Université de Liège

MATH0471-A-a : Multiphysics integrated computational project

Deadline 3 : surface tension, adhesion and viscosity effects; first tests of microfluidics applications.

KOLLASCH Killian s202793
SANTORO Luca s201807

Professor: C. Geuzaine
Professor: R. BOMAN

Academic year : 2023-2024

I Surface tension

I.1 Microscopic approach

It has been proposed to implement the particle acceleration (due to cohesion) by:

$$\frac{D\boldsymbol{v}_i}{Dt} = -\frac{\alpha}{m_i}\sum_j m_j(\boldsymbol{x}_i - \boldsymbol{x}_j)W_{ij}, \tag{I.1}$$

where α is a coefficient to control the surface tension of the fluid and W_{ij} is the kernel used.

Nevertheless, this microscopic approach did not provide pertinent results to be used in further simulation. One thus brings our attention onto another method.

I.2 Macroscopic approach

A much better approach is to consider both the cohesive forces and forces to minimize the surface area.

__**Cohesion**__

First, this new cohesion force involves both the mass of the prescribed particles and its associated neighbours. Another parameter α is used here but is completely different from the α involves in other equations, it will be discussed later:

$$\boldsymbol{F}_i^{\text{cohesion}} = -\alpha\sum_j m_i m_j \frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{||\boldsymbol{x}_i - \boldsymbol{x}_j||} W^{cohesion}(||\boldsymbol{x}_i - \boldsymbol{x}_j||). \tag{I.2}$$

where j refers to the neighbouring index.

Also, W^{cohesion} is another kernel used only on the *surface tension* scope and is defined as:

$$W^{\text{cohesion}}(r) = \frac{32}{\pi h^9}\begin{cases} (h-r)^3 r^3 & \text{if } \frac{h}{2} < r < h, \\ 2(h-r)^3 r^3 - \frac{h^6}{64} & \text{if } r \leq \frac{h}{2}. \end{cases} \tag{I.3}$$

In contrast to the previous model, cohesive forces can become positive and negative. In this way repulsion forces for close particles are generated which prevents undesired particle clustering at the free surface.

Also, one computes a force in order to minimize the surface area. This additional force counteracts the surface curvature, which requires the computation of the surface normals. The normals can be determined by a so-called weighted normal (influenced by its neighbours):

$$\boldsymbol{n}_i = h\sum_j \frac{m_j}{\rho_j}\nabla W_{ij}. \tag{I.4}$$

The magnitude of the resulting vector is close to zero in the interior of the fluid and proportional to the curvature at the free surface. Hence, a symmetric force that counteracts the curvature can be defined as:

$$\boldsymbol{F}_i^{\text{curv}} = -\alpha\sum_j m_i(\boldsymbol{n}_i - \boldsymbol{n}_j). \tag{I.5}$$

Eventually, the resulting force is written as:

$$\boldsymbol{F}_i^{\text{surface tension}} = K_{ij}\left(\boldsymbol{F}_i^{\text{cohesion}} + \boldsymbol{F}_i^{\text{curv}}\right), \tag{I.6}$$

where K_{ij} is a symmetric factor that amplifies the surface tension forces at the free surface. At the surface ρ_i and ρ_j are underestimated due to particle deficiency and $K_{ij} > 1$ while for a particle with a full neighbourhood $K_{ij} \approx 1$.

I.3 Analysis

One did not find any clues about the value of the parameter α. Thus, several tests have been performed to find the right adjustment, Looking Eq (I.2) allows us to forecast the behaviour of the simulations. A greater value of α induces a higher impact onto the "forming sphere".

Indeed, using the following geometric parameters:

- $L_d = [2, 2, 2]$,
- $O_d = [0, 0, 0]$,
- $matrixLong = [[1, 1, 1]]$,
- $matrixOrig = [[0.5, 0.5, 0.5]]$,
- $dt = 0.001$.

where L_d, O_d are the length and the origin of the whole domain of computation and $matrixLong$ and $matrixOrig$ are the lengths and origins of all other fixed or moving cubes particles (here, a single cube of moving particle).

Given that the gravity is not involved in the following simulations, (only the surface tensions) one does expect the cube ("in the void") to become a sphere at the end of the process.

Using $s = 0.05$ and $\alpha = 10$ yields to the following results (Fig. 5):

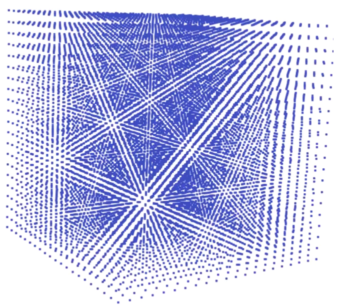

Figure 1: Surface tensions applied to cube, ($s = 0.05, \alpha = 10$) at $timestep = 0$.

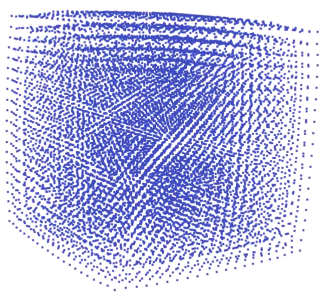

Figure 2: Surface tensions applied to cube, ($s = 0.05, \alpha = 10$) at $timestep = 100$.

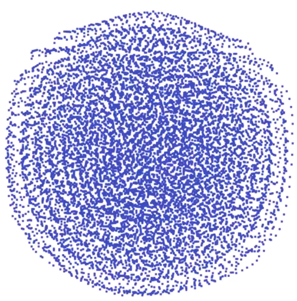

Figure 3: Surface tensions applied to cube, ($s = 0.05, \alpha = 10$) at $timestep = 400$.

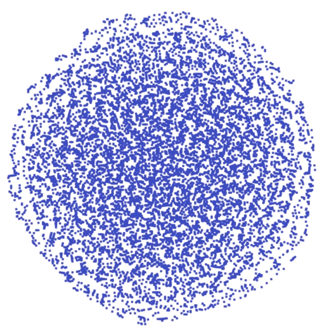

Figure 4: Surface tensions applied to cube, ($s = 0.05, \alpha = 10$) at $timestep = 1200$.

Figure 5: Surface tensions applied to cube, ($s = 0.05, \alpha = 10$).

This set of parameter provides pretty good results and leads to the expected behaviour. One may conclude that the *surface tension* implementation (at least in this configuration) works correctly. But one will now discuss the limitation of the method and its associated parameters.

Using $s = 0.1$ and $\alpha = 10$ yields to the following results(Fig 9):

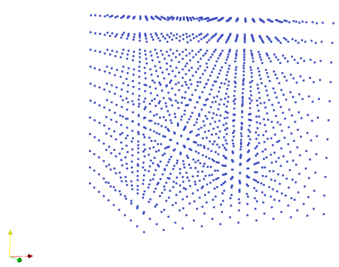

Figure 6: Surface tensions applied to cube, $(s = 0.1, \alpha = 10)$ at $timestep = 0$.

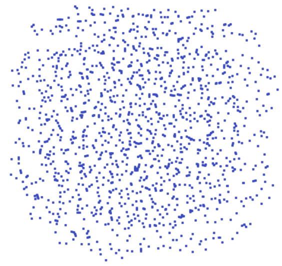

Figure 7: Surface tensions applied to cube, $(s = 0.1, \alpha = 10)$ at $timestep = 100$.

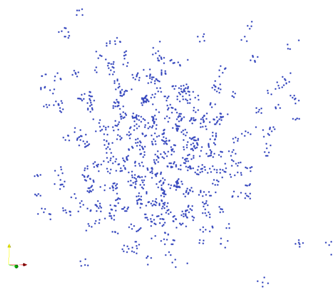

Figure 8: Surface tensions applied to cube, $(s = 0.1, \alpha = 10)$ at $timestep = 400$.

Figure 9: Surface tensions applied to cube, $(s = 0.1, \alpha = 10)$.

One can notice that the cube does indeed turn into a sphere, but the "inertia" of the boundary particles is so high that some of them go away. It seems that the boundary particles are not attracted enough to the centre of the sphere (by the neighbouring particles), resulting in a "spreading process".

Using now $s = 0.1$ and $\alpha = 0.8$ leads to (Fig. 14):

Indeed, a smaller α tends to decrease the "inertia" of the influenced boundary particles, but it seems like the boundary particles (after some time) still tend to go away.

A plausible explanation of this relationship between s and α involves the number of neighbours for each boundary particles. Indeed, reducing the particle spacing s leads to a higher number of particles, but one keeps in mind that the *neighbouring area* is also accordingly reduces. Nevertheless, our explanation of the phenomenon is that for small number of particles, the *surface tension* forces should not be too high, because the attractive force (induced by the *neighbouring*) has to be greater than the repulsive force. But this particular phenomenon has to be studied in more details.

II Adhesion forces

To simulate the attractive forces between fluid particles and the boundary, an adhesion force is introduced as:

$$\boldsymbol{F}_i^{\text{adhesion}} = -\beta \sum_k m_i m_k \frac{\boldsymbol{x}_i - \boldsymbol{x}_k}{||\boldsymbol{x}_i - \boldsymbol{x}_k||} W^{\text{adhesion}}(||\boldsymbol{x}_i - \boldsymbol{x}_k||). \tag{II.1}$$

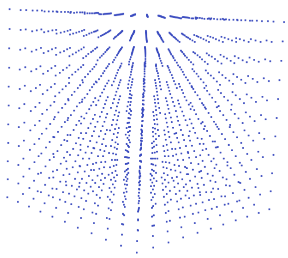

Figure 10: Surface tensions applied to cube, $(s = 0.1, \alpha = 0.8)$ at $timestep = 0$.

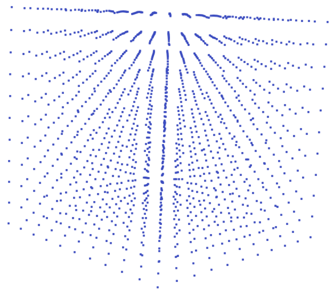

Figure 11: Surface tensions applied to cube, $(s = 0.1, \alpha = 0.8)$ at $timestep = 100$.

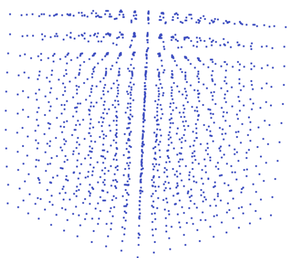

Figure 12: Surface tensions applied to cube, $(s = 0.1, \alpha = 0.8)$ at $timestep = 400$.

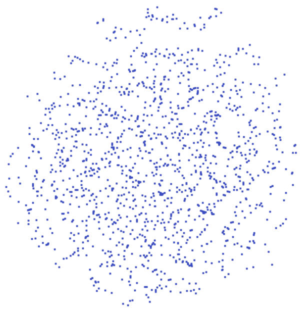

Figure 13: Surface tensions applied to cube, $(s = 0.1, \alpha = 0.8)$ at $timestep = 1800$.

Figure 14: Surface tensions applied to cube, $(s = 0.1, \alpha = 0.8)$.

where β is the adhesion coefficient and k denotes a neighbouring boundary particle.

Also, a specialized kernel is used to mimic the adhesion influence:

$$W^{\text{adhesion}}(r) = \frac{0.007}{h^{3.25}} \begin{cases} \sqrt[4]{-\frac{4r2}{h} + 6r - 2h} & \text{if } \frac{h}{2} < r < h, \\ 0 & \text{otherwise.} \end{cases} \tag{II.2}$$

But this method has not been analyzed yet.

III What is next ?

Some elements of the code should be enhanced, for example there is a list of point that should be taken into account in the future and what should be improved:

- Taking into account adhesion forces.
- Taking into account the external force in the function checkTimeStep. Up to now, only the gravity is involved in the function.
- 2D version of the SPH algorithm to fasten the analysis.
- Reviewing literature in order to simulate more complex microfluidic behaviours.
- Studying the speed earns by using OpenMP.