EUROGRAPHICS 2007 / D. Cohen-Or and P. Slavík
(Guest Editors)

Volume 26 (2007), Number 3

Textured Liquids based on the Marker Level Set

1067

Abstract

In this work we propose a new Eulerian method for handling dynamics of liquids and their surface textures. Our approach is based on a new method for interface advection we term the Marker Level Set (MLS). The MLS method uses surface markers and a level set for tracking the surface of the liquid, achieving more efficient and accurate results than popular methods like the Particle Level Set method (PLS). Another novelty is that the surface markers allow the MLS to handle surface texture advection as well, a very rare capability in the world of Eulerian simulation of liquids. We present several simulations of liquids and their texture maps evolving dynamically.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics andRealismAnimation

1. Introduction

In nature one encounters countless examples in which a liquid flows and carries along various textures on its surface: foam being tossed and turned by the might of an overturning wave, a patch of oil spill or various floating particles on the ocean surface, milk poured into coffee, etc. This brings forth the idea that the simulation of liquids in general must be complemented by an associated texture advection methodology, mirroring the real life. Graphics literature is scarce with such methods that could be applied for the specific situation of liquids carrying surface textures. It is an understatement to say that a robust method supplying such capabilities of advecting textures along with a realistic simulation of the fluid physics would be a welcome addition to the methods for simulation of fluids. This paper supplies such a method that attempts to fill the empty space in this area of graphics research. Its use may go easily beyond realistic simulations of the type mentioned above (foam on waves, oil patches on water, milk inside coffee) to artistic simulations like having the Wicked Witch or another character melting, with its surface textures following the velocity in a natural manner.

In our approach we regard the surface and its texture to be an integrant part of the liquid, therefore moving with the same velocity as the liquid. Simply trying to evolve dynamically in texture space is unfeasible in the general case we consider, due to possible topological changes. The same "topological" reason suggests the use of a level set based approach. Another issue is related to the finiteness of computational resources. In the specific case of an Eulerian fluid simulation the need for high accuracy given finite resources is tackled with the use of adaptive methods, like the octree approach of [LGF04]. Another helpful ingredient is constituted of elements that offer sub-pixel accuracy, like the particles from the particle level set method (PLS) of [EMF02]. Our method combines two essential basic ingredients that offer also the solution to the necessities outlined above. These two ingredients are a *level set* and a *set of markers* coupled to the zero level set, such that they lie essentially on the liquid surface. This enables the use of the markers as texture carriers, complementing the capability of the level set to model painlessly topological changes. The level set in itself cannot be used for texture advection because the zero level set only listens to the normal component to the surface of the velocity. We term our new approach the Marker Level Set method (MLS) and we will discuss it at length in this paper. We will show that the MLS method can be regarded as a more efficient but equally accurate alternative to the PLS method, while having the natural extra capability of carrying surface textures. In their 2004 Siggraph presentation Digital Domain programmers reported that, for simulating the giant wave traveling on the streets of New York City in the movie Day After Tomorrow, they needed to implement surface particles that carried the foam texture, and had to constantly reinitialize their surface positions. MLS takes care automatically of such issues. MLS would also be the natural solution to choose for animations like the one of the Terminator body melting shown in Terminator 3. That scene was animated with the use of the PLS, as reported in [REN*04]. Using the MLS method for animation of that type would benefit both from its texture advection capability as well as from its effi-

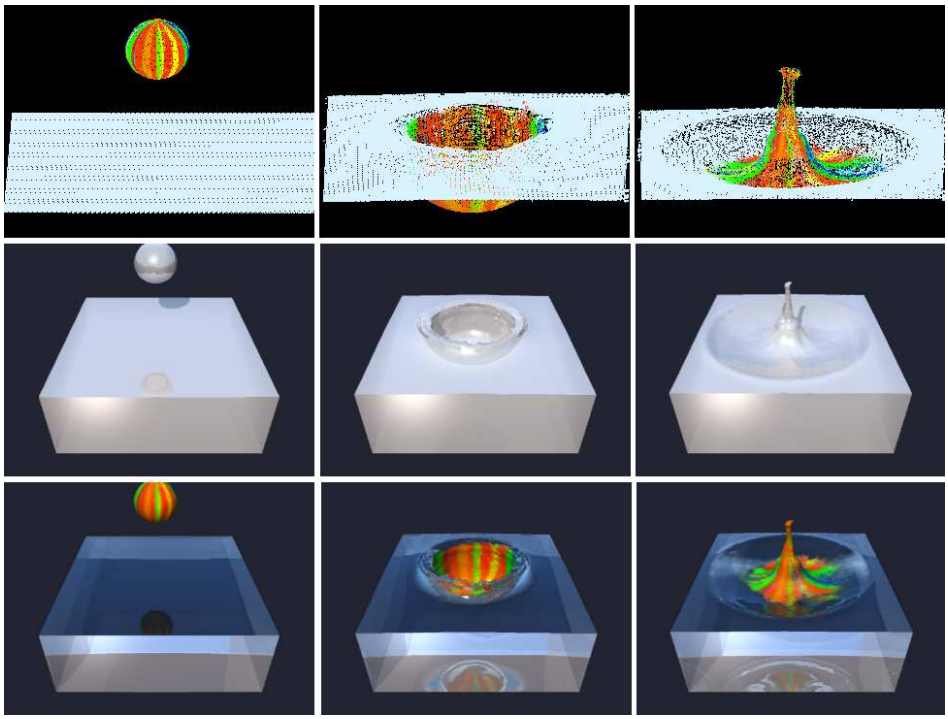

Figure 1: *The MLS method provides a dual description to a surface: surface markers coupled with a level set. From left to right, several stages in a falling water droplet simulation, at times 0.23, 0.39, 0.51. From top to bottom: the colored markers, the zero level set rendered with a "milk" shader, the zero level set rendered based on marker color and transparency.*

ciency over the PLS, due to the fact that MLS uses markers only *on* the surface, whereas the PLS places them in a tubular neighborhood of the surface.

In summary, the main contribution of this paper is the introduction of the Marker Level Set to graphics and showing its use as both a liquid simulation tool on one hand and as a texture advection tool on the other. Its accuracy and versatility recommend it as a strong new advection method with use in graphics. In the following we present first an overview of related previous work and we follow with a detailed description of the MLS and its use. Finally we present a series of animations with and without textured surfaces.

2. Previous work

We discuss here previous work in fluids simulation with special emphasis on interface tracking and texture advection. We are interested mainly in physics-based approaches, and therefore we look at true three dimensional techniques. [FM96] are the initiators of fluid simulation based on solving the 3D Navier-Stokes, and their paper has been followed by many excellent pieces of work. A necessarily incomplete list would include [Sta99], who introduced the stable semi-Lagrangian advection methods, [FF01] who used particles in an Eulerian setting to define the body of liquid, [EMF02] who introduced the particle level set to graphics, [HK05] who proposed a sharp method for treating discontinuous variables and obtained beautiful two-phase flow animations of bubbles, [MUM*06] who introduced a nice method for simulating boiling flows and [LSSF06] who showed impressive multiphase simulations. The PLS came into play specifically as an enhancer of the surface tracker used in [FF01], first in terms of accuracy, but also in efficiency, due to the particles being seeded only in a tubular region about the interface, rather than inside the whole body of liquid. In turn, our method also provides an efficiency enhancement over the PLS this time, due to particles being seeded only along the interface. Similarly with

the PLS on the other hand, our markers provide subpixel information that the Eulerian grid loses due to discretization errors. Such information is not directly available in methods like the one proposed by [BGOS06], where Lagrangian information is given by the surface triangles obtained by isocontouring the (Eulerian) level set. Our method differs also from the various particle methods used in graphics, for example [MSKG05, PTB*03, KAG*05]. These methods are customarily variants of the smoothed particle hydrodynamics method of [Mon92], and as such they place particles throughout the body of the liquid - MLS is more efficient in this respect, placing particles only along the interface. Also, their surface reconstruction uses often an implicit method that is employed at every time step to recover a smooth surface from the particles. Because there is no dynamic link between the implicit surface and the particles (in MLS this link exists) those methods are prone to lack of temporal coherence. Regarding texture advection, there are several methods that concentrate on advecting the texture through the flow field, e.g. [Wit99, Sta99, Ney03], that work well for general fluid simulations, but not for our specific case of liquid surface (rather than volume) texture advection. [Sta03] showed beautiful fluid texture advection on arbitrarily surfaces of fixed shapes, while MLS can handle any shape the liquid surface dynamically evolves into. To address the particular context of liquids, [REN*04] proposed a method that advects texture particles, initialized near the interface, through the fluid flow field. Rendering is performed by interpolating texture coordinates from the nearest 64 particles to the point where a traced ray intersects the surface. Our rendering approach is very similar to theirs, even though we will hint at possible improvements in the conclusion section. Their method, while closest to MLS among all the current advection methods, differs from MLS in an essential way, in that the markers they use are "passive" color carriers, whereas our method recongizes their essential capacity of carrying both texture and motion characteristic information. We also provide a full mechanism for deletion and addition of markers, including local color inheritance. [WH04] and [HNB*06] stored three-dimensional texture coordinates in a grid structure and advected them as scalar fields. To avoid artifacts resulting from volumetric advection the authors used extrapolation techniques to force the gradient of the texture field to be perpendicular to the interface normal. The method of [BGOS06] allows for advection of texture coordinates (or other surface properties) on the actual surface, even though no mechanism for texture generation in the case of topological break-ups was offered. We note that the MLS has built-in a marker addition routine in such cases, and along with it local interpolation of texture from neighboring markers.

3. The MLS simulation framework

Our simulation approach includes the usual two ingredients for simulating liquid flows, namely a surface tracker - the MLS - coupled with a Navier-Stokes solver. MLS is also the essential component for texture advection. Let us look at them in turn.

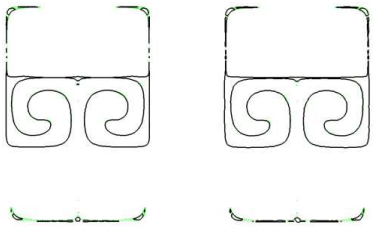

Figure 2: *Vortex deformation field simulations using MLS at t=1. No particle redistribution on the left, with particle addition and deletion on the right. The surface markers are visible in light green behind the black reconstruction of the level set. Grid size:* 100×100.

3.1. Overview of the MLS

The Marker Level Set method combines the advantages of a level set framework, for example the ability to recover a *smooth* interface separating two regions of the domain and the simplicity of dealing with topological changes, with the advantages of particle tracking, for example the lack of diffusion encountered by Eulerian methods. The main idea of the MLS is to place markers on the surface and have them define the interface between the liquid and gas at any moment in time (note that this is much like the original level set idea to have the zero contour represent the interface at any moment in time). The markers suffer almost no numerical diffusion upon (second order accurate) advection, therefore they provide excellent information about the motion characteristics and are used to update the level set such that the level set becomes zero on the markers. The level set, on the other hand, can handle painlessly topological changes and it provides information about markers that may need to be deleted or added. This philosophy bears a strong resemblance to the PLS method, however the placement of markers only on the interface makes the method more efficient. In [Ano07] we show that the MLS supersedes the PLS in accuracy in most of the standard 2D tests. In Figure 2 we show the result of one of the "hardest" of the 2D deformation tests, with better area conservation results than the PLS. The MLS also has very good numerical properties concerning advection of sharp corners, as we can see in the Zalesak test results from Figure 3. In Figure 4 we also see how MLS handles a 3D periodic deformation test (the Enright test with period 2.5). This test also showcases the texture advection capabilities of the method. The initial texture returns after the deformation to its exact initial position.

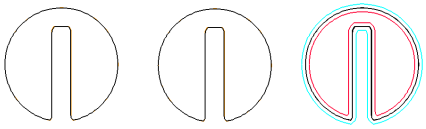

Figure 3: *Zalesak's problem using MLS. The MLS solution is in black, theory in orange. From left to right: after one rotation, after two rotations, after one rotation showing also the adjacent level sets. The resolution is* 100×100.

3.2. Setup and general algorithm

Our general setup consists of an interface between two regions of our computational domain Ω and a smooth velocity vector field $\mathbf{u}$ defined everywhere in the domain (or at least in a tubular neighborhood of the surface). The interface (a closed smooth surface, in three dimensions) is advected with the velocity $\mathbf{u}$. We provide a dual description to the interface, as the zero interface of a level set ϕ (the Eulerian part) and a set of points lying on the surface itself (the Lagrangian part). For numerical purposes the level set is discretized on a regular Eulerian grid while the set of points is discretized as a finite set of markers. The dual descriptions are coupled in order to provide an accurate solution to the advection problem.

After initialization, the MLS algorithm follows the following algorithm, at every time step:

1. level set advection
2. marker advection
3. marker deletion
4. level set correction
5. marker addition

Each marker carries color information that does not change during simulation. Markers added in step 5 are assigned color based on kernel interpolated local color. Let us discuss each of the steps in turn.

3.2.1. Initialization

The initial level set is defined procedurally in this work. A linear reconstruction of the zero level set is obtained with marching cubes and along each of the surface triangles we place a number of markers, proportionally with the area of the triangle. For very small triangles we only place three markers, at the vertices. For robustness we limit throughout the simulation the maximum number of markers present in a cell. The higher the maximum number the better is the local marker distribution over time and the more accurate the interface representation is. Moreover, a good marker distribution impacts positively the surface color reconstruction, because there is less of a need to add new markers in under-resolved regions of the zero level set. For all the simulations in this paper the maximum number is 48, which was chosen based on experience, to provide a good balance between memory use and accuracy, for the specific grid sizes we used. For more resolved grids this number can be actually reduced while if a lot of RAM memory is available it could be increased (we tried to fit within 1GB of RAM memory for the MLS and the fluid solver). The marker color is initialized also procedurally, or semi-procedurally, based on a bitmap image.

3.2.2. Level set advection

The level set is advected by discretizing the level set equation $\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0$. The order of accuracy of the discretization matters little, as long as the CFL number is kept below one, as also noted by [EMF02]. We used either semilagrangian advection or BFECC [KLLR06] with similar results. This is due to the corrective effect of the markers, whose advection accuracy is the one that really matters. We also perform periodic reinitialization of the level set [SSO94]. To that end, we solve the reinitialization equation $\frac{\partial \phi}{\partial t} = sgn(\phi^0)(1 - |\nabla \phi|)$ using an implementation of the reinitialization procedure proposed by [RS00].

3.2.3. Marker advection and deletion

The markers placed along the zero level set are used for several purposes: 1) to track characteristic information, 2) to help reconstruct the interface in regions where the level set method has failed to accurately preserve mass, and 3) to provide tangential dynamics information that the level set method discards. After the marker initialization process, we advect the particles using, at each time step, the evolution equation

$$\frac{dx}{dt} = \mathbf{u}(x^p) \tag{1}$$

where x^p is the position of the particle and $\mathbf{u}$ is its velocity. Particle velocities are interpolated from the velocities on the underlying grid using tricubic interpolation. Similarly to the proponents of the PLS, we found that we need the numerical solution of the above equation to be second-order accurate for good results, and we used the second-order Runge-Kutta in order to accomplish this. As already mentioned, the total number of markers that lie inside a cell is limited. If more markers than the limit enter a full cell we delete the new comers. When dealing with stretching/compressing velocity fields, markers may clump together; limiting the maximum number of markers per interface cell helps keep the markers relatively well distributed. We also employ a special deletion step (step 3 in the main MLS algorithm) in which we remove

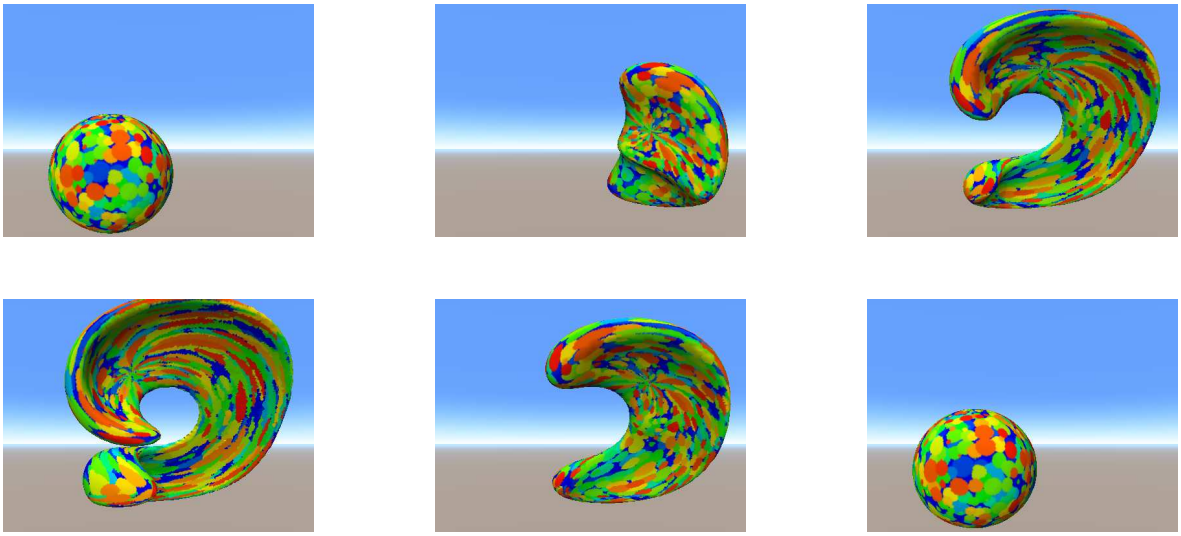

Figure 4: *Color Enright test with period T = 2.5. First row times: 0, 0.16T, 0.32T. Second row times: 0.5T, 0.75T, T.*

the markers that are farther than a certain threshold from the zero level set. This ensures for example that upon reconnection there are no markers "stuck" inside the liquid, and upon break-up there are no markers left floating around.

3.2.4. Level set correction

The MLS computes correction terms for the level set values at each of the nodes located in a tubular neighborhood of the interface. Namely, in two dimensions for example, for each such node (i, j) we compute the new value of the level set as $\phi_{i,j}^{new} = \phi_{i,j} - \lambda_{i,j}$, with $\lambda_{i,j}$ local corrections of the level set. These local corrections are based on the positions of the closest local markers and are computed as an average of the current level set values at these markers:

$$\lambda_{i,j} = \sum_k w_k \phi(x_k) / \sum_k w_k \tag{2}$$

where x_k are the positions of the markers from a small neighborhood of (i, j) and w_k are weights associated to these markers and the node (i, j). $\phi(x_k)$ is the interpolated value of the level set function at the marker location. By performing the local correction of the level set we effectively force its zero level to "align" with the markers. Thus, our correction procedure is the following: for each node (i, j) close enough to the set of markers we compute the updated level set value $\phi_{i,j}^{new} = \phi_{i,j} - \sum_k w_k \phi(x_k) / \sum_k w_k$. We use Gaussian weights of the form:

$$w(q) = \begin{cases} \frac{e^{-(q/c)^2} - e^{-1/c^2}}{1 - e^{-1/c^2}}, & 0 \le q \le 1; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

where $q = q_{i,j}(x) = dist(x, (i, j))/\rho$, $c \in [0, 1]$ is a constant and ρ is the kernel radius.

3.2.5. Marker addition

The addition of markers is done in a manner similar to the initialization of the markers. If we find a sign variation for the level set inside a cell we add markers along the linear surface reconstruction inside that cell. The markers are added in a regular disposition along each surface triangle with a large enough area inside that cell. The color of the added markers is defined with the help of the same type of kernel-based interpolation outlined in equations (2) and (3), used for each color channel rather than for the level set function.

3.3. The fluid simulator

The fluid simulator uses the technology proposed by [Sus03] and introduced to graphics by [MMS04], but instead of CLSVOF (the coupled level set and volume of fluid method) we use MLS for interface advection. The MLS method does not have a built-in mass conservation logic, as CLSVOF does, but its excellent characteristic tracking properties ensure that its mass preservation is also very good. For example, in all the standard 2D tests for surface trackers MLS preserved mass to a fraction of a percent, while in the physically realistic 3D simulations presented here the mass loss was less than three percent for averagely sized computations (e.g. $64 \times 64 \times 64$ grids), and one can only expect the mass conservation to improve even more with more resolved grids. Moreover, due to the same motion characteristic tracking built into MLS, its preservation of high-gradient

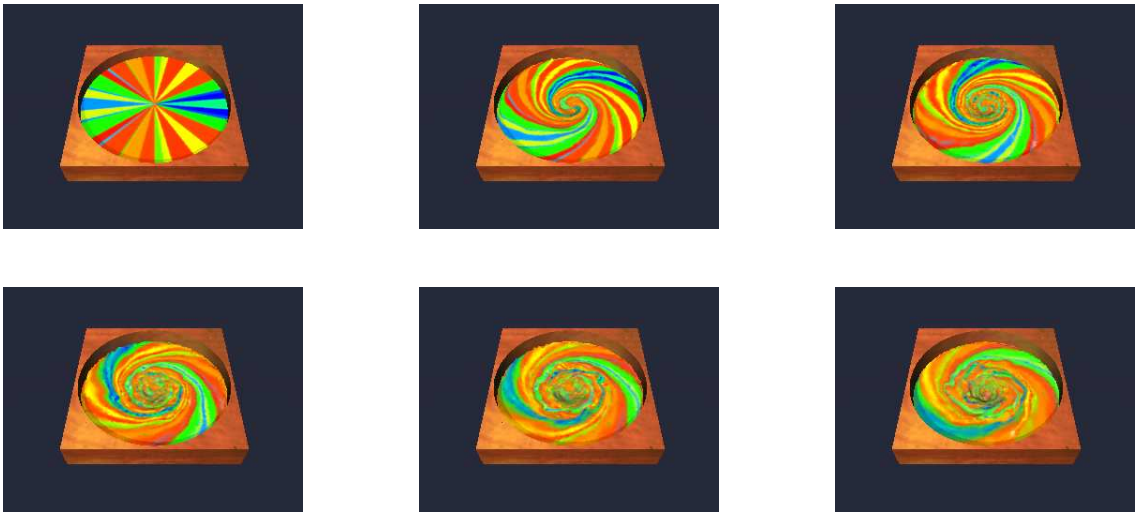

Figure 5: *Swirly water simulation. From left to right and top to bottom time t = 0,1,2,3,4,5.*

or high-curvature quantities (corners, for example) is better than CLSVOF's for a given resolution.

The fluid simulator is given by the Navier-Stokes simulator coupled with the MLS. It consists of solving, at every time step:

- balance of momentum equations
- balance of mass equation
- MLS equations

We have discussed already the MLS equations. For the rest of the solver we used the inviscid incompressible form of the Navier-Stokes equations:

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla p + \rho \mathbf{g} \quad (4)$$

$$\nabla \cdot \mathbf{u} = 0 \quad (5)$$

Here $\mathbf{u}$ denotes the velocity, p the pressure, ρ is the fluid density and $\mathbf{g}$ is the vector gravity. $\frac{D}{Dt}$ is the material (advective) derivative:

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{u} \cdot \nabla \quad (6)$$

A second order upwind finite difference scheme is used to discretize the nonlinear advective terms. The vapor pressure is considered constant (one-phase formulation). The density ρ is controlled at any time step by the level set:

$$\rho = \begin{cases} \rho_{liquid}, & \phi \geq 0; \\ 0, & \phi < 0. \end{cases} \quad (7)$$

The Navier-Stokes equations are solved using the projection approach of [BCG89].

4. Simulation results

We ran several 3D simulations to test the power of our new method. The results are very encouraging judging from their accuracy and the display of the color advection capabilities of the MLS.

4.1. The color deformation test

We used the Enright test [EMF02] as a basis of both our color deformation and 3D interface stretching tests. The grid size is $100 \times 100 \times 100$ and the stretching period is 2.5. The initial sphere follows the deformation field up to its maximum deformation instant, after which the velocity is reversed. Both the shape of the sphere and the colors were stretched smoothly and afterwards returned to their original state with insignificant diffusion. Our results are similar in this respect to the ones obtained by [BGOS06].

4.2. The swirl

In this simulation (Figure 5) we set up a small-depth body of liquid with a radial stripe texture on the surface and a radially-decreasing rotational velocity. This setup forces the liquid to twist and produces nice color mixing on the surface. Due to the higher center velocity and gravitational pull, the surface suffers a central depression. This forms a wavefront that is bounced back and forth to the circular walls several times, while the surface color continues to mix. The grid size was again fairly coarse, $64 \times 64 \times 16$. The presence of the subpixel color information stored by the markers is essential once again for providing robust local color information, even in the presence of such a strongly distorting flow.

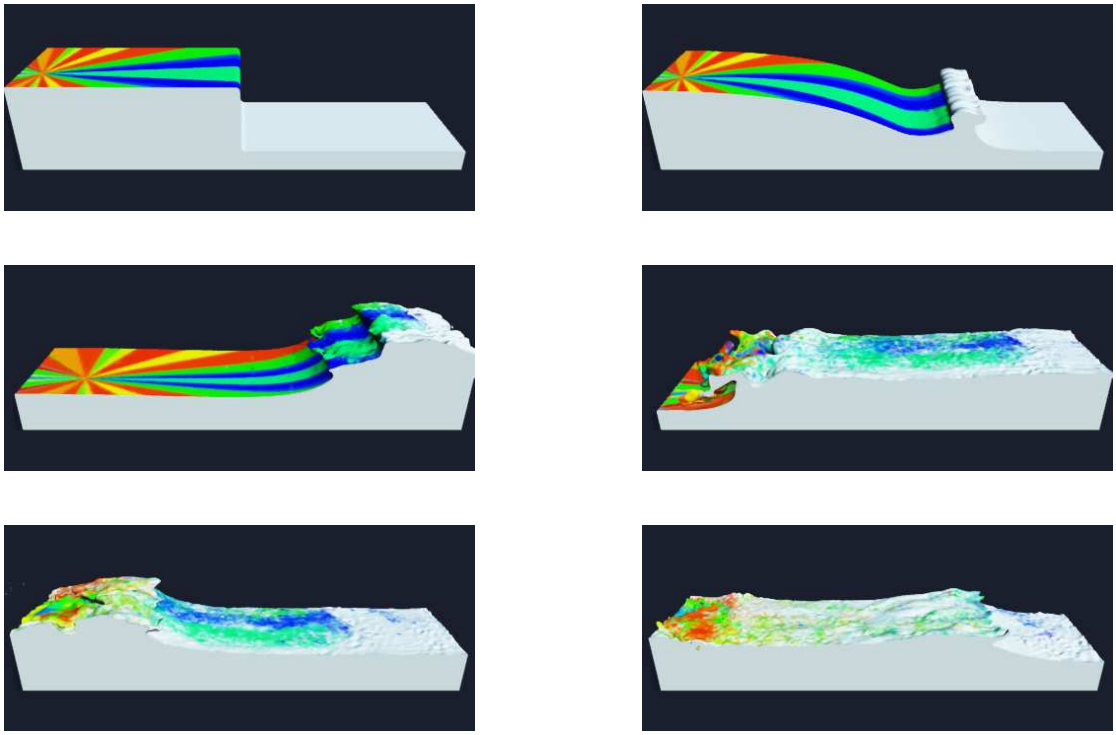

Figure 6: *Dam breaking simulation with matte rendering. From left to right and top to bottom time t = 0,1,2,3,4,5 seconds.*

4.3. A dam breaking problem

We set up a dam breaking problem, with the top half multicolored and the bottom half using just one color. The water falls due to gravity and leads to the formation of several overturning waves, entraining the surface texture as well (Figure 6). Here the power of the MLS over the standard level set technique is very obvious: the level set would only be able to produce simulations of unicolored or uncolored transparent surfaces. MLS can instead provide smooth texture dynamics for the whole simulation, regardless of topological changes of any sort. Moreover, by defining the bottom color in this simulation to be transparent one can obtain simulations of mostly transparent water with the surface partially colored (Figure 7). The $20 \times 5 \times 5$ domain was discretized with a $128 \times 32 \times 32$ grid. It is apparent that even for such a coarse grid the results are satisfactory.

4.4. Rendering

All the results in this paper are rendered using the following routine: first a cubic interpolated isosurface is extracted from each level set. The isosurface is rendered using either a standard ray-tracer, without considering color properties, or an MLS-augmented ray-tracer, that uses the markers local to each ray intersection point as the basis for a kernel-based color integration. The kernels are similar to the ones used by the MLS itself for defining the color of the added markers. For rendering the transparent colored water we use a base color (white for our multicolor simulations) as the transparent color and use again local kernel interpolation for smoothing out the transparency regions (namely, the closer a color is to white the more transparent it becomes).

5. Conclusion and future work

We presented a novel method for liquid advection with the additional capability to be used for texture advection. This method, the Marker Level Set, combines in a unique way a level set and a set of markers placed on the interface, such that color information is carried along by the markers and texture advection is thus possible. The MLS was presented in depth and along with it several illustrative simulations were shown (a deformation test, a dam breaking, and a boiling colored liquid experiment). While successful as both a sharp interface tracker and as a texture advecting tool, we feel that there is room for improvement in the way the textures are rendered and initialized. Our particle based rendering method may be updated with mesh-based functions that smoothly integrate the marker color to vertices, prior to rendering. Another exciting avenue for future research would

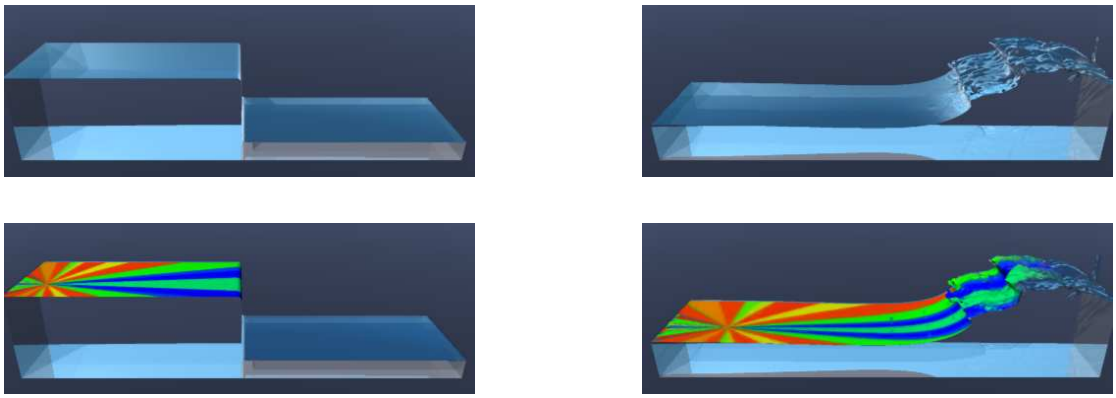

Figure 7: *Dam breaking simulation with transparent rendering. From left to right time t = 0,2 seconds. First row: no color from markers. Second row: with surface color interpolated from markers.*

be transferring the surface color to various objects the liquid interacts with. This seems to require little effort to implement but may prove to be a quite useful extension of the present MLS technology. We also plan to extend the current framework to handle 3D textures besides the liquid surface textures, so that underwater color would be taken in account as well.

6. Acknowledgements

Will be completed after the paper review.

7. References

References

[Ano07] ANONYMOUS: submitted to Journal of Computational Physics.

[BCG89] BELL J., COLELLA P., GLAZ H.: A second-order projection method for the incompressible Navier-Stokes equations. *Journal of Computational Physics 85* (1989), 257–283.

[BGOS06] BARGTEIL A. W., GOKTEKIN T. G., O'BRIEN J. F., STRAIN J. A.: A semi-lagrangian contouring method for fluid simulation. *ACM Transactions on Graphics 25*, 1 (2006).

[EMF02] ENRIGHT D., MARSCHNER S., FEDKIW R.: Animation and rendering of complex water surfaces. *ACM TOG 21*, 3 (2002), 736–744.

[FF01] FOSTER N., FEDKIW R.: Practical animation of liquids. *Proceedings of SIGGRAPH 2001*, 23–30 (2001).

[FM96] FOSTER N., METAXAS D.: Realistic animation of liquids. *Graphical Models and Image Processing 58* (1996), 471–483.

[HK05] HONG J.-M., KIM C.-H.: Discontinuous fluids. In *Siggraph 2005* (2005), ACM Proceeedings, ACM, ACM Press / ACM SIGGRAPH.

[HNB*06] HOUSTON B., NIELSEN M. B., BATTY C., NILSSON O., MUSETH K.: Hierarchical RLE level set: A compact and versatile deformable surface representation. *ACM Transactions on Graphics 25*, 1 (2006), 151–175.

[KAG*05] KEISER R., ADAMS B., GASSER D., BAZZI P., DUTRE P., GROSS M.: A unified Lagrangian approach to solid-fluid animation. *Proceedings of Eurographics Symposium on Point-Based Graphics* (2005).

[KLLR06] KIM B., LIU Y., LLAMAS I., ROSSIGNAC J.: Advections with significantly reduced dissipation and diffusion. *IEEE Transactions on Visualization and Computer Graphics* (2006).

[LGF04] LOSASSO F., GIBOU F., FEDKIW R.: Simulating water and smoke with an octree data structure. *ACM Transactions on Graphics 23* (2004), 457–462.

[LSSF06] LOSASSO F., SHINAR T., SELLE A., FEDKIW R.: Multiple interacting liquids. *ACM Transactions on Graphics* (2006).

[MMS04] MIHALEF V., METAXAS D., SUSSMAN M.: Simulation and control of breaking waves. In *Proceedings of the 2004 Symposium of Computer Animation* (2004), ACM SIGGRAPH/Eurographics, ACM Press.

[Mon92] MONAGHAN J. J.: Smoothed particle hydrodynamics. *Annu. Rev. Astron. Physics 30* (1992), 543.

[MSKG05] MULLER M., SOLENTHALER B., KEISER R., GROSS M.: Particle-based fluid-fluid interaction. In *Proceedings of the 2005 Symposium of Computer Animation* (2005), ACM SIGGRAPH/Eurographics, ACM Press.

[MUM*06] MIHALEF V., UNLUSU B., METAXAS D.,

SUSSMAN M., HUSSAINI Y.: Physics-based boiling simulation. In *Proceedings of the 2006 Symposium of Computer Animation* (2006), ACM SIGGRAPH/Eurographics, ACM Press.

[Ney03] NEYRET F.: Advected textures. *Proceedings of SIGGRAPH/Eurographics Symposium on Computer animation 2* (2003), 147–153.

[PTB*03] PREMOZE S., TASDIZEN T., BIGLER J., LEFOHN A., WHITAKER R. T.: Particle-based simulation of fluids. *Eurographics 22*, 3 (2003), 401–410.

[REN*04] RASSMUSEN N., ENRIGHT D., NGUYEN D., MARINO S., SUMNER N., GEIGER W., HOON S., FEDKIW R.: Directable photorealistic liquids. *Proceedings of SIGGRAPH/Eurographics Symposium on Computer animation 4* (2004), 193–2002.

[RS00] RUSSO G., SMEREKA P.: A remark on computing distance functions. *Journal of Computational Physics 163* (2000), 51–67.

[SSO94] SUSSMAN M., SMEREKA P., OSHER S.: A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics 114* (1994), 146–159.

[Sta99] STAM J.: Stable fluids. *ACM SIGGRAPH 1999* (1999), 121–128.

[Sta03] STAM J.: Flows on surfaces of arbitrary topology. *ACM Transactions On Graphics (TOG) 22*, 3 (2003), 724–731.

[Sus03] SUSSMAN M.: A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *Journal of Computational Physics 187* (2003), 110–136.

[WH04] WIEBE M., HOUSTON B.: The tar monster: Creating a character with fluid simulation. *Proceedings of ACM SIGGRAPH 2004 Sketches and Applications* (2004).

[Wit99] WITTING P.: Computational fluid dynamics in a traditional animation environment. *Proceedings of ACM SIGGRAPH 1999 2* (1999), 129–136.