Surface Tension Approximation in Semi-Lagrangian Level Set Based Fluid Simulations for Computer Graphics

Israel Pineda and Oubong Gwun
Chonbuk National University
israel_pineda_arias@yahoo.com, obgwun@jbnu.ac.kr

ABSTRACT

In this work we present an approximation of surface tension intended to create more visually plausible results of a large-scale fluid simulation for Computer Graphics applications. We modify the computation of the Navier-Stokes equations to work with an alternative version of the velocity field that includes our additional logic. We describe how to compute this additional information and how to store it. This leads to more realistic results of the simulation with only a minimum increase in computation time. After presenting some background information, we show the details of how our algorithm accomplishes its goal and then we present some results and the corresponding analysis.

KEYWORDS

Surface Tension, Fluid Simulation, Semi-Lagrangian, Level Set, Computer Graphics

1 INTRODUCTION

Surface Tension is one of the many forces that shape the behavior of water and other fluids. It is especially true when two or more fluids interact leading to an interface between them. In computer based simulations, particularly in Computer Graphics field, this force is usually omitted because it is considered that there is no significant effect when applying it. Commonly, when the object of the simulation is a large-scale volume of fluid. In the opposite case, low-scale fluid simulations need surface tension to achieve most of the desired effects like drop formations and capillarity. In our case, we explore surface tension only for large-scale scenarios.

After presenting the physical background and the context needed for our simulation we will present our method to improve the quality of the simulation through the inclusion of surface tension. Our method incorporates surface tension force into a Semi-Lagrangian fluid solver. We describe how to modify the fluid solver to include this new force in a dense and coarse grid and analyze the numerical and visual results obtained from the experiment. Finally, we present our results and conclusions.

With our implementation we have been able to replicate the shivering of the water when it is poured in a recipient or when the container itself moves. The shivering effect gives extra detail in the fluid in certain scenarios and it is very often found in nature. This kind of behavior can produce significant improvement in the visual quality of the simulation.

2 RELATED WORK

Different scientific disciplines have done a vast study of surface tension. They all have studied surface tension with their own goals, approaches and methods. In this section, we briefly describe the core ideas and concepts of the related work.

The mathematical model behind the vast majority of fluid simulations is given by Navier-Stokes equations. It gives the necessary fundamentals to understand surface tension and to simulate it. Some simulations also make use of the Young equation. More details about this equations in section §3.

The building blocks for our fluid solver are the
well discussed techniques for fluid simulations as stated by Bridson [1] who describes how to use and implement the Navier-Stokes equations and where the reader can find all the details for a standard computer based fluid simulation. Enright et al. [2] used the level set method to track the fluid surface, other authors have also studied similar and related approaches [2, 3, 4, 5]. Batty et al. [6] extend the ideas for a better solid-fluid coupling.

With the equations and the work mentioned a fluid simulation can be implemented.

Computer Graphics is concerned about obtaining visual plausible results in an affordable time. In this regard, most of the time it has to sacrifice some information for acceleration. In fluids simulations it is often surface tension, which is considered as not necessary for most of the previous research. However, more holistic approaches exists, like the extension of the Ghost Fluid Method (GFM) by Kang et al.[7, 8] where viscosity, surface tension and gravity are considered by storing additional data in what is called “ghost cells”. Kang[4] describes an approach applied to the simulation of soup bubbles. Another work that uses surface tension is Losasso[9]. More recently Clausen et al. [10] have studied related topics.

There are two different scenarios for fluid simulation in Computer Graphics and they are usually handled as totally different simulations. The first one is when we simulate large-scale amounts of the fluid and the second one is when we simulate small-scale amounts of fluids. The former case deals with medium to big bodies of fluid and the latter with small bodies of fluid usually in the order of millimeters. In each case there are some forces that affect less than others, mainly the surface tension and buoyancy forces are both considered not strictly necessary for large-scale fluids, therefore, they are usually omitted. For further reference you can review work on large-scale fluid simulation (e.g., [11]). In the latter case, for small-scale fluid simulations the reader could look through related papers [12, 13, 14, 15].

The difference of our algorithm with the previous work is that most of it relies in the usage of triangle meshes or layers of processing [11] or it actually manipulates the pressure, in this latest case the computing time is affected. For the first difference we work with level set method to track the surface so we do not need a mesh, of course we can get it if needed, especially for rendering using marching cubes algorithm. In the second difference our algorithm does not actually modify the pressure like GFM, at least, not directly, so it does not affect the computation time required.

3 PHYSICAL BACKGROUND

The physical behavior of a fluid is encapsulated in a set of Partial Differential Equations (PDE), more specifically, the Navier-Stokes set of equations and the Young equation give the mathematical model. Hereby, in this section we briefly review the most important concepts we need to understand in order to appreciate our proposed method and refer the reader to the relevant documentation.

Navier-Stokes equations describe the behavior of the fluid in terms of its local velocities, traditionally the Navier-Stokes equations are represented as:

\[ \frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u} \]  

(1)

\[ \nabla \cdot \vec{u} = 0 \]  

(2)

Where, equation (1) is known as the momentum equation and produces a velocity field \( \vec{u} \) that describes the movement of the fluid. And, equation (2) known as the incompressibility equation ensures a constant volume of the fluid over time.

Additionally, to be able to track where the surface of the fluid defined by equations (1) and (2) is at a given time, we have to solve the Level Set equation. Level Set equations is defined as:

\[ \phi_t + \vec{u} \cdot \nabla \phi = 0 \]  

(3)

In the other hand, the Young equation is only applicable when the target phenomena is drop-like formations using the contact angle
Table 1: Nomenclature

<table>
<thead>
<tr>
<th>Equation</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navier-Stokes</td>
<td>(\vec{u})</td>
<td>Velocity field</td>
</tr>
<tr>
<td></td>
<td>(\nabla \vec{u})</td>
<td>Gradient</td>
</tr>
<tr>
<td></td>
<td>(\rho)</td>
<td>Density</td>
</tr>
<tr>
<td></td>
<td>(p)</td>
<td>Pressure</td>
</tr>
<tr>
<td></td>
<td>(\nabla \cdot \nabla \vec{u})</td>
<td>Laplacian</td>
</tr>
<tr>
<td></td>
<td>(\vec{g})</td>
<td>Gravity</td>
</tr>
<tr>
<td></td>
<td>(\nu)</td>
<td>Viscosity</td>
</tr>
<tr>
<td>Young Eq.</td>
<td>(\gamma)</td>
<td>Surface Tension</td>
</tr>
<tr>
<td></td>
<td>(s_g)</td>
<td>Solid-Gas</td>
</tr>
<tr>
<td></td>
<td>(s_l)</td>
<td>Solid-Liquid</td>
</tr>
<tr>
<td></td>
<td>(l_g)</td>
<td>Liquid-Gas</td>
</tr>
<tr>
<td></td>
<td>(\theta)</td>
<td>Contact angle</td>
</tr>
</tbody>
</table>

criteria, and it is not always consider for simulations. The equation is:

\[
\gamma_{sg} = \gamma_{sl} + \gamma_{lg} \cos \theta
\]  

(4)

For detailed interpretation of the symbols in the presented equations refer to table 1.

Bridson[1] studies the graphics approach to Navier-Stokes in more detail and Wang et al. [12] provide more information about the Young equation.

Surface Tension, itself, is not a characteristic of the material. This means that surface tension is a property of two bodies interacting with each other. However, it is important to mention that due to the characteristics of air this is usually considered to have zero effect in simulations. All the forces and computations are executed only for the fluid. This implies that we need to find where and how to allocate the surface tension information of our algorithm within the fluid.

4 THE ALGORITHM

In this section we present the details of our proposed method. We start showing how to obtain all the necessary information for the computation, then we dig into the details of the algorithm itself and lastly we discuss how these modifications affect the overall simulation leading to results shown in section §5.

4.1 Computing Input Parameters

One strength of our method is that all the necessary data for its implementation can be easily found in any standard fluid solver. Namely, the input parameters we need are: mean curvature \(\kappa\), surface tension factor \(\gamma\), normal at a given point \(\vec{N}\) and distance for the updating band \(T\). The least problematic one is of course the normal \(\vec{N}\) which has well established ways to be computed. In the case of surface tension factor \(\gamma\), it is a configurable parameter that has to be set by the user according to the simulation, in the case of water-air interface at normal conditions it is \(0.073 \cdot \frac{1}{\text{m}^2}\). Similarly, the threshold \(T\) that defines how much of the fluid will be consider is a user defined value.

However, mean curvature \(\kappa\) is a more interesting value. Bridson [1] shows that the mean curvature can be calculated using the Laplacian operator, which is usually already implemented in fluid solvers due to its frequent appearance in this kind of simulations. The mean curvature \(\kappa\) can be define by equation (5) where it is computed as the divergence \((\nabla \cdot)\) of the gradient \((\nabla)\).

\[
\kappa = \nabla^2 \phi = \nabla \cdot \nabla \phi
\]  

(5)

With the parameters described we can modify the Navier-Stokes equation as presented in the following section.

One inconvenience of our method is that in order to know which section of the fluid will be affected we need to compute the distance function \(\phi\) in an earlier stage. This is because in a fluid solver the distance function \(\phi\) is typically computed when the viscosity is applied and not before, as it is in our case. However, it is not a problem because we use this value only as input, we do not modify it, so we can still use it in the next stage avoiding duplicated computation.

4.2 Description of the Method

Broadly speaking we work with a Semi-Lagrangian convection simulation over a standard Marker-And-Cell (MAC) grid and use a level set to track the fluid surface. Our simulation is based on the well
As usual in fluid simulation of Computer Graphics we neglected the gas, usually air, outside the liquid body for our tests, however, because our implementation is based on Losasso et al. [9], it should be no problem to simulate the air independently or include more fluids. That means that even though we could use two level sets, one for water and one for the air, in our experiments we have only dealt with one level set.

Our method directly affects the velocity field \( \vec{u} \) which determines the behavior of the fluid. Changing the velocity field \( \vec{u} \) means that the posterior processing, viscosity and pressure will lead to different results, and optimistically better results. To know where to update the velocity field \( \vec{u} \) we make use of a threshold variable \( T \) and the distance function \( \phi \) to control how much of the velocity field will be updated in each iteration of the solver.

In contrast with our method, real surface tension makes the pressure to be different at each point of the fluid which leads to big increment in computation time. In our approximation that is not the case because we consider the pressure to be uniform all over the fluid. That intrinsically means that we only have experimented with incompressible fluids, constant pressure. This technique of using the body forces block to modify the behavior of the fluid has been previously used to modify the behavior of the fluid (e.g., [9]).

5 RESULTS

For our test we are using an Intel® Core™ i3-3220 CPU @ 3.30GHz processor with 4GB of RAM memory, the environment runs a Linux-Based Operating System. We run our program using OpenMP where applicable, however, the surface tension computation runs sequentially.

For our simulations we are using one cubic meter as the domain of the problem. Our examples run in a rather coarse grid which helps to visualize better our results, the resolution of the grid is 70x70x70.
5.1 Example

The presented example shows how the inclusion of surface tension using the new proposed method leads to the shivering behavior of the fluid. This example is a falling body of water, initially configured as an analytic sphere with radius equals to 10cm, that water body falls from 50cm altitude. In this case we are using water properties for the fluid. Viscosity is configured to be $0.001 \text{ N} \cdot \text{s} / \text{m}^2$, and surface tension factor equals to $0.073 \text{ J} / \text{m}^2$. The difference in the obtained surface after several frames of simulations and after the fluid body have hit the ground can be seen in figure 2, for our example the ground is modeled as a solid and we need to handle the respective boundary conditions.

Figure 3 shows the silhouettes of a cross section cut of the surfaces with the view aligned with the Z axis where the difference can be better appreciated as the silhouettes of the fluid. These silhouettes allow us to notices that in regions where there is a plain aspect in the original simulation our method can generate low-scale irregularities due to the new force that resemble a shivering-like behavior. This behavior enhances the result of the falling water.

For rendering purposes we use the marching cubes algorithm to obtain a mesh of triangles. The final, ray-traced, results of the two simulations is presented in figure 4. In this image we can see how the water presents a shivering effect visible in various settings.

Refer to figure 5 for a comparison of the execution times between the simulation with surface tension and without surface tension, the time measured considers the generation of seventy frames at different grid resolutions. Even though the execution time is considerably big, it is mainly due to the data structure we are using, the dense MAC grid. In this case a basic dense grid stores all the information, but it potentially could use other data structure to accelerate its execution.

5.2 Limitations

For surface tension computation the Young equation and the logic behind contact angles is very important because it can generate...
Figure 3: Silhouettes comparison, the silhouette on the top is taken from a simulation without surface tension. The one in the bottom is a simulation using our algorithm. In the latter it is possible to notice more low-scale formations.

Figure 4: Ray-Traced results comparison. Top: without surface tension. Bottom: with surface tension.

very interesting results like drop formation and capillarity, in our presented results we do not consider this logic. The data structure we are using, dense MAC grid, leads to very high computational time.

6 CONCLUSION

Using the proposed method in this paper we have been able to replicate a shivering effect of water using an approximation of surface tension. Therefore, the simulation has improved in quality and realism.

The time increment is proportional to the time obtained without surface tension, that means that our method can be safely used without any considerable impact in the performance of the simulation.

Figure 5: Execution time comparison

Here we have compared visual results and execution times with favorable outcome.

Even though other methods for surface tension exists, ours is unique because it does not work with triangle meshes, it is a fast approximation, and avoid the complexity of dealing with the variation in pressure.

In the future we would like to investigate how our proposed method behaves in different configuration like different data structures, inviscid and compressible fluids.

7 REFERENCES


