EUROGRAPHICS 2009 / P. Dutré and M. Stamminger
(Guest Editors)

Volume 28 (2009), Number 2

Simulation of two-phase flow with sub-scale droplet and bubble effects

Viorel Mihalef[1] Dimitris Metaxas[1] Mark Sussman[2]

[1]Center for Computational Biomedicine Imaging and Modeling, Rutgers University , USA
[2]Department of Mathematics, Florida State University, USA

Abstract

We present a new Eulerian-Lagrangian method for physics-based simulation of fluid flow, which includes automatic generation of sub-scale spray and bubbles. The Marker Level Set method is used to provide a simple geometric criterion for free marker generation. A filtering method, inspired from Weber number thresholding, further controls the free marker generation (in a physics-based manner). Two separate models are used, one for sub-scale droplets, the other for sub-scale bubbles. Droplets are evolved in a Newtonian manner, using a density-extension drag force field, while bubbles are evolved using a model based on Stokes' Law. We show that our model for sub-scale droplet and bubble dynamics is simple to couple with a full (macro-scale) Navier-Stokes two-phase flow model and is quite powerful in its applications. Our animations include coarse grained multiphase features interacting with fine scale multiphase features.

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

1. Introduction and previous work

Fluid simulation methods geared towards the special effects industry face the formidable task of being able to handle a wide range of physical scales in order to provide realistic results. In large scale scenes that involve liquid surfaces, the simulation of small scale effects like droplets and bubbles is usually decoupled from the main simulation of the body of liquid, in order to be performed offline. The advantage of this approach is having more control over the small scale structure dynamics, at the cost of missing some of the real physics involved, and most of the times involving the extra geometric cost of "aligning" the small scale dynamics with the already computed larger scale dynamics (for example projecting offline generated foam particles on an already computed ocean surface with intrinsic dynamics). In this paper we present a unified framework for concurrently simulating large and small scale dynamics of liquids, in an attempt to get closer to that perfect simulator of all possible scales, Nature. Solving the 3D Navier-Stokes equations on Eulerian grids can be a very successful approach to obtaining high-quality simulations of the large scale dynamics of liquids, as shown for example by [FM96], [FF01] or [EMF02].

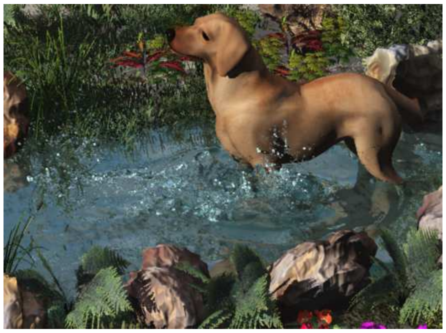

Figure 1: *An application of our new algorithm: spray generation.*

A natural method for tackling smaller scales consists in using grid refinement (e.g. RLE, octrees); however, while this does bring an improvement, it is still not enough for simulating very small scale phenomena such as spray or small

Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

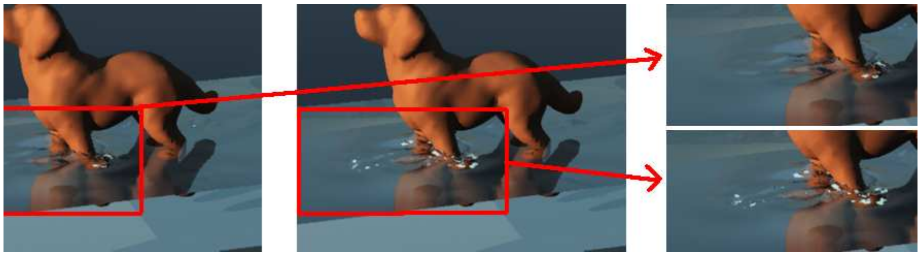

Figure 2: *As a result of our new thresholding technique (left image) extraneous bubbles generated in the middle image (obtained using only a curvature thresholding criterion) are removed after using the combined curvature-velocity gamma parameter. The two right images show close-ups of the region where rogue droplets are eliminated by our method.*

bubbles. Such diffuse/dispersed phases are better simulated with particle methods, like smoothed particle hydrodynamics (SPH). In fact, a recent publication by [LTKF08] proposes a coupling at runtime between Eulerian and SPH components, in order to cover the whole range of scales, relying on each component to cover the scale in which it performs best. The method we propose is similar in spirit, being also an Eulerian-Lagrangian hybrid which models the large scales with the help of the Marker Level Set proposed by [MMS07], while the small scales are tackled by a particle system. However, our method uses a much simpler framework (no SPH involved) while still accounting in a reasonable way for the two-way momentum transfer between the large and small scale phases. As a result, our simulations do not feature the "graininess" usually associated with SPH methods, while still being able to produce the splashes. Moreover, our framework can also provide nice dynamics of small bubbles dispersed in the water phase. This is similar to the functionality provided by the non-dissipative water method proposed by [SSK05]. However, in Song et al.'s approach, the movement of small droplets or bubbles (they handle both) is obtained by standard numerical integration from grid values, hence grid effects will dominate and subscale dynamics will not be very accurate. In contrast, we make use of subscale dynamic models and Lagrangian dynamics, which results in enhanced accuracy.

Our method consists in solving the Navier-Stokes equations on an Eulerian grid, using the Marker Level Set (MLS) as an accurate interface tracker. The MLS method provides naturally a set of deleted markers at every time step, which indicate subgrid information lost by the level set. Note that similar subgrid information has been used in a few papers [GSLF05, KCC*06, LIGF06, GH04] based on the particle level set (PLS) to represent spray or bubbles. MLS is used in a similar manner in our paper, which has not been done before. Also, none of the above papers proposes a unified model for both spray and bubbles, as we do in this paper. Moreover, in this paper we use a full two-phase flow solver for the Navier-Stokes equations (the above papers only use one-phase), which is updated to account for momentum transfer between droplets and air. In order to do this, we use a variable-density framework for the gas phase, so that liquid droplet momentum is properly taken into account after solving the pressure Poisson equation and updating the air velocity. This has a double impact: on one hand, it enables us to define a quite accurate drag force field in the air; on the other, it allows us to automatically (and cheaply) simulate the impact of droplets on the body of water. The set of deleted markers generated by the MLS method is filtered using a physics-based criterion so that we discard free markers generated in regions of small instability (measured by us as a product of velocity variation and curvature). We do keep free markers generated in highly-unstable areas, as a sign of droplet/bubble generation. This is more accurate than older methods like [OH95] that use only a velocity threshold, and is also preferable to the standard simulation methods such as [TFK*03, GSLF05], which use a purely geometric criterion (curvature) to detect the generation of small droplets or bubbles. After all, a curved interfacial region is more likely to break if the local velocity field variation is higher. Finally, our method introduces a simple kinematic model based on Stokes' law to account for small bubble motion. This is comparable in effectiveness with the more complex one proposed by [CPPK07]. In this paper we do not focus on small bubble accumulation and foam generation, as they do, but rather we propose a unified framework for macro and micro scale two-phase flow simulation, which can be quite easily augmented with extra models such as the one proposed by them for beer foam.

Besides the works mentioned above there are several others that need mentioning. We should note, however, that none of them takes into account the balance of aerodynamic and surface tension forces (as characterized by Weber's number) for generation of sub-scale elements (either

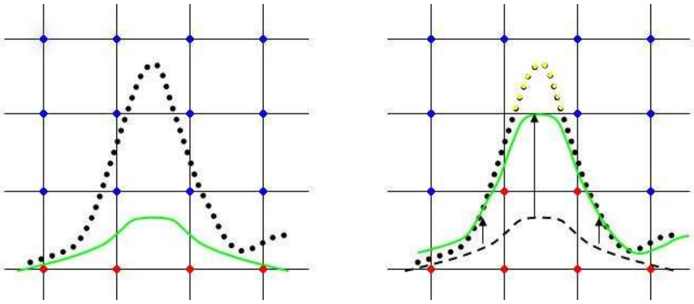

Figure 3: *After one update step of the MLS (left: before update, right: after update), surface markers (yellow) away from the interface (green) are used as an indicator of subscale info lost by the level set. They are deleted and taken into consideration for being transformed into physical free particles. The red nodes are positive (liquid), the blue ones are negative (air).*

droplets or bubbles). Several Siggraph Sketches [BC06, GLR*06,Tho06] have recently reported on various tools that were implemented to aid in the creation of full scale effects. At SCA in 2006 there were two papers [MUM*06, ZYP06] that tackled simulation of bubbles in boiling environments. Their bubbles are large in size compared to ours, consequently they use an Eulerian grid for their advection. This is similar to [HK05] who proposed a useful way to treat variables discontinuous across the interface, and to [KLL*07], who also enforce bubble volume preservation. Droplets are a natural occurrence in any SPH simulation, for example see [LTKF08]. Interesting SPH papers that emphasize bubble simulation are [TSS*07, MSKG05], and more recently (and with more bubbles) [HLYK08].

In the next sections we first present our simulation framework (in section 2), then several results and concluding remarks.

2. Simulation method

Generation of small-scale dynamics has to take into account the ratio of the aerodynamic force related to dynamic pressure, to the force of surface tension. This is expressed in fluid mechanics by an important parameter, largely ignored in computer graphics simulation of liquids, namely the Weber number (as defined for example in [Sir99])

$$We = \frac{\rho \Delta U^2 L}{\sigma} \tag{1}$$

Here ρ is the density, ΔU is the relative gas-liquid velocity, L is the characteristic dimension of the local liquid geometry and σ is the surface tension coefficient (so that σ/L stands for the surface tension force). The aerodynamic force is conveniently defined as $\rho \Delta U^2$, so that it captures both the case of a high gas velocity over slow liquid (a stormy ocean would be a good example), or a high velocity liquid jet in a static gas, breaking into droplets (in which case $\Delta U = U_{liquid}$). Moreover, considering the relative velocity underlines that the parameter concerns specifically the boundary layer, which is the region where droplets will break away from the main body of liquid. Depending on the local shape, breakup can occur above some critical value of We. Below this value, the surface tension forces are large enough to overcome the dynamic pressure, and there is no local breakup. At larger Weber numbers, the domination of the aerodynamic force leads to distortion and spray creation. This process creates small droplets with a smaller Weber number, and, once their size is small enough, they drop below the critical level and the breakup process stops.

One of the concerns of this paper is designing a filter that recognizes the interfacial areas in which such droplets are generated. For this we make use of the Weber number but also take into account the possible artifacts induced by our numerical method. Namely, the MLS method deletes any surface markers from a non-interfacial cell. However, these markers are assumed to possibly represent small droplets detached from the interface only if they are situated in high-curvature locations, otherwise they are discarded. Markers like the yellow ones in Figure 3 would be kept by this reasoning, while if they appeared in fairly flat regions they would most likely be discarded. In computer graphics, such local shape dependence is almost exclusively the only criterion used for detecting when local topological changes occur. Namely, the value of the local mean curvature is used as a threshold that controls the switching from the large-scale simulator (for example finite differences on an Eulerian grid) to a small-scale simulator, like ballistic dynamics for droplets, SPH, etc. In this work we propose to go beyond mere geometry, and take into account dynamics as well. For this, we propose a switching parameter that combines the absolute value of the mean curvature (characterizing possible numerical artifacts) and the square of the velocity (proportional to the Weber number)

$$\gamma = |\kappa|\Delta U^2 \tag{2}$$

We abstract the Weber number as ΔU^2 because the density and surface tension coefficients are fixed for the simulation, while the characteristic dimension L can be taken to equal the grid spacing dx - also fixed. Thresholding the gamma parameter would, according to our discussion, provide the filtering we are looking for. For example, regions of smaller curvature, that would not generate any small scale structures using the standard purely geometric criterion, could still generate some small droplets/bubbles if the velocity were large enough. This alleviates a limitation that the Eulerian framework imposes, in that short time span phenomena like the one just mentioned may be smoothed away by the solver. Moreover, the gamma parameter acts as a filter for parasitic droplets/bubbles that could be generated in regions in which the curvature is higher than normal, but the velocity is small. Such a situation is illustrated in Figure 2, with extraneous droplets generated in the middle image by thresholding of curvature only, and with the extra droplets removed in the left image, after using the velocity as well, as part of the gamma filter.

Other novelties that our paper proposes, besides this physics-based gamma-filtering technique, are the following: an extension of the MLS method to generate a first batch of free markers, in regions of high curvature, where the level set loses accuracy; a coupling technique from droplets to Eulerian grid which ensures quite accurate transfer of momentum; an adaptation of the two-phase flow model used by [MUM*06], which employed the CLSVOF method of [SP00] to use with the MLS method. We will provide more details on each of these in the subsequent sections.

First, we will give an overview of our simulation method. We use two level set functions, one for modeling the liquid/gas phase (positive in the liquid, negative in the gas), the other to account for the presence of solid geometries in the domain (positive in the fluid, negative in the solid). This setup follows the method proposed by [MKM*08] for interaction of pre-animated models with liquid flow. There are two main differences between our approach and theirs, namely here we actually have interaction of the animated meshes with the whole two-phase flow (they used one-phase flow), and our fluid level set is replaced by MLS, in our case (they used CLSVOF). We solve a variable density Navier-Stokes in the whole domain, and use deleted markers from MLS to account for possible small-scale structure generation. After a filtering step based on the gamma number we use these markers to generate small-scale dynamics, namely markers inside the water will be treated as small bubbles, while markers in the air will be treated as droplets. Both sets of markers are evolved in a Lagrangian manner, based on physically-meaningful rules. In particular, momentum is transferred back from the droplets to the Eulerian grid through local extrapolation. As an *important convention*, when we talk in this paper about markers, from now on we will mean either the surface markers specific to the MLS method, or the deleted markers that we generate from the MLS. In other words, markers only have geometric meaning. Once the deleted markers have been gamma-filtered they will be called particles, or free particles, and take on physical meaning (they will carry density properties) for example. Thus, at any time step our algorithm does the following:

1. Advection: first the MLS markers and the free particles, then the level sets
2. Particle generation: first generate new MLS deleted markers, then use gamma-filtering to produce new particles
3. the standard MLS correction steps
4. Air density change (for droplet-to-air momentum transfer)
5. Momentum solver (variable density Navier-Stokes)

Our momentum solver is similar to [MUM*06] or [SSH*07] and we refer the reader to those papers. Our water density was $1g/cm^3$ and air density was $0.00125g/cm^3$ in all simulations. The viscosity and surface tension were set to zero, unless noted otherwise. Our MLS solver is similar to the one introduced by [MMS07], with the exception that the smoothing kernel parameters we use are $c = 0.2$ and a kernel radius of $\sqrt{3}$ (these values are in grid units). We use un upper-limit of 50 markers/cell. In the following we will focus on the marker and particle generation and on the particle dynamics, for both droplets and bubbles.

2.1. Free marker and particle generation

The MLS method is a hybrid method that relies on a set of surface markers that are used to send and receive information from an underlying level set, which is updated based on the more accurate positions of the markers. Similarly with the PLS, even after this correcting update, the Eulerian level set may not align perfectly with the Lagrangian markers, as can be seen in Figure 3. This happens especially in high-curvature regions, therefore the extra markers in the top cell in the figure can be used to mark exactly such regions. They are detected with a simple routine that checks for each cell if the number of markers in that cell is nonzero, but the level set values at the cell corners do not change sign, which means that the level set does not detect an interface present in the cell. Such “free” markers are given the opposite sign of the phase they reside in, for example in the figure the yellow markers become free positive particles, namely droplets. Of course one can use various geometric techniques to get a better covering with markers of the lost volume of fluid, but this first approximation we used already gave good results in practice. This set of free markers is subsequently filtered using the $\gamma = |\kappa|\Delta U^2$ parameter. We compute ΔU as a variation per cell of the velocity, in all the interfacial cells. The mean curvature at the cell center is approximated using the

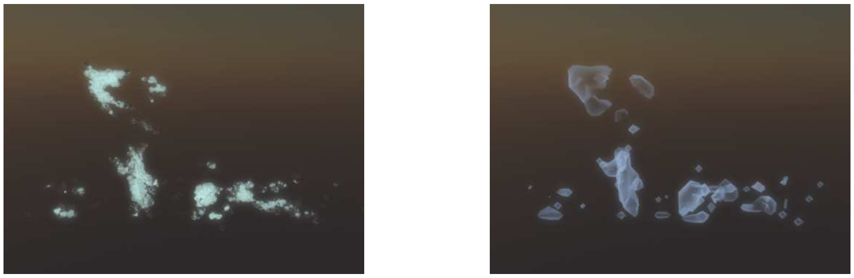

Figure 4: *Left: droplet positions. Right: a level set of the extrapolated density field shows good agreement with droplet position. See section 2.2.2 and the Appendix for more details.*

usual formulas based on the level set function (see for example [OF02], page 12).

2.2. Small particle dynamics

After the gamma-filtering step we have a set of physical small resolution particles to deal with. Both droplets and bubbles are ballistically advected using drag force fields based on the grid velocity. In the case of droplets one has to also take into consideration how their own momentum affects the Eulerian simulation, while in the case of bubbles their momentum is negligible (and neglected by us). The radius of the particles is varied around a mean value of 1mm, using a finite support normal random field, similarly with [GH04]. This helps quite a bit visually the final simulation results. For example, some (smaller) bubbles travel more slowly to the surface, while heavier droplets fall faster.

2.2.1. Collisions

Inter-particle collision is known to affect the dynamics of dense sprays. The interested reader can find a good discussion of experimental results in [QL97], detailing the various regimes of interacting droplets, namely coalescence, bouncing, near head-on and off-center separation. A recent paper proposing a new parametric model for droplet interaction that may be adapted for graphics purposes is [MR07]. Bubble interaction is less well researched. However, the surface tension forces at the small sizes considered in this paper are much higher than the colliding forces, due to the small momentum of bubbles, therefore small bubble collision was disregarded. We do take into account droplet interaction. We implemented an algorithm that handles collision efficiently in a hybrid Eulerian-Lagrangian manner, and we describe it in the (next) droplet section. Finally, in terms of interaction with the liquid, we check at each time step the position of the particles and delete them accordingly if they enter their own phase type.

2.2.2. Droplets

For droplets we use Newton's law with drag forces enabled, namely solving the ODE system:

$$\begin{aligned} m\frac{du}{dt} &= F_{drag}(x) + m\mathbf{g} \\ \frac{dx}{dt} &= u(x) \end{aligned}$$

Here m is the particle mass, while the drag force is $F_{drag} = -\alpha(u-U)^{\beta}$, where U is the underlying air velocity, α is a multiplication parameter and β is 1 for low Reynolds numbers (Stokes flow) or 1.28 for higher Reynolds numbers. Due to the cubic dependence of the mass on the particle radius, the smaller particles are much more heavily influenced by drag, therefore we implemented the lower Reynolds number formulation (recall that the Reynolds number and the radius are proportional). We detail here the two possible formulations for the drag force, following [Sir99]. First let us recall that the Reynolds number $Re = \frac{2VR}{\nu_g}$ is a measure of the balance of inertial forces and viscous forces. Here V is the (relative) velocity , R is the local characteristic dimension (e.g. radius for a spherical particle) and ν_g is the kinematic gas viscosity. For low Re numbers (less than 1), corresponding to slower flow and/or smaller particles, the viscous effects are defining for the drag, and the absolute drag force takes the form $F^1_{drag} = 6\pi\nu_g\rho_g R\Delta u$. For higher Re numbers ([Sir99] cites $Re > 30$) the pressure drag is dominant and the absolute drag force is better approximated by the formula $F^2_{drag} = \frac{C_D}{2}\rho_g\pi R^2\Delta u^2$ (the more familiar "square velocity dependence"). Here C_D is the drag coefficient, which (very importantly!) is dependent on Re, for example for $Re < 1$ one has, up to a good approximation, $C_D = \frac{24}{Re}$. Note that for such a Re one obtains, by direct substitution of the Re definition in the second drag force formulation, that the two drag formulations are identical $F^1_{drag} = F^2_{drag}$. For $Re > 30$ the drag coefficient is better approximated by $C_D = \frac{24}{Re^{0.72}}$. By substitution one obtains that the drag force becomes $F^2_{drag} = 22.8\rho_g\nu^{0.72}(R\Delta u)^{1.28}$. For a unified description of

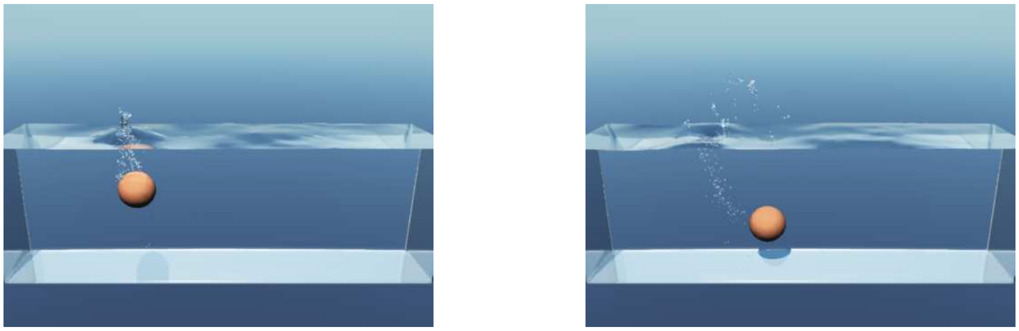

Figure 5: *Ball splashing simulation, showing formation of bubbles and droplets.*

the drag force one could use the second formulation and interpolate the *Re* exponent between 1 and 0.72 on the reference Reynolds number interval $[1, 30]$.

The air velocity is computed by the two-phase flow Navier-Stokes solver. In order for this velocity to be meaningful we need to take into consideration the local droplet density. To this end we extrapolate at each node location the density values from the neighboring droplet, by using the same kernel parameters as used for MLS. We use a cutoff equal to the liquid density in case the new interpolated density overshoots the liquid density. We describe the extrapolation technique in detail in the Appendix section. The final results can be seen in Figure 4. It is visible that the new density gives a nice approximation of the particle positions, and, as a consequence, also contributes automatically to the transfer of momentum from the droplets to main body of water.

Moreover, the droplet density function allows us to model collisions by assuming that the collision probability is directly proportional with the droplet density. If the cell centered droplet density exceeds a certain threshold (very close to the liquid density) we declare that collisions are taking place inside the cell. We assume that all collisions are coalescent and create larger liquid patches, which are determined geometrically from the local density values. In order to implement this we change the local cell level set values proportionally with the density. As a consequence, the highest density values will lie inside the newly formed liquid patch. This models local droplet collision by restituting the coalesced droplets to the body of liquid. Subsequently the new patch of liquid is treated using Navier-Stokes just like the main body of liquid and all topological changes are taken care of by the level set.

In order to approximately conserve momentum we also make sure to modify the local Eulerian velocity appropriately. This is important especially for single particles reentering the body of liquid. To this end, the local (gas) velocity is overwritten using local extrapolation of the particle velocity components in exactly the same manner as for density (described in the Appendix).

2.2.3. Bubbles

We modeled the dynamics of small bubbles by regarding them as spherical bubbles in Stokes flow. Bubble hydrodynamics theory says that, after its formation, a bubble rapidly accelerates to its terminal velocity. This terminal velocity is determined by the balance between the buoyant rise force, and the drag force. This can be calculated for small, spherical bubbles with Reynolds number less than one, yielding the general form of Stokes' Law: $v_{Stokes} = -\frac{2}{9}\frac{(\rho_{liquid}-\rho_{bubble})\mathbf{g}r^2}{\mu}$, where $\mathbf{g}$ is the gravity vector, r is the bubble radius and μ is the dynamic viscosity. In this work we assumed that $\rho_{liquid} \gg \rho_{bubble}$, in which case one obtains the simpler form of Stokes' Law that we implemented: $v_{Stokes} = -\frac{2}{9}\frac{\mathbf{g}r^2}{\nu}$, where ν is the kinematic viscosity of the liquid. The Stokes velocity was added to the local liquid velocity (obtained by interpolation of the grid values to the bubble position) to give the final bubble velocity: $v_{bubble} = v_{Stokes} + v_{liquid}$. The viscosity coefficient was the one for water, $10^{-6}m^2/s$. As a final note, the general form of Stokes' Law mentioned above can be also used for mixtures of two liquids with different densities.

3. Results

We built a simulation system that solves the Navier-Stokes equations with subscale effects enabled or disabled. As the turbulence of the fluid motion was increased, we noticed a proportionally increased number of droplets and bubbles. This is a realistic and expected consequence of the application of the gamma-filtering technique. We present several simulation images and accompanying videos, and show how their visual richness is enhanced when we also render the subscale structure (droplets and bubbles). All computations

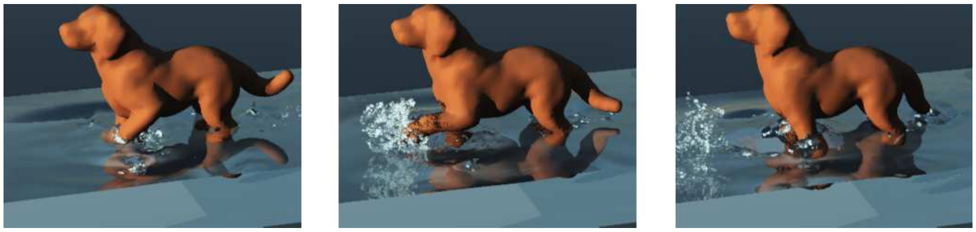

Figure 6: *Water pouncing dog simulation (fast version). Frames at times t = 0.375, 0.425, 0.46 seconds.*

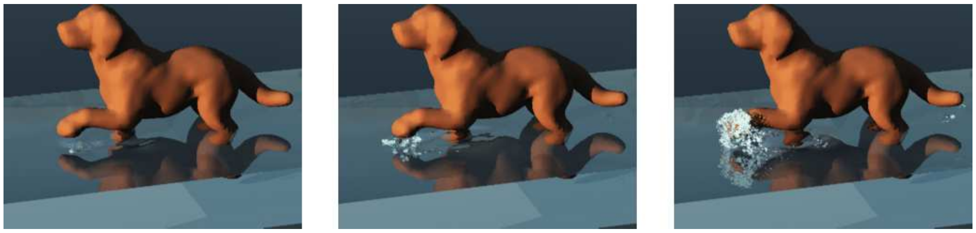

Figure 7: *Comparative look at water pouncing dog simulations. Slow, medium ($\times 2$) and fast ($\times 4$) versions of the dog motion are respectively shown from left to right, along with corresponding spray formation.*

were performed on a 2.4GHz Intel Core 2 Quad machine (although we did not use multi-threading). Our simulation domain was a rectangular box and solid objects along with gas/liquid interfaces were assumed to be embedded within this box. Particles were deleted once they exited the domain. The particle disappearance can be precluded either by using a larger simulation domain, or by using pure ballistics (no drag) in the case of droplets.

The simulations were ray traced in Vue D'Esprit. In several of the simulations we ray traced the particles as possibly overlapping spheres and obtained a "foamy" effect, albeit at the price of aliasing in some cases. While this effect was also useful for an easier visual identification of turbulent areas, a more correct approach would be to single ray-trace the overlapping areas. We achieved this as well using boolean unions of the particle spheres (including the union of droplets with the main body of liquid). Using the sphere blending capability of Vue we also ray traced overlapping spheres as blended boolean unions. This improved the visual quality of our animations, especially in the closeup scenes (as visible in the attached video). In the following we take a closer look at the various test and showcase simulations.

3.1. Splashing ball, a basic test of the system.

The "ball simulation" (Figure 5) was our basic test for the performance of the system when using an external object influence. It features a pre-animated ball falling into a water pool, and circling out of the water once before falling again inside. The included figures and videos illustrate the formation and dynamics of both bubbles and droplets. Due to the drag forces, bubbles rise and droplets fall faster or slower, depending on their size. The physical dimensions of the simulation domain were $0.2m \times 1.2m \times 1.2m$. The domain was discretized on a $16 \times 96 \times 96$ grid. The computations took about 30 seconds per frame. The maximum bubble and droplet count was around 500, usually staying in the one to two hundred level.

3.2. Droplet showcase: water pouncing dog.

As a showcase for the droplet generation and dynamics we animated a dog pouncing at a pool of water (Figure 6), with her tail occasionally touching the water as well. Most visible are the droplets generated when the paw or the tail leave the water, so this can be a good showcase for droplet formation. We started from a basic motion of the dog, and then repeated the calculations first doubling, then quadrupling the rate of motion for the dog; this is done in order to assess the influence of the variable velocity on the spray formation. In Figure 7 we show a frame from each of these simulations, chosen at the same dog animation frame. Consequently, the corresponding timings were half, respectively a quarter of the original animation (on the left). It is noticeable how the increased velocity contributes decisively to the generation of extra spray. The computational domain measured $1.2m \times 0.6m \times 0.6m$, and was discretized on a $128 \times 64 \times 64$

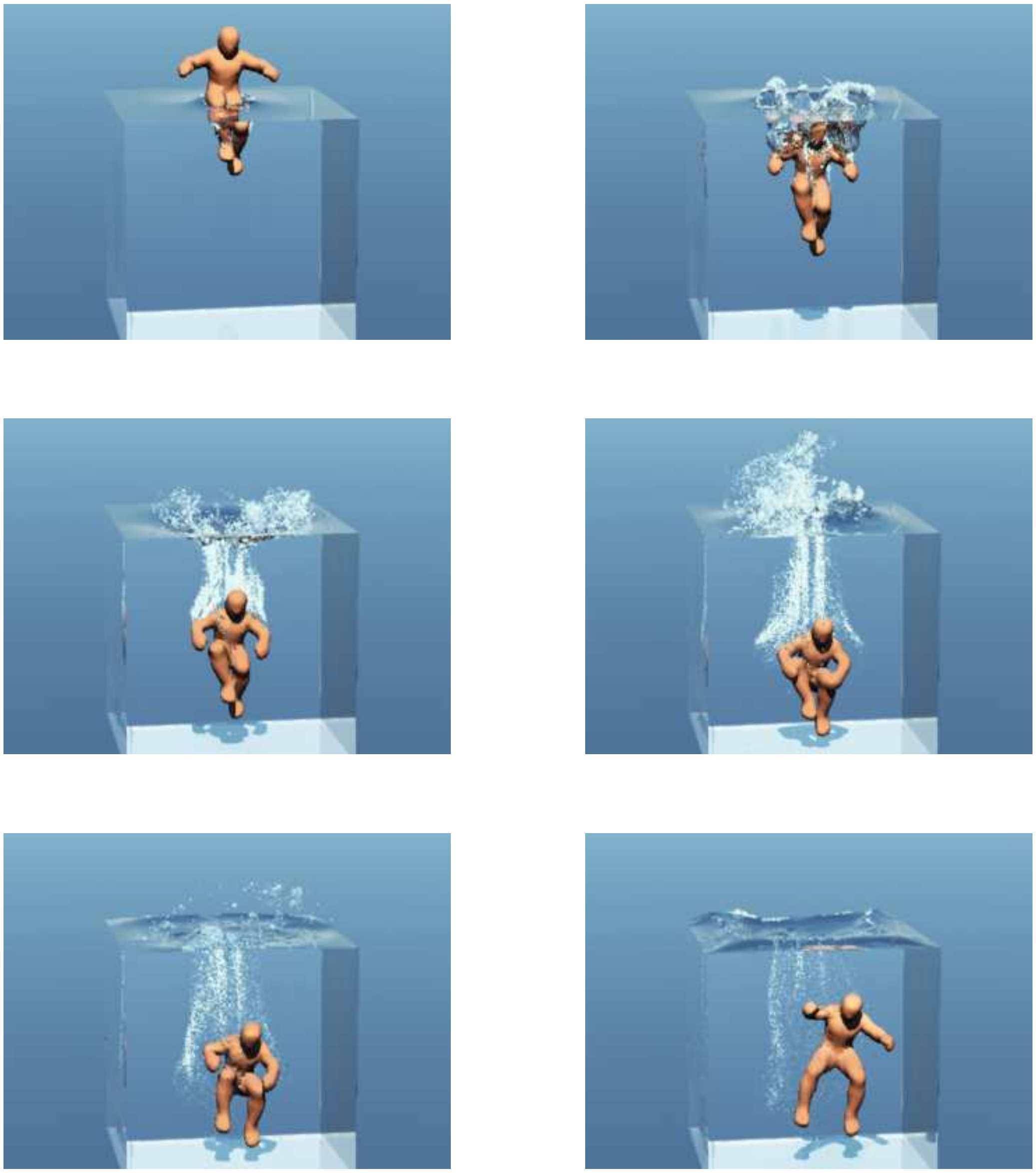

Figure 8: *Diver simulation. From left to right and top to bottom, times t = 0.125, 0.165, 0.215, 0.335,0.455,0.585 seconds.*

grid. The program generated a frame on average every 4 minutes.

We used this setup also for showing the enhancement that the use of the gamma parameter brings into play. In Figure 2 and in the associated video we notice how several regions of small velocity are incorrectly generating droplets when using a curvature-only thresholding technique. They show up as high-frequency noise in the video. The animations using the gamma-thresholding do not suffer from such issues, because they tie droplet or bubble generation to the local physics, involving the velocity variation as well.

3.3. Bubble showcase: pool diver.

Another simulation we produced is that of a diver performing a cannonball jump. We used a $72 \times 72 \times 144$ grid that discretized a $1m \times 1m \times 2m$ physical domain and ran at about one frame every 7 minutes. Viscosity and surface tension were also enabled. As seen in Figure 8, our system is capable of generating a wealth of bubbles and droplets. A more refined grid could generate even more subscale structures, and therefore would be especially useful for generating realistic simulations of pool divers surrounded by bubble clouds.

3.4. A post-processing alternative.

A small variation of our system allows the user to define both the droplet and the bubble dynamics as a post-processing step, the trade-off being losing some dynamics realism. In order to set up our one-phase "control" solver, we saved on the hard disk, at each time step, the *generated* particles and the grid velocity files. We used this information in post-processing, by simply tracing the particle trajectories, and deleting them once they entered their own phase. The droplet motion used the dynamic formulation outlined in section 2.2.2, with zero drag, while the bubble motion used the formulation from section 2.2.3. This post-processing setup is ideal for finding the intended dynamics of the small scale structures by the animator after running only once the Navier-Stokes solver. Post-processing trials, which may include varying geometric parameters like the droplet size or physical parameters like fluid viscosity, enable the animator to find the "best" dynamics for the droplets and bubbles. Afterwards, one can run again the two-phase Navier-Stokes solver with the newly found parameters in order to obtain a more physically accurate simulation. A video with post-processed diver simulations at a $64 \times 64 \times 128$ resolution is available. The number of free markers is set to be smaller than the one used in the previous subsection.

3.5. Physical parameter control

We note that the framework we introduced in this paper features several physical parameters that can also serve as control parameters that animators can use. Increasing the γ parameter decreases the number of particles. The drag coefficient α controls how much a droplet influences (or is influenced) by the local velocity. Increasing it would create more of a "flocking" behavior for the droplets. For more control, one can also make the drag coefficient dependent on the droplet radius. Last, but not least, the bubble radii and the gas viscosity can be used to create more drag (for smaller radii or larger viscosity). As mentioned elsewhere in the paper, varying the radii of the bubbles and droplets is instrumental in creating realistic simulations. Using more realistic shapes for the droplets and bubbles, rather than simply spheres, as we did here, is another method to enhance realism.

4. Conclusion

We presented in this paper a unified framework for macro and micro scale fluid simulation, based on an extension of the Marker Level Set method. It is a hybrid Eulerian-Lagrangian method in which the Eulerian solves the variable-density two-phase flow Navier-Stokes using the MLS method as the interface tracker, while the Lagrangian part advects ballistically the subscale structures (bubbles and droplets) taking into account drag forces as well. Momentum is transferred from droplets to the body of liquid using a local extrapolation of the droplet density to the grid. A novel method for detecting small scale structure generation inspired from Weber number thresholding refreshes the older technology used in computer graphics based only on geometric criteria (curvature value).

Although our framework is very useful for providing small scale structures, one can observe that, for certain simulations, the final results could be enhanced if the droplets would merge together and form thin sheets. This was not a focus of this paper, but we plan to explore using the extrapolated density function defined in subsection 2.2.2 for capturing such droplet merger phenomena. Similarly, we plan to explore implementing more advanced mechanisms for bubble and droplet interaction, like the one proposed by [MR07]. Also as future work we would like to use the particles deleted upon crossing into the same phase as themselves to generate local texture information, stored in the MLS surface markers. This, combined with defining rules for foam dynamics, should enable us to obtain physics-based simulation of ocean foam.

Acknowledgements

The authors want to thank the reviewers for their very useful suggestions and questions that led to substantial improvement of the original exposition. The first author thanks Iuliana Radu for her great help with proofreading and generating Figure 3. The first two authors were supported by NSF grant 0702591.

References

[BC06] Brown S., Collier R.: Flooding ice age: The meltdown using wavesynth and point based froth. *ACM Siggraph Sketches* (2006).

[CPPK07] Cleary P. W., Pyo S. H., Prakash M., Koo B. K.: Bubbling and frothing liquids. In *ACM TOG* (2007), vol. 26.

[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* (2001), 23–30.

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

[GH04] Greenwood S. T., House D. H.: Better with bubbles: enhancing the visual realism of simulated fluid. In *Proceedings of the 2004 Symposium of Computer Animation* (2004), ACM SIGGRAPH/Eurographics, ACM Press, pp. 287–296.

[GLR*06] Geiger W., Leo M., Rasmussen N., Losasso F., Fedkiw R.: So real it'll make you wet. *ACM Siggraph Sketches* (2006).

[GSLF05] Guendelman E., Selle A., Losasso F., Fedkiw R.: Coupling water and smoke to thin deformable and rigid shells. *ACM TOG 24* (2005), 973–981.

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

[HLYK08] Hong J.-M., Lee H.-Y., Yoon J.-C., Kim C.-H.: Bubbles alive. *Proceedings of SIGGRAPH 2008* (2008).

[KCC*06] Kim J., Cha D., Chang B., Koo B., Ihm I.: Practical animation of turbulent splashing water. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium of Computer Animation* (2006), ACM SIGGRAPH/Eurographics, ACM Press, pp. 335–344.

[KLL*07] Kim B., Liu Y., Llamas I., Jiao X., Rossignac J.: Simulation of bubbles in foam with the volume control method. *ACM TOG 26*, 3 (2007).

[LIGF06] Losasso F., Irving G., Guendelman E., Fedkiw R.: Melting and burning solids into liquids and gases. *IEEE TVCG 12* (2006), 343–352.

[LTKF08] Losasso F., Talton J., Kwatra N., Fedkiw R.: Two-way coupled SPH and particle level set fluid simulation. *IEEE TVCG 14*, 4 (2008), 797–804.

[MKM*08] Mihalef V., Kadioglu S., M.Sussman, Metaxas D., Hurmusiadis V.: Interaction of multiphase flow with animated models. *Graphical Models (in press)* (2008).

[MMS07] Mihalef V., Metaxas D., Sussman M.: Textured liquids based on the marker level set. *Computer Graphics Forum 26*, 3 (2007).

[MR07] Munnannur A., Reitz R. D.: A new predictive model for fragmenting and non-fragmenting binary droplet collisions. *International Journal of Multiphase Flow 33* (2007), 873–896.

[MSKG05] Muller M., Solenthaler B., Keiser R., Gross M.: Particle-based fluid-fluid interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium of Computer Animation* (2005), ACM SIGGRAPH/Eurographics, ACM Press, pp. 237 – 244.

[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.

[OF02] Osher S., Fedkiw R.: *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, New York, 2002.

[OH95] O'Brien J. F., Hodgins J. K.: Dynamic simulation of splashing fluids. In *Computer Animation* (1995), pp. 198–205.

[QL97] Qian J., Law C.: Regimes of coalescence and separation in droplet collision. *Journal of Fluid Mechanics 331* (1997), 59–80.

[Sir99] Sirignano W. A.: *Fluid Dynamics and Transport of Droplets and Sprays*. Cambridge University Press, 1999.

[SP00] Sussman M., Puckett E. G.: A coupled level set and volume of fluid method for computing 3D and axisymmetric incompressible two-phase flows. *Journal of Computational Physics 162* (2000), 301–337.

[SSH*07] Sussman M., Smith K., Hussaini M., Ohta M., Zhi-Wei R.: A sharp interface method for incompressible two-phase flows. *Journal of Computational Physics 221*, 2 (2007), 469–505.

[SSK05] Song O., Shin H., Ko H.: Stable but nondissipative water. *ACM TOG 24*, 1 (2005), 81–97.

[TFK*03] Takahashi T., Fujii H., Kunimatsu A., Hiwada K., Saito T., Tanaka K., Ueki H.: Realistic animation of fluid with splash and foam. In *Eurographics 2003* (2003).

[Tho06] Thornton J. D.: Directable simulation of stylized water splash effects in 3d space. *ACM Siggraph Sketches* (2006).

[TSS*07] Thuerey N., Sadlo F., Schirm S., Mueller M., Gross M.: Real-time simulations of bubbles and foam within a shallow-water framework. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium of Computer Animation* (2007), ACM SIGGRAPH/Eurographics, ACM Press, pp. 191 – 198.

[ZYP06] Zheng W., Yong J., Paul J.: Simulation of bubbles. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium of Computer Animation* (2006), ACM SIGGRAPH/Eurographics, ACM Press, pp. 325 – 333.

5. Appendix

As we have seen, in order to be able to use the air velocity as a drag factor, we need to take into consideration the local droplet density. To this end we extrapolate at each node location the density values from the neighboring droplet. Namely, we compute the density at each grid node (i,j) from the air region as $\rho_{i,j}^{new} = \sum w_k(p_k)\rho_k$, where the sum is taken over all particles p_k in a neighborhood of the node. We denoted by ρ_k the density of the particle p_k. In our case all particles are liquid droplets, hence we have that $\rho_k = \rho_{liquid}$ and all we have to do is multiply that with a sum of weights. The weight functions w_k are similar to the ones used by MLS, namely

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

where $q = q_{i,j}(x) = dist(x,(i,j))/\sigma$, $c \in [0,1]$ is a constant and σ is the kernel radius (the distance beyond which the weight vanishes). In our computations we used $c = 0.2$ and $\sigma = \sqrt{3}$. We use a cutoff equal to the liquid density in case the new interpolated density overshoots the liquid density. Namely, the final air density equals

$$\rho_{air} = \begin{cases} \rho^{new}, & \rho_{new} < \rho_{liquid}; \\ \rho_{liquid}, & \text{otherwise.} \end{cases} \tag{4}$$