Patel Jonathan Cohen[†] Anson Chu

University of Southern California, Institute for Creative Technologies
[†]Rhythm and Hues Studios

Abstract

The particle level set technique has been adopted in computer graphics as the method of choice for tracking the surface of simulated liquids. In this poster, we describe a novel technique for modeling such an interface. Our technique is based on a set of oriented particles that provides a piecewise linear approximation to the interface. Using this improved model, we obtain a more accurate representation of the water surface and reduced mass loss during simulation.

1. Introduction

For liquid simulations the interface between the liquid and the air needs to be tracked accurately. Level set methods [OF03] are widely used to track the motion of material interfaces. This method is particularly effective when the interface is stretched, teared, or undergoes changes in topology. However, one of the main limitations of using a fixed grid is that the motion of the interface cannot be tracked at a finer granularity than the grid cells. In particular, the level set technique can wrongfully merge characteristics within a grid cell.

The particle level set technique from Enright et al. [ELF05] attempts to address this issue by advecting massless particles into the flow. These signed particles are seeded in the proximity of the interface and advected passively. Since they sample space at a sub-cell granularity, the particles can be used to correct the level set at each timestep. As a result, the mass of the simulated fluids is better preserved, and the animated fluids look more believable. Because it is very effective at producing realistic fluids, it has been adopted as the method of choice for simulating liquids by a large part of the graphics community. To evaluate the error at a grid cell, the algorithm considers the level set defined by nearby particles. Each particle approximates the level set locally using a spherical model (as illustrated in the left part of figure 1). This sphere is centered at the particle, and its radius relates to the sampled distance field at the location of the particle at the previous timestep. A limitation of this technique is that it only tracks the characteristic curve at the center of the sphere, yet uses the entire sphere to correct the levelset. This assumes that within the sphere, all characterstic curves are parallel to each other. This assumption becomes increasingly incorrect as the granularity of the deformation in a flow become smaller than the size of the sphere.

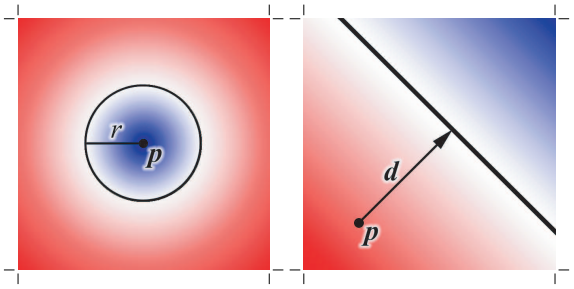

Figure 1: *Level set defined by a particle. Left: the model proposed by Enright et al. [ELF05] models the interface as a sphere centered at the particle. Right: the oriented particle models the interface locally as a plane.*

Our goal is to provide a more accurate model for the interface by using *oriented particles*. Instead of providing a spherical approximation, an oriented particle provides a linear approximation where the particle locally models the level set as the distance to a plane (as illustrated in the right part of figure 1). With this model the set of particles provides a piecewise linear approximation of the interface. This simple modification has resounding consequences on the quality of the correction provided by the particles.

2. Oriented Particle Level Set

Error quantification. We define an oriented particle by two quantities: the location of the particle $\boldsymbol{p}$ and its orientation $\boldsymbol{d}$. The vector $\boldsymbol{d}$ is constructed so that it points in the direction of the interface and its magnitude is equal to the magnitude of the level set at the particle, so that:

$$\boldsymbol{d} = -\phi(\boldsymbol{p})\frac{\nabla\phi}{|\nabla\phi|}(\boldsymbol{p}).$$

The local approximation, ϕ_p, provided by an oriented particle is the signed distance to the plane passing through the point $\boldsymbol{p}+\boldsymbol{d}$ and perpendicular to $\boldsymbol{d}$. This function can be computed using:

$$\phi_p(\boldsymbol{x}) = s_p(\frac{\boldsymbol{d}}{|\boldsymbol{d}|}\cdot(\boldsymbol{x}-\boldsymbol{p})+|\boldsymbol{d}|),$$

where s_p changes the sign of the distance based on the sign of the particle. To speed up the evaluation of this function, the norm of $\boldsymbol{d}$ and its direction can be precomputed. Using this implementation, an evaluation costs only a dot product and an addition. The vectors $\boldsymbol{d}$ are reset from the level set after it has been corrected.

The advection of an oriented particle requires the advection of two points: $\boldsymbol{p}$ and $\boldsymbol{p}+\boldsymbol{d}$. We use a Semi-Lagrangian scheme similar to [ELF05]. $\boldsymbol{d}$ is recomputed from these two points after advection. By advecting $\boldsymbol{d}$, we can model the effect of the flow around the particle not only as a translation but we can also take rotation and scaling into account.

Error correction. Our correction algorithm was designed to address two issues. First, it is a local algorithm that corrects values independently at each voxel. The particles could be used to perform global surface reconstruction of the interface but it would be too costly for our purpose. Second, for a given voxel, we would like to sample nearby characteristics so that closer particles contribute comparatively more. With these in mind, we chose to compute a scaled average of the contribution of all the particles within the voxel. In other words, at a point $\boldsymbol{x}$, we derive the corrected values of the level set, $\phi_{cor}(\boldsymbol{x})$, from the set of escaped particles within the voxel, E, and the value of the level set computed by the advection procedure, $\phi_{adv}(\boldsymbol{x})$, using:

$$\phi_{cor}(\boldsymbol{x}) = \frac{\Sigma_{p\in E} m(|\boldsymbol{x}-\boldsymbol{p}|)\phi_p(\boldsymbol{x}) + \alpha\phi_{adv}(\boldsymbol{x})}{\Sigma_{p\in E} m(|\boldsymbol{x}-\boldsymbol{p}|)+\alpha},$$

where m is a non-decreasing scalar function, and α is a positive constant. m is used to tune the influence of a particle with respect to its distance to the point $\boldsymbol{x}$ and α tells us how much we keep from the advection step. In our experiments, we use the function $m(a) = a/3$ and $\alpha = 1$.

A particle is defined as escaped if it is on the wrong side of the interface regardless of its distance to the interface.

3. Results

We compare the performance of the oriented particle level set technique with the original technique as described in [ELF05]. We perform these comparisons on several variations of the Zalesak sphere experiment [Zal79]. Our experiments are three-dimensional, as a result we use Zalesak spheres as described in [ELF05]. All our experiments involve one full rotation of the sphere. They were conducted on a 2.4GHz Windows NT workstation with 2Gb of memory. Our default experiment uses a $30\times30\times30$ grid, with 64 particles per voxel, and a timestep equal to half the bound given by the CFL condition. We conducted three sets of experiments that introduce variations in grid size, particle density, and timestep.

	Spherical			Oriented		
	Time	Mass loss	Area loss	Time	Mass loss	Area loss
Grid size						
50	4	11	10	6.2	2	2
30	1	19	14	2	2	3
20	3.45	34	25	1	9	4
Particle density						
64	1	19	14	2	2	3
32	1.8	23	16	2.26	4	6
16	1.3	28	19	1.6	11	11
8	1	30	21	1.1	15	12
0	.51	52	42	.51	52	42
Timestep						
.5	1	19	14	2	2	3
1	.48	23	19	1.9	3	4
2	.25	19	20	1	1	2
3	.38	27	34	.68	1	3

Table 1: *Performance comparisons. Timings are given in minutes. The mass and area losses are given in percentage of the initial values ($t = 0$) and the timesteps are expressed as fractions of the CFL bound.*

References

[ELF05] ENRIGHT D., LOSASSO F., FEDKIW R.: A fast and accurate semi-lagrangian particle level set method. *Computers and Structures*, 1 (2005), 479–490.

[OF03] OSHER S., FEDKIW R. P.: *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003.

[Zal79] ZALESAK S.: Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics 31* (1979), 335–362.