LMECA2300 – Project
Fluid animation with the SPH method

The seminars have introduced you to the *Smoothed Particle Hydrodynamics* method. This Lagrangian method provides realistic fluid animations in a short time. Your main task is to make a fluid animation performed by the SPH method.

Basic requirements

Your animation has to involve movement of a fluid. In orther words, most of the particles describing your fluid should move according to some physical laws. For example, a lid driven cavity is the *kind of physics* which is expected.

The animation has to be built from the **BOV library**. A **movie of 5 minutes maximum** showing your fluid animation has to be uploaded on `YouTube`. The movie may give details about the animation, with either subtitles or voice off.

The implementation yielding the fluid animation has to be written in `C`. Sources of the implementation have to be available on a **git repository**. The implementation should be compiled using `cmake` (preferably) or (at least) a `Makefile`. A *readme* would be appreciated, specially if the implementtation provides some options at run.

A **report of 5 pages** maximum giving interesting details about the fluid animation has to be sent to the teaching team. The report should focus on *original* description of the represented physics, *innovative* discrete schemes and *groundbreaking features* of the implementation.

Completing the basic requirements allows you to get a grade up to 14/20.

Upgrades

In order to get a higher grade, *only one* proposed improvement has to be *fully* realized:

- **Complex physics**

  The represented physics is challenging (Neumann condition, multiphase fluid, ...). There are solid wall(s) *and* free surface(s). A sensitivity analysis of all the parameters governing the fluid dynamics is performed. The implementation has to be *stable*; the animation should *not* explode, *whatever* the duration of the animation.

- **Great animation with shaders**

  The animation has to be *fully interactive*. The window (where the animation runs) has to be refreshed *as often as* the screen/monitor does. Shaders should be used to get a *great rendering* of the scene (i.e. animation). It shoud be possible to browse among each feature (pressure, density, ...) of the particles, by *displaying each feature as a continuous field*. `BOV` library should *not* be modified.

- **Parallel implementation**

  The implementation is parallelized *as much as possible* using `MPI`(preferably), or `OpenMP`. The speedup is *real* and *exhibited* (a table giving raw data of CPU time versus number of processors/threads is expected). A suitable identification of which parts *are* embarrasingly parallel and which ones are *not* is performed. Parallelization of the *non*-embarrasingly parallel parts is explained in details.