Chapter 1

General problem

1.1 Introduction

In this section we give details on our choices for the generic application confronted to both computation and communication walls in irregular context. This problem, the Smoothed Particle Hydrodynamics method implementation, is described on a physics aspect. We point out the difficulties involved in the resolution on supercomputers and more especially accelerators. A lot of work have been done on the comprehension of the physical aspect to produce a code that meet the behavior of real physics problems.

The first section is a presentation of the SPH method itself and the overall limitation we have to face. Then we describe different kind of problems we use as a benchmark of the application itself.

1.2 Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics (SPH) is an explicit numerical mesh-free Lagrangian method. It is used to solve hydrodynamical partial differential equations (PDEs) by discretized them into a set of fluid elements called particles. This computational method was invented for the purpose of astrophysics simulations by Monaghan, Gingold and Lucy in 1977 [Luc77, GM77]. This first SPH work conserved mass and they later proposed a method which also conserves linear and angular moment [GM82]. The method was extended for general fluid simulation and many more fields from ballistics to oceanography. The development of new reliable, parallel and distributed tools for this method is a challenge for future HPC architectures with the upcoming Exascale systems.

1.2.1 General description

The method, as illustrated in figure 1.1, computes the evolution of physical quantities for every particle regarding its neighbors in the radius of its smoothing length h. The particles in this radius are then valued according to their distance using a smoothing function W, also called a kernel. The fundamental SPH formulation for any physical quantity A is then to compute with all the neighbors of b of a a particle by:

$$A(\vec{r})_a \simeq \sum_b \frac{m_b}{\rho_b} A(\vec{r_b}) W(|\vec{r} - \vec{r_b}|, h) \tag{1.1}$$

On a physics aspect, this method has several advantages: it can handle deformations, low densities, vacuum, and makes particle tracking easier. It also conserves mass, linear and angular momenta, and energy by its construction that implies independence of the numerical resolution. Another strong benefit of using SPH is its exact advection of fluid properties. Furthermore, the particle structure of SPH easily combines with tree methods for solving Newtonian gravity

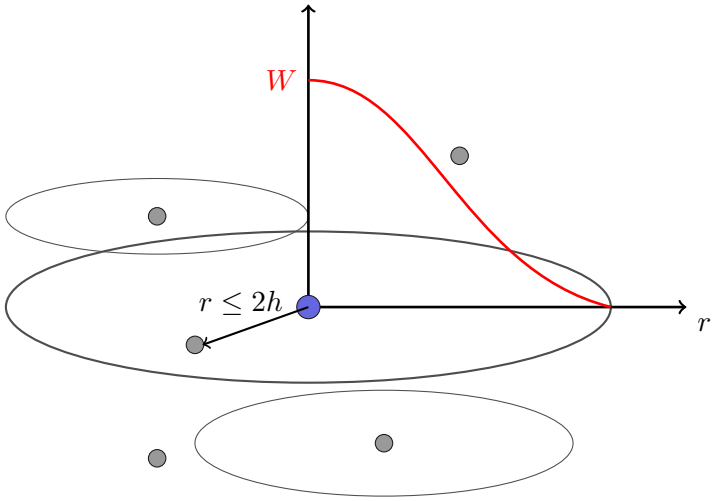


Figure 1.1: SPH kernel W and smoothing length h representation

through N-body simulations. As a mesh-free method, it avoids the need of grid to calculate the spatial derivatives.

However, there are cons to consider using SPH: it cannot be extend to all PDE formulations; it requires careful setup of initial distribution of particles; further, it can be struggle to resolve turbulence-dominated flows and special care must be taken when handling high gradients such as shocks and surface structure of neutron stars. Many works are leading to handle more cases and to push the limitations of this method [DRZR17, LSR16, RDZR16].

In this work, we are solving Lagrangian conservation equations (Euler equations) for density, energy and momentum of an ideal fluid [LL59] such that:

$$\frac{d\rho}{dt} = -\rho(\nabla \cdot \vec{v}), \quad \frac{du}{dt} = -\frac{P}{\rho}(\nabla \cdot \vec{v}), \quad \frac{d\vec{v}}{dt} = -\frac{1}{\rho}(\nabla P) \tag{1.2}$$

with ρ the density, P the pressure, u the internal energy and v the velocity, where $d/dt = \partial/\partial_t + \vec{v} \cdot \nabla$ which is convective derivative.

By using the volume element $V_b = m_b/\rho_b$, we can formulate the Newtonian SPH scheme [Ros09] such that

$$\rho_a = \sum_b m_b W_{ab}(h_a) \tag{1.3}$$

$$\frac{du_a}{dt} = \frac{P_a}{\rho_a^2} \sum_b m_b \vec{v}_{ab} \cdot \nabla_a W_{ab} \tag{1.4}$$

$$\frac{d\vec{v}_a}{dt} = -\sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} \right) \nabla_a W_{ab} \tag{1.5}$$

where $W_{ab} = W(|\vec{r}_a - \vec{r}_b|, h)$ is the smoothing kernel. The equations we would like to solve allow for emergence of discontinuities from smooth initial data. At discontinuities, the entropy increases in shocks. That dissipation occurs inside the shock-front. The SPH formulation here is inviscid so we need to handle this dissipation near shocks. There are a number of way to handle this problem, but the most widespread approach is to add artificial viscosity (or artificial dissipation) terms in SPH formulation such that:

$$\left(\frac{du_a}{dt} \right)_{art} = \frac{1}{2} \sum_b m_b \Pi_{ab} \vec{v}_{ab} \cdot \nabla_a W_{ab} \tag{1.6}$$

$$\left(\frac{d\vec{v}_a}{dt} \right)_{art} = -\sum_b m_b \Pi_{ab} \nabla_a W_{ab} \tag{1.7}$$

In general, we can express the equations for internal energy and acceleration with artificial viscosity

$$\frac{du_a}{dt} = \sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{\Pi_{ab}}{2} \right) \vec{v}_{ab} \cdot \nabla_a W_{ab} \tag{1.8}$$

$$\frac{d\vec{v}_a}{dt} = -\sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} + \Pi_{ab} \right) \nabla_a W_{ab} \tag{1.9}$$

Π_{ab} is the artificial viscosity tensor. As long as Π_{ab} is symmetric, the conservation of energy, linear and angular momentum is assured by the form of the equation and antisymmetry of the gradient of kernel with respect to the exchange of indices a and b. Π_{ab} may define in different ways and here we use [MG83] such as:

$$\Pi_{ab} = \begin{cases} \frac{-\alpha \bar{c}_{ab} \mu_{ab} + \beta \mu_{ab}^2}{\bar{\rho}_{ab}} & \text{for } \vec{r}_{ab} \cdot \vec{v}_{ab} < 0 \\ 0 & \text{otherwise} \end{cases} \tag{1.10}$$

$$\mu_{ab} = \frac{\bar{h}_{ab} \vec{r}_{ab} \cdot \vec{v}_{ab}}{r_{ab}^2 + \epsilon \bar{h}_{ab}^2} \tag{1.11}$$

Using the usual form c_s as $c_s = \sqrt{\frac{\partial p}{\partial \rho}}$. The values of ϵ, α, and β have to be set regarding the problem targeted. As an example we used for the Sod shock tube problem: $\epsilon = 0.01h^2$, $\alpha = 1.0$, and $\beta = 2.0$.

There are many possibilities for the smoothing function, called the kernel. As an example the Monaghan's cubic spline kernel is given by:

$$W(\vec{r_{ij}}, h) = \frac{\sigma}{h^D} \begin{cases} 1 - \frac{3}{2}q^2 + \frac{3}{4}q^3 & \text{if} \quad 0 \leq q \leq 1 \\ \frac{1}{4}(2-q)^3 & \text{if} \quad 1 \leq q \leq 2 \\ 0 & \text{otherwise} \end{cases} \tag{1.12}$$

where $q = r/h$, r the distance between the two particles, D is the number of dimensions and σ is a normalization constant with the values:

$$\sigma = \begin{cases} \frac{2}{3} & \text{for 1D} \\ \frac{10}{7\pi} & \text{for 2D} \\ \frac{1}{\pi} & \text{for 3D} \end{cases} \tag{1.13}$$

In the computation of forces we also need to apply the gradient of the smoothing kernel. This is, in our example, for the cubic spline kernel:

$$\vec{\nabla}_i W(\vec{r_{ij}}, h) = \frac{\sigma}{h^{D+1}} \times \begin{cases} (-\frac{3}{h} + \frac{9}{4h}q)\vec{r_{ij}}, & \text{si } 0 \leq \frac{r}{h} < 1 \\ (\frac{-3}{r} + \frac{3}{h} - \frac{3}{4h}q)\vec{r_{ij}}, & \text{si } 1 \leq \frac{r}{h} < 2 \\ 0, & \text{si } \frac{r}{h} \geq 2 \end{cases} \tag{1.14}$$

Figure 1.2 is a representation of the cubic spline kernel with $\sigma = 1$ and $h = 1$. The abscissa axis represent r, distance between the particles and ordinate the value of the smoothing kernel. When the support of the function, 2 is reached the particles are ignored $W = 0$, represented in red on the figure.

To sum up, the SPH resolution scheme and its routines are presented on algorithm 1. The Equation of State (EOS) and the integration are problem dependent and will be define for each test case in section 1.3.

The main downside for the implementation of this method is the requirement for local computation on every particle. The particles have to be grouped locally to perform the computation of (1.3), (1.8) and (1.9). A communication step is needed before and after (1.3) to get the local physical data to be able to compute (1.8) and (1.9). The tree data structure allows us to perform $O(Nlog(N))$ neighbor search but also add a domain decomposition and distribution layer.

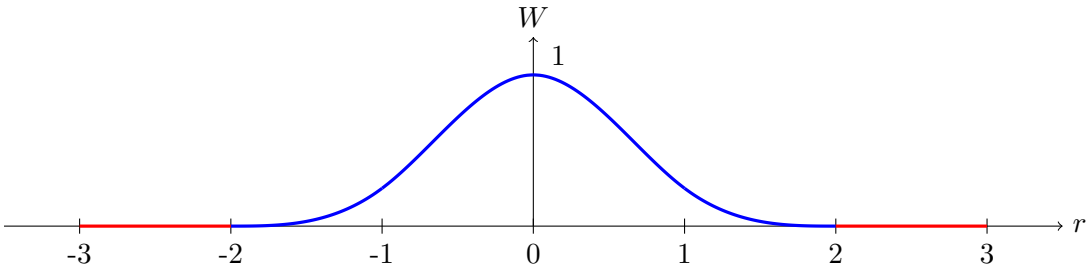


Figure 1.2: Cubic spline kernel example with $\sigma = 1$ and $h = 1$

Algorithm 1 SPH loop algorithm

```
while not last step do
    Compute density for each particle (1.3)
    Compute pressure using EOS
    Compute acceleration from pressure forces (1.9)
    Compute change of internal energy for acceleration (1.8)
    Advance particles after integration
end while
```

1.2.2 Gravitation

For classical problems like fluid flow the gravitation can directly be applied on the particles with the force:

$$\vec{a_g} = m\vec{g} \tag{1.15}$$

In order to consider astrophysics problems we need to introduce self-gravitation and gravitation. Each particle imply an action on the others base on its distance and mass. The equation of gravitation for a particle i with j other particles is:

$$\vec{f}_{ai} = \sum_j -G\frac{m_i m_j}{|\vec{r_i} - \vec{r_j}|^3}\vec{r_{ij}} \tag{1.16}$$

This computation involve an $O(N^2)$ complexity and thus is not applicable directly. We applied the method called Fast Multipole Method, FMM and discussed in [BG97]. This method is perfectly adapted to our tree representation of the domain and particles.

This method aim to compute the gravitation up to an approximation determined by the user.

This method is based on Taylor series. The gravitation function of equation 1.16 can be approximate on a particle at position $\vec{r}$ by the gravitation computed at the centroid at position $\vec{r_c}$:

$$\vec{f}(\vec{r}) = \vec{f}(\vec{r_c}) + ||\frac{\partial \vec{f}}{\partial \vec{r}}|| \cdot (\vec{r} - \vec{r_c}) + \frac{1}{2}(\vec{r} - \vec{r_c})^\intercal \cdot ||\frac{\partial \vec{f}}{\partial \vec{r}\partial \vec{r}}|| \cdot (\vec{r} - \vec{r_c}) \tag{1.17}$$

From equation 1.16 we compute the term $||\frac{\partial \vec{f}}{\partial \vec{r}}||$:s

$$\frac{\partial \vec{f}}{\partial \vec{r}} = -\sum_p \frac{m_p}{|\vec{r_c} - \vec{r_p}|^3}\begin{bmatrix} 1 - \frac{3(x_c-x_p)(x_c-x_p)}{|\vec{r_c}-\vec{r_p}|^2} & -\frac{3(y_c-y_p)(x_c-x_p)}{|\vec{r_c}-\vec{r_p}|^2} & -\frac{3(z_c-z_p)(x_c-x_p)}{|\vec{r_c}-\vec{r_p}|^2} \\ -\frac{3(x_c-x_p)(y_c-y_p)}{|\vec{r_c}-\vec{r_p}|^2} & 1 - \frac{3(y_c-y_p)(y_c-y_p)}{|\vec{r_c}-\vec{r_p}|^2} & -\frac{3(z_c-z_p)(y_c-y_p)}{|\vec{r_c}-\vec{r_p}|^2} \\ -\frac{3(x_c-x_p)(z_c-z_p)}{|\vec{r_c}-\vec{r_p}|^2} & -\frac{3(y_c-y_p)(z_c-z_p)}{|\vec{r_c}-\vec{r_p}|^2} & 1 - \frac{3(z_c-z_p)(z_c-z_p)}{|\vec{r_c}-\vec{r_p}|^2} \end{bmatrix} \tag{1.18}$$

And we propose a compact version of the matrix with:

$$||\frac{\partial f^a}{\partial r^b}|| = -\sum_c \frac{m_c}{|\vec{r} - \vec{r_c}|^3}\left[\delta_{ab} - \frac{3.(r^a - r_c^a)(r^b - r_c^b)}{|\vec{r} - \vec{r_c}|^2}\right] \tag{1.19}$$

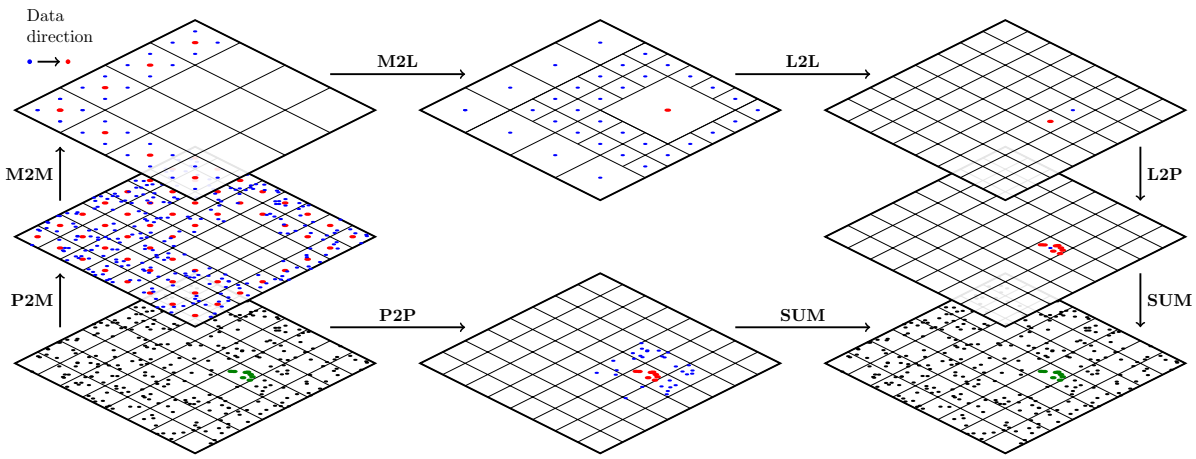


Figure 1.3: Fast Multipole Method schematics. Particles to Multipole (P2M), Multipole to Multipole (M2M), Multipole to Particles (M2P), Multipole to Local (M2L), Local to Local (L2L) and Particles to Particles (P2P). Schematic inspired from [YB11]

With δ_{ab} the Kronecker delta:

$$\delta_{ab} = \begin{cases} 1, & \text{if } a = b. \\ 0, & \text{if } a \neq b. \end{cases} \tag{1.20}$$

We note that a and b variate from 0 to 2 and $r^0 = x$, $r^1 = y$, and $r^2 = z$ as usual sense. For the term $||\frac{\partial \vec{f}}{\partial \vec{r} \partial \vec{r}}||$ we give the compact version by:

$$\begin{aligned} ||\frac{\partial^2 f^a}{\partial r^b \partial r^c}|| = -\sum_c \frac{3m_c}{|\vec{r} - \vec{r_c}|^5} \Big[\frac{5(r^a - r_c^a)(r^b - r_c^b)(r^c - r_c^c)}{|\vec{r} - \vec{r_c}|^2} - \\ \Big(\delta_{ab}(r^c - r_c^c) + \delta_{bc}(r^a - r_c^a) + \delta_{ac}(r^b - r_c^b) \Big) \Big] \end{aligned} \tag{1.21}$$

The method is summed up in figure with the different equations. We consider Centers Of Mass, COM, to be the centroid of particles based on their position. In several steps the information is first transmitted to the COMs, computing their position and mass.

1.3 Applications of SPH

The previous equations are generic and describe the behavior of SPH method. In order to check our

1.3.1 Sod shock tube

The Sod shock tube is the test consisting of a one-dimensional Riemann problem with the following initial parameters [Sod78].

$$(\rho, v, p)_{t=0} = \begin{cases} (1.0, 0.0, 1.0) & \text{if} \quad 0 < x \leq 0.5 \\ (0.125, 0.0, 0.1) & \text{if} \quad 0.5 < x < 1.0 \end{cases} \tag{1.22}$$

In our code, we use the same initial data as in section **??** with ideal gas EOS such as:

$$P(\rho, u) = (\Gamma - 1)\rho u \tag{1.23}$$

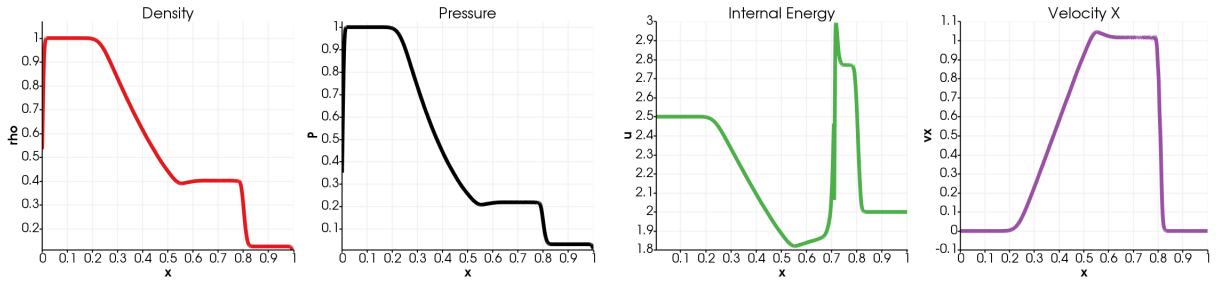


Figure 1.4: Sod shock tube with FleCSPH

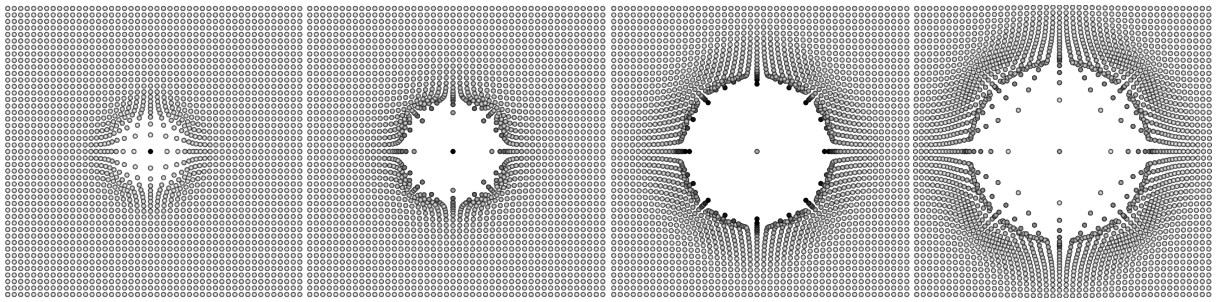


Figure 1.5: Sedov Blast Wave with FleCSPH at respectively $t = 0.01$, $t = 0.03$, $t = 0.06$ and $t = 0.1$

where Γ is the adiabatic index of the gas, we set $\Gamma = 5/3$.

This test is used to check the physical accuracy of the code and thus the tree search itself. A simulation of our Sod shock experimentation is presented on Fig. 1.4 and shows physically correct results.

1.3.2 Sedov blast wave

A blast wave is the pressure and flow resulting from the deposition of a large amount of energy in a small very localized volume. There are different versions of blast wave test and we consider comparing it with the analytic solution for a point explosion as given by Sedov [Sed46], making the assumption that the atmospheric pressure relative to the pressure insider the explosion negligible. Here, we test 2D blast wave. In this simulation, we use ideal gas EOS with $\Gamma = 5/3$ and we are assuming that the undistributed area is at rest with a pressure $P_0 = 1.0^{-5}$. The density is constant ρ_0, also in the pressurized region.

An example of our Sedov Blast wave experimentation is presented on Fig. 1.5 and shows physically correct results.

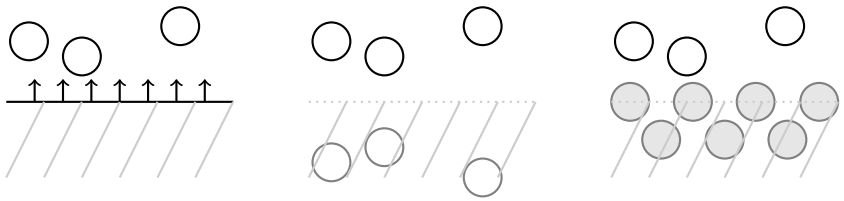
1.3.3 Fluid flow

After performing the tests regarding the physics reliability, we worked on fluid flow problem in 2D and 3D to reach high number of particles. The details can be found in [GGRC+12]. This test is based on an ideal EOS given by:

$$P = B\left[\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right] \tag{1.24}$$

with $\gamma = 7$ and $B = c_0\rho_0/\gamma$ being $\rho_0 = 1000\, kg.m^{-3}$ the reference density.

For this experiment, realistic boundaries conditions were needed. Several methods are possible with SPH we focused on the main ones, the repulsive wall, the mirror particles [LP91] and the dummies particles implementation [AHA12]. Those boundaries conditions implementation are presented in figure 1.6.



Repulsive force Mirror particles Dummy particles

Figure 1.6: Different boundaries condition methods

For the current implementation, we used the dummies particles method. The wall particles are just considered as normal particles, with specific equations, and their quantities are evolved during the run. The main difference is that their position does not evolve at the end of the step. They are identified in the code with a specific type, provided during the data generation.

1.3.4 Astrophysics: neutron stars coalescence

The final aim of our tests is to simulate astrophysical events. We are interested in one of the most important event recently discovered. Last year the Laser Interferometer Gravitational Wave, LIGO, detected the first gravitational wave generated by binary neutron stars merging [AAA+17b] and also more complexes event with Binary Black Holes coalescence in [AAA+17a].

Solving Lane-Emden Equation

We need to determine the density function based on the radius.

As we consider the star as a polytropic fluid, we use the equation of Lane-Emden which is a form of the Poisson equation:

$$\frac{d^2\theta}{d\xi^2} + \frac{2}{\xi}\frac{d\theta}{d\xi} + \theta^n = 0 \tag{1.25}$$

With ξ and θ two dimensionless variables. There is only exact solutions for a polytropic index $n = 0.5$, 1 and 2. In our work we use a polytropic index of 1 which can correspond to a NS simulation.

For $n = 1$ the solution of equation 1.25 is:

$$\theta(\xi) = \frac{sin(\xi)}{\xi} \tag{1.26}$$

We note $\xi_1 = \pi$, the first value of ξ as $\theta(\xi) = 0$. $\theta(\xi)$ is also defined as:

$$\theta(\xi) = \left(\frac{\rho(\xi)}{\rho_c}\right)^{\frac{1}{n}} = \frac{\rho(\xi)}{\rho_c} \tag{1.27}$$

With ρ_c the internal density of the star and ρ the density at a determined radius. ξ is defined as:

$$\xi = Ar = \sqrt{\frac{4\pi G}{K(n+1)}\rho_c^{(n-1)/n}} \times r = \sqrt{\frac{2\pi G}{K}} \times r \text{ (for } n = 1)$$

With K a proportionality constant.

From the previous equations we can write the stellar radius R as:

$$R = \sqrt{\frac{K(n+1)}{4\pi G}}\rho_c^{(1-n)/2}\xi_1 = \sqrt{\frac{K}{2\pi G}} \times \xi_1 \tag{1.28}$$

(We note that for $n = 1$ the radius does not depend of the central density.)

If, for example, we use dimensionless units as $G = R = M = 1$ (for the other results we use CGS with $G = 6.674 \times 10^{-8} cm^3 g^{-1} s^{-2}$) We can compute K as:

$$K = \frac{R^2 2\pi G}{\xi_1^2} \tag{1.29}$$

	NS_1	NS_2	NS_3	NS_4
Radius (cm)	$R = G = M = 1$	1500000	1400000	960000
K	0.636619	95598.00	83576.48	39156.94

Then we deduce the density function of r as :

$$\rho(\xi) = \frac{sin(A \times r)}{A \times r} \times \rho_c \text{ with } A = \sqrt{\frac{2\pi G}{K}}$$

As we know the total Mass M, the radius R and the gravitational constant G we can compute the central density as:

$$\rho_c = \frac{MA^3}{4\pi(sin(AR) - ARcos(AR))}$$

Then we normalize the results to fit $R = M = G = 1$: $K' = K/(R^2 G)$, $m_i' = m_i/M$, $h_i' = h_i/R$, $\vec{x_i}' = \vec{x_i}/R$

1.4 Conclusion

As the SPH method is used in a large panel of fields from astrophysics to fluid mechanic, there are numerous related works. We can cite a code developed in the LANL, 2HOT [War13] that introduced the Hashed Oct Tree structure used in our implementation. There is also GADGET-2 [Spr05], GIZMO [Hop14] and the most recent publication is GASOLINE [WKQ17] based on PKDGRAV, a specific tree+gravity implementation. Several implementations already implement GPU code and tree construction and traversal, one can cite GOTHIC [MU17], presenting gravitational tree code accelerated using the latest Fermi, Kepler and Maxwell architectures. But a lot of GPU accelerated work still focused on fluid problems and not on astrophysical problems [HKK07, CDB+11]. We also note that these implementations focus on SPH problems and does not provide a general purpose and multi-physics framework like we intent to provide through FleCSPH and FleCSI.

Bibliography

[AAA+17a] Benjamin P Abbott, R Abbott, TD Abbott, F Acernese, K Ackley, C Adams, T Adams, P Addesso, RX Adhikari, VB Adya, et al. Gw170814: A three-detector observation of gravitational waves from a binary black hole coalescence. *Physical review letters*, 119(14):141101, 2017.

[AAA+17b] Benjamin P Abbott, Rich Abbott, TD Abbott, Fausto Acernese, Kendall Ackley, Carl Adams, Thomas Adams, Paolo Addesso, RX Adhikari, VB Adya, et al. Gw170817: observation of gravitational waves from a binary neutron star inspiral. *Physical Review Letters*, 119(16):161101, 2017.

[AHA12] S Adami, XY Hu, and NA Adams. A generalized wall boundary condition for smoothed particle hydrodynamics. *Journal of Computational Physics*, 231(21):7057–7075, 2012.

[BG97] Rick Beatson and Leslie Greengard. A short course on fast multipole methods. *Wavelets, multilevel methods and elliptic PDEs*, 1:1–37, 1997.

[CDB+11] Alejandro C Crespo, Jose M Dominguez, Anxo Barreiro, Moncho Gómez-Gesteira, and Benedict D Rogers. Gpus, a new tool of acceleration in cfd: efficiency and reliability on smoothed particle hydrodynamics methods. *PloS one*, 6(6):e20685, 2011.

[DRZR17] Zili Dai, Huilong Ren, Xiaoying Zhuang, and Timon Rabczuk. Dual-support smoothed particle hydrodynamics for elastic mechanics. *International Journal of Computational Methods*, 14(04):1750039, 2017.

[GGRC+12] Moncho Gomez-Gesteira, Benedict D Rogers, Alejandro JC Crespo, Robert A Dalrymple, Muthukumar Narayanaswamy, and José M Dominguez. Sphysics–development of a free-surface fluid solver–part 1: Theory and formulations. *Computers & Geosciences*, 48:289–299, 2012.

[GM77] Robert A Gingold and Joseph J Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly notices of the royal astronomical society*, 181(3):375–389, 1977.

[GM82] RA Gingold and JJ Monaghan. Kernel estimates as a basis for general particle methods in hydrodynamics. *Journal of Computational Physics*, 46(3):429–453, 1982.

[HKK07] Takahiro Harada, Seiichi Koshizuka, and Yoichiro Kawaguchi. Smoothed particle hydrodynamics on gpus. In *Computer Graphics International*, pages 63–70. SBC Petropolis, 2007.

[Hop14] Philip F Hopkins. Gizmo: Multi-method magneto-hydrodynamics+ gravity code. *Astrophysics Source Code Library*, 2014.

[LL59] L. D. Landau and E. M. Lifshitz. *Fluid mechanics*. 1959.

[LP91] Larry D Libersky and AG Petschek. Smooth particle hydrodynamics with strength of materials. In *Advances in the free-Lagrange method including contributions on adaptive gridding and the smooth particle hydrodynamics method*, pages 248–257. Springer, 1991.

[LSR16] SJ Lind, PK Stansby, and Benedict D Rogers. Incompressible–compressible flows with a transient discontinuous interface using smoothed particle hydrodynamics (sph). *Journal of Computational Physics*, 309:129–147, 2016.

[Luc77] Leon B Lucy. A numerical approach to the testing of the fission hypothesis. *The astronomical journal*, 82:1013–1024, 1977.

[MG83] J.J Monaghan and R.A Gingold. Shock simulation by the particle method sph. *Journal of Computational Physics*, 52(2):374 – 389, 1983.

[MU17] Yohei Miki and Masayuki Umemura. Gothic: Gravitational oct-tree code accelerated by hierarchical time step controlling. *New Astronomy*, 52:65–81, 2017.

[RDZR16] Huilong Ren, Zili Dai, Xiaoying Zhuang, and Timon Rabczuk. Dual-support smoothed particle hydrodynamics. *arXiv preprint arXiv:1607.08350*, 2016.

[Ros09] Stephan Rosswog. Astrophysical smooth particle hydrodynamics. *New Astronomy Reviews*, 53(4):78 – 104, 2009.

[Sed46] Leonid I Sedov. Propagation of strong shock waves. *Journal of Applied Mathematics and Mechanics*, 10:241–250, 1946.

[Sod78] Gary A Sod. A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. *Journal of Computational Physics*, 27(1):1 – 31, 1978.

[Spr05] Volker Springel. The cosmological simulation code gadget-2. *Monthly Notices of the Royal Astronomical Society*, 364(4):1105–1134, 2005.

[War13] Michael S Warren. 2hot: an improved parallel hashed oct-tree n-body algorithm for cosmological simulation. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, page 72. ACM, 2013.

[WKQ17] James W Wadsley, Benjamin W Keller, and Thomas R Quinn. Gasoline2: a modern smoothed particle hydrodynamics code. *Monthly Notices of the Royal Astronomical Society*, 471(2):2357–2369, 2017.

[YB11] Rio Yokota and Lorena A Barba. Treecode and fast multipole method for n-body simulation with cuda. In *GPU Computing Gems Emerald Edition*, pages 113–132. Elsevier, 2011.