An efficient interface capturing method for a large collection of interacting cells immersed in a fluid

Meriem Jedouaa, Charles-Henri Bruneau, Emmanuel Maitre

An efficient interface capturing method for a large collection of interacting cells immersed in a fluid

M. Jedouaa[1], C.H. Bruneau[2], E. Maitre[1]
1. Laboratoire Jean Kuntzmann, Univ. Grenoble Alpes and CNRS, Grenoble, France
meriem.jedouaa@imag.fr,emmanuel.maitre@imag.fr
2. Univ. Bordeaux IMB, INRIA Bordeaux-Sud-Ouest Team MEMPHIS, CNRS UMR 5251
charles-henri.bruneau@math.u-bordeaux1.fr

Abstract

An efficient method to capture an arbitrary number of fluid/solid or fluid/fluid interfaces in a level-set framework is built, following the ideas introduced for contour capturing in image analysis. Using only three label maps and two distance functions it is possible to get the distance between the closest cells and to apply the collision force whatever the number of cells is. The method is applied to rigid solid bodies in order to compare to the results available in the literature. In that case a global penalization model uses the label maps to follow the solid bodies all together without a separate computation of each body velocity. Consequently, the method is very efficient when dealing with a large number of cells. Numerical simulations are performed in two- and three-dimensions under gravity force.
Keywords: level set method, multiple cells, fluid/structure interaction, collision model.

1 Introduction

Numerical simulations of fluid-structure interaction (FSI) have attracted an increasing interest and several methods have been proposed during the last decades. A popular and wide spread method is the Arbitrary Lagrangian Eulerian approach (ALE) introduced by Donea in 1982 (see [7]) and extensively studied by several teams [15, 13, 17, 27, 33]. The ALE strategy is an hybrid method that combines the Lagrangian and Eulerian descriptions using a mobile non structured grid that follows the normal displacement of the fluid/structure interface. The fluid and solid equations are solved individually and continuity conditions for the velocity and stress tensor are explicitly discretized at the interface. The main weakness of the ALE method is its difficulty of implementation, especially when dealing with large displacements in dimension three. In addition the added mass effect [3] has been a long standing difficulty which has been worked around only recently [10, 8, 9, 11, 12]. Moreover, the computational grid has to be remeshed when the elements get too distorted, which could be a very costly procedure in three-dimension. Another method, introduced by Cottet and Maitre in [5, 6], is to use a purely Eulerian formulation for describing the fluid/structure interaction. This approach was inspired by the immersed boundary method of Peskin [26] where the forces at the interface were described in a Lagrangian manner. In the model [5], a level set method is used to capture the interface. The level set method was developed in [24] to treat problems involving interfaces. It is used in many domains because of its several advantages: its implementation is very easy, the topological changes are directly handled and one single level set function can capture an arbitrary number of interfaces. This last property is largely used as it can be a very efficient tool to capture several interfaces, for instance when merging and splitting of interfaces are allowed. The present work aims at dealing with a dense suspension of cells immersed in a fluid. In this kind of application, using one level set is not always sufficient as will be explained thereafter. The other existing level set models for capturing a large number of cells are either computationally expensive or cannot be used to handle collisions. Indeed, in [4] one level set function is required for each cell leading to a huge computational cost although contacts are easily avoided. In [32], a formulation using $\log_2 n$ level set functions to represent n different regions is designed. This model substantially reduces the number of level set functions and can handle very easily complex topologies. This approach is based on the four color theorem. However, it is not able to reconstruct all distances between cells, as a consequence this model is not useful when one wants to deal with several bodies immersed in a fluid. Indeed, when dealing with several cells, the investigation of fluid/structures interaction raises the problem of collisions between the bodies. These collisions can be handled if the distance between cells is known. As confirmed by theoretical works [16], the hydrodynamical forces between two bodies

following a Navier-Stokes flow avoid them to enter in contact at finite time. Numerically, however, it is necessary to have enough discretization points between two interfaces in order to resolve hydro dynamical forces. This lack of discretization points could lead to numerical collision and coalescence of bodies. A first approach proposed in [18] is to refine the mesh near the inter particle gap in order to resolve accurately these lubrication forces. However, this strategy leads to a high computational cost as several refinements are necessary and the frequency of refinements is not known a priori. Consequently, a collision model appears to be necessary to develop numerical simulations with tractable cost at relatively low resolution. The hydrodynamical forces between two smooth bodies is a function of the bodies size and shape, the distance between them, and the sign is associated to the relative velocity of the two cells. Following this idea, B. Maury introduced a first order approximation of these lubrication forces (see [21]). In [14] short range repulsive forces between particles are introduced and tend to avoid contacts. Another approach is to impose a minimal distance between cells to forbid overlaps, this was achieved in [22] by using a minimization procedure on a global functional of the cells position. In [23] a scheme for inelastic collisions is developed allowing to impose a minimal distance between the cells. At last in [4] a repulsion force model is developed in the framework of vortex methods to avoid contact.
In this work a new type of algorithm is designed to enable these contacts efficiently by adding a short range repulsive force. This algorithm is derived from the multi geometric deformable model (MGDM) introduced by J. Bogovic [2] for image segmentation. The proposed algorithm can handle multiple deforming bodies and avoid collision using a short range repulsive force depending on the distance to the closest interface, following [4]. The main advantages of this method is that it requires only five fields and one level set function to capture an arbitrary number of cells and it can, at the same time, deal with collisions. This substantially reduces the computational cost, as will be illustrated below. The level set function captures all interfaces and is transported with the fluid velocity. Then a local fast marching algorithm is performed at each time step to find the the closest neighbours and their associated distance functions. Let us point out that this sorting algorithm depends on the number of cells. Indeed, if we denote by M the number of grid points and N the number of cells then the worst complexity of this sorting algorithm is $\mathcal{O}(NM(log(NM)))$. In the case of spherical rigid structures it is possible to avoid it by advecting the center of each sphere and so another faster approach is employed. This work combines the advantage of the MGDM method which efficiently captures a large number of bodies and their relative neighbours and of the collision model introduced in [4] using a level set decomposition.

This paper is organized as follows: In section 2 is presented the proposed model with a careful description of the three label maps and the two distance functions. In section 3, the forces used to avoid collisions are described. In sections 4 and 5 is shown how to apply the method to the case of rigid bodies immersed in an incompressible fluid. In section 6 is proposed a bench of numerical simulations starting with a qualitative study of the grid convergence on the sedimentation of 25 circular rigid disks in two dimensions. Then, other simulations on circular or spherical rigid bodies are performed and compared to the results of the literature. Finally, some conclusions are derived.

2 The proposed model

In this section, we first recall some basic principles of the level set method. Then, we provide a description of the method used to capture multiple interfaces. This method, inspired by the multi geometric deformable model of J. Bogovic [2], is introduced in the context of several cells immersed in a fluid. The main idea is to partition the entire fluid/structures domain into several objects making a decomposition of the domain. Then a local configuration of these objects is obtained with the first and second neighbours and their associated distance functions.

2.1 Outline of the level set method

Pioneered by Osher and Sethian in [24], the level set method is very popular to treat problems involving interfaces. It is widely used for numerical analysis of surfaces and shapes. The general idea of the level set method is to define a scalar function that assumes a 0 value on the location of the interface to capture.
Let Ω be a bounded domain in $\mathbb{R}^d$ ($d = 2$ or $d = 3$) partitioned into two sub domains Ω_1 and Ω_2 and Γ be the interface between Ω_1 and Ω_2. The aim is to follow the evolution of the interface Γ that is defined as the zero value of a level set function ϕ. At each time t, the interface Γ is characterized by:

$$\Gamma(t) = \{x \in \Omega, \phi(x,t) = 0\}$$

The level set function has to be Lipschitz continuous in the whole domain Ω. It is for example defined as:

$$\begin{cases} \phi(x) < 0 & x \in \Omega_1 \\ \phi(x) = 0 & x \in \Gamma \\ \phi(x) > 0 & x \in \Omega_2 \end{cases}$$

The displacement of the interface is obtained by the evolution of the level set function ϕ. Let u be the velocity in the domain Ω, the level set function is the solution of the scalar transport equation:

$$\partial_t \phi + u \cdot \nabla \phi = 0. \tag{1}$$

The velocity field u can depend on the space, the time, the geometric properties of the curve and/or the physics of the problem. For example, if we consider a problem of fluid/structures interaction this velocity field is the velocity of the fluid or of the structures. We usually define the level set function as a signed distance function that is regular in each corresponding domain.

$$\phi(x) = \begin{cases} -d(x,\Gamma) & x \in \Omega_1 \\ d(x,\Gamma) & x \in \Omega_2 \end{cases} \tag{2}$$

where

$$d(x,\Gamma) = \min_{y \in \Gamma} \|x - y\|.$$

Using an implicit function to capture the interface tells us directly to which region belongs a point x with the help of a Heaviside function H and its corresponding Dirac function ζ applied to $\phi(x)$. In practice, a regularized version of these functions H_ε and ζ_ε is used on the interface in order to reduce gride:

$$H_\varepsilon(\phi(x)) = \begin{cases} 0 & \phi(x) \le -\varepsilon \\ \frac{1}{2}(1 + \frac{\phi(x)}{\varepsilon} + \frac{\sin(\frac{\pi\phi(x)}{\varepsilon})}{\pi}) & |\phi(x)| \le \varepsilon \\ 1 & \phi(x) \ge \varepsilon \end{cases} \quad \zeta_\varepsilon(\phi(x)) = \begin{cases} 0 & \phi(x) \le -\varepsilon \\ \frac{1}{2\varepsilon}(1 + \cos(\frac{\pi\phi(x)}{\varepsilon})) & |\phi(x)| \le \varepsilon \\ 0 & \phi(x) \ge \varepsilon \end{cases}$$

where ε represents half of the interface thickness. While an advantage of the level set method is to handle automatically changes of topology, in our case this property is problematic as splitting or merging of interfaces are directly taken into account by the level set function. Moreover geometrical characteristics of the curve such as normal vectors n and curvature κ are obtained explicitly using the level set function:

$$n = \frac{\nabla \phi}{|\nabla \phi|} \qquad \kappa = \nabla \cdot n.$$

2.2 Level set functions for multiple interfaces

Let us consider N cells Ω_i, $i \in \{1, ..., N\}$ inside the domain Ω such as none intersects with $\Omega_i \in \Omega$ pairewise disjoints and let us denote Γ_i the frontier of the cell Ω_i and Ω_{N_f} the fluid background sub domain with $N_\text{f} = N + 1$. A big advantage of the level set method is that one level set function can capture an arbitrary number of interfaces between the cells and the fluid. Let ϕ be the level set function which captures the union of the N cells Ω_i, $i \in \{1, ..., N\}$ defined by:

$$\phi(x) = \begin{cases} -d(x, \cup_{i=1}^N \Gamma_i) & x \in \cup_{i=1}^N \Omega_i \\ d(x, \cup_{i=1}^N \Gamma_i) & \text{elsewhere} \end{cases}$$

Using one single level set function yields a very low computing time as it is independent on the number of cells. However, it is not possible to specify a different velocity model and/or a force for each cell. If two cells are too close the curvature and the normal are not well computed and a strong drawback is that we do not have any information on the distance between cells. So this method can lead to collision and merging of cells. Another way to capture multiple cells is to use one level set function for each cell. Then each cell interface Γ_i is captured by one level set function ϕ_i. Thus it is possible to specify a different speed or force to a cell Ω_i by using the level set function ϕ_i and we get the distance between all the cells. Indeed, the distance between two cells Ω_j and Ω_i is given by the level set functions ϕ_i and ϕ_j as follows:

$$\forall x \in \Omega_i, \phi_j = d(x, \Gamma_j) \quad \text{or} \quad \forall x \in \Omega_j, \phi_i = d(x, \Gamma_i)$$

This multiple level set decomposition has been widely used in image segmentation and in fluid/structures interaction. A big inconvenient is its computational cost when the number of cells increases.

2.3 Domain decomposition

Let $\Omega \subset \mathbb{R}^d$ ($d = 2$ or 3) be a bounded domain which contains N cells immersed in an incompressible fluid, we denote Ω_i and Γ_i the interior and the boundary of the i^{th} cell. We consider the fluid Ω_{N_f} as an object . Thus the

fluid/structures domain Ω is partitioned into $N+1$ objects as:

$$\begin{cases} \forall i \neq j, \Omega_i \bigcap \Omega_j &= \emptyset \\ \Omega_{N_\mathrm{f}} &= \Omega \backslash \overline{\{\bigcup_{i=1}^{N} \Omega_i\}} \\ \Gamma_{N_\mathrm{f}} &= \bigcup_{i=1}^{N} \Gamma_i. \end{cases} \tag{3}$$

In order to locate the different objects in the domain we introduce a set of label maps and distance functions.

2.3.1 Label maps

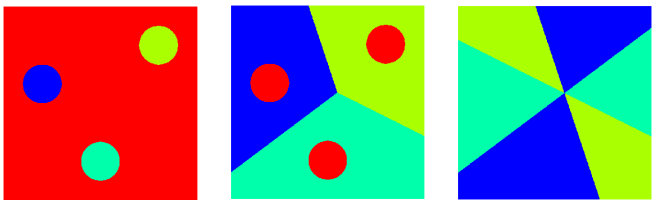
At every point x of the fluid/structure domain Ω, we define the label functions L_0, L_1, L_2 as:

$$\forall x \in \Omega,\ \forall i \in \{1, ..., N+1\}, \quad \begin{cases} L_0(x) &= i \text{ if } x \in \Omega_i \\ L_1(x) &= \underset{j \neq L_0(x)}{\arg\min}\ d(x, \Gamma_j) \\ L_2(x) &= \underset{j \notin \{L_0(x), L_1(x)\}}{\arg\min}\ d(x, \Gamma_j). \end{cases}$$

The label map L_0 provides a partition of the whole computational domain Ω into $N+1$ different objects. The label map L_1 identifies the index of the first closest object at all points in Ω. The label map L_2 identifies the index of the second closest object at all points in Ω. As a consequence, the label map function L_2 gives the index of the first closest cell for any x in the whole computational domain. For example:

$$\begin{cases} L_0(x) = i & \text{if } x \in \Omega_i \\ L_1(x) = j & \text{if the first closest object to } x \text{ is } \Omega_j \\ L_2(x) = k & \text{if the second closest object to } x \text{ is } \Omega_k. \end{cases}$$

Figure 1 shows an illustration of the three label maps in the case of three structures immersed in a fluid. Taking



L_0 L_1 L_2

Figure 1: Illustration of the three label maps, the red color represents the object related to the fluid. For each point of a cell the first closest object is the fluid (in red) and the second closest object is the closest cell.

advantage of this local configuration of the closest object, one can define two related distance functions.

2.3.2 Distance functions

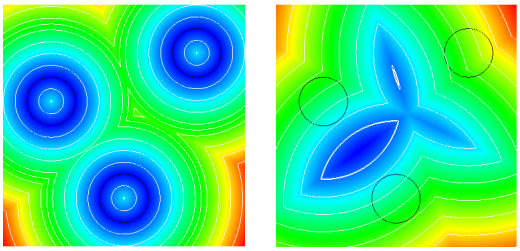
We define two distance functions φ_1 and φ_2 as:

$$\forall x \in \Omega, \quad \begin{cases} \varphi_1(x) = d(x, \Gamma_{L_1(x)}) \\ \varphi_2(x) = d(x, \Gamma_{L_2(x)}). \end{cases}$$

The distance function $\varphi_1(x)$ is the distance from x to the first closest object's boundary $\Gamma_{L_1(x)}$ and $\varphi_2(x)$ is the distance from x to the second closest object's boundary $\Gamma_{L_2(x)}$. As we consider the problem of cells immersed in a fluid, the distance function φ_2 gives the distance between two closest cells. Thus, avoiding contacts is equivalent to require:

$$\forall x \in \Omega\ , \quad \varphi_2(x) > 0.$$

At any point of the domain Ω, the distance function φ_1 captures the union of all cells interfaces and φ_2 gives the distance to the first closest cell. As a consequence, on each point of a cell, we have the distance to the closest one allowing to define a collision model to the closest interface. For a configuration of three bodies the Figure 2 shows an example of the two distance functions related to the label maps of the Figure 1.



φ_1 φ_2

Figure 2: Illustration of the distance functions, the black contours represent the boundary of the three cells.

2.4 Evolution of the label maps and distance functions in the general case

The evolution is based on the transport of one level set function which captures the union of all interfaces and then a local fast-marching method is performed enabling a re-initialization of the label and distance functions.
Let u be the velocity of the fluid, ϕ evolves with equation (1) starting from:

$$\phi(x,0) = \phi^0(x)$$

Given a time step δt, we set $t = n\delta t$ and $\phi^n(.) \approx \phi(.,t)$.

Denoting by Γ_i^t the position of the interface Γ_i at time t we get:

$$\cup_{i=1}^{N} \Gamma_i^t = \{x \in \Omega; \phi^n(x) = 0\}$$

By definition the first distance function φ_1 is the absolute value of this level set function ϕ and the region where the level set function is non negative corresponds to the fluid, which gives at every time the label function L_0 in the fluid region.

$$\forall x \in \Omega, \quad \varphi_1^n(x) = |\phi^n(x)|$$

$$\forall x \in \Omega,\ \forall i \in \{1,...,N\},\ L_0^n(x) = \begin{cases} i \neq N_{\mathrm{f}} & \text{if } \phi^n(x) < 0 \\ N_{\mathrm{f}} & \text{if } \phi^n(x) \geq 0 \end{cases} \tag{4}$$

Taking advantage of these relations we can make the distance function φ_1 and the label L_0 evolve in time. Then a fast marching method is performed to redistanciate and redefine φ_1 and φ_2 and at the same time redefine L_1 and L_2.
The general algorithm performs the following steps:

1. Transport the level set function ϕ with the fluid velocity,
2. Define φ_1 as the absolute value of ϕ,
3. Evolve the Label Function L_0 by redefining its values near the interfaces,
4. Perform a multi label fast marching to redefine the label maps L_1, L_2 and the function φ_2 and redistanciate φ_2 and φ_1,
5. Redefine ϕ using the updated function φ_1.

Redefinition of L_0

In order to evolve the label function L_0, we use the level set function ϕ. At each time step, we change the label value L_0, near the interface at the points where the condition (4) is not verified. Namely, if the level set function is positive, we set L_0 to the label of the fluid ($L_0 = N_{\mathrm{f}}$) and if the level set function is negative and the label function L_0 is still the label of the fluid, we assign to L_0 the value of its neighbours which are different from N_{f}.Let dt denotes the time step. The procedure can be summarized as:

if $\phi^{n+1}(x) \geq 0$ **then**
  $L_0^{n+1}(x) = N_{\mathrm{f}}$
else
  if $(L_0^{n+1}(x) = N_{\mathrm{f}})$ **then**

$L_0^{n+1}(x) = L_0^{n+1}(x_{\text{neighbour}} \notin \Omega_{N_f})$
end if
end if

This simple strategy is allowed only if the distance between two cells is strictly greater than two spacial discretization steps. It will have to be taken into account by our collision model.

Multiple label fast marching method

We present here the local fast marching algorithm that allows to evolve the functions φ_1, φ_2 and the label maps L_1 and L_2. This local fast marching is an extension of the fast marching method [29] that was introduced in [31] and [2]. We use this procedure in order to redistanciate the distance functions φ_1 and φ_2 and at the same time redefine the label maps L_1 and L_2. To redistanciate the distance functions φ_1 and φ_2, we introduce a distance function d. In order to do that, we solve the following eikonal equation in the entire computational domain Ω:

$$|\nabla d| = 1 \tag{5}$$

using a 1st order numerical scheme [28] with an horizontal and a vertical space steps Δx and Δy in two-dimensions we solve:

$$\max(\max(D_x^- d_{ij}, 0)^2, \min(D_x^+ d_{ij}, 0)^2) + \max(\max(D_y^- d, 0)^2, \min(D_y^+ d_{ij}, 0)^2) = 1 \tag{6}$$

where $D_x^- d_{ij} = \frac{d_{ij} - d_{i-1,j}}{\Delta x}$, $D_x^+ d_{ij} = \frac{d_{i+1,j} - d_{i,j}}{\Delta x}$, $D_y^- d_{ij} = \frac{d_{ij} - d_{i,j-1}}{\Delta y}$, $D_y^+ d_{ij} = \frac{d_{i,j+1} - d_{i,j}}{\Delta y}$. At initialization, the function d is equal to φ_1 on the interfaces. All boundaries spread out from each object simultaneously and are associated to the number of objects. Interfaces spread in two directions, inside and outside cells.
There are three sets of points: alive(A), narrow-band (NB) and far away (F). At each point x of the narrow-band is associated an integer $\text{lab}(x)$ which corresponds to the number of the interface that spreads and the distance value $d(x)$ (distance from the interface $\Gamma_{\text{lab}(x)}$). The algorithm performs the following steps.
Initialization phase:

- The points on the interfaces are tagged as A
- The closest points to the interfaces are tagged as NB and we set $\text{lab}(x \in NB) = L_0(x_{\text{neighbours}})$
- All other points are tagged as F and at these points we set: $L_1(x) = L_2(x) = -\infty, d(x) = -\infty$

Iterative phase:

while $(NB \neq \varnothing)$ **do**
Find x_m such as $d(x_m) = \min_{(x \in NB)} d(x)$
Delete x_m from the NB
if it is the first time that x_m has been visited **then**
$\varphi_1(x_m) = d(x_m)$
$L_1(x_m) = \text{lab}(x_m)$
else
$\varphi_2(x_m) = d(x_m)$
$L_2(x_m) = \text{lab}(x_m)$
x_m is tagged as A
end if
if $((L_1(x_{\text{neighbours}}) = -\infty$ **)or** $(L_2(x_{\text{neighbours}}) = -\infty))$ **then**
Compute $d(x_{neighbours})$
Add $x_{\text{neighbours}}$ in NB
end if
end while

All interfaces are propagated simultaneously. At initialization, the narrow-band contains the closest points to the interfaces. The algorithm computes the new values only at the nodes belonging to the narrow-band and accepts just one of them, the one corresponding to the minimum value. If the point x_m has not been visited yet, this minimum value corresponds to the distance $\varphi_1(x_m)$ and the label associated is $L_1(x_m)$. Thus, this point has to be suppressed from the narrow-band.
The second boundary that reachs x_m gives us $\varphi_2(x_m)$ and $L_2(x_m)$. When the two distance functions and the two label maps are defined for a point x, this point is considered as alive and is definitely deleted from the narrow-band. Consequently, the present algorithm stops propagation when the label function L_2 (and so the distance function φ_2) is defined for all points.

Let us underline some differences between the classical fast marching method and our algorithm. In the case of several level set functions, each interface is spread until all points are visited. So, if N level set functions capture

N interfaces, N fast marching are performed. Let M be the number of grid points contained in the narrow band of each cell, we assume that each narrow band contains the same number of grid points. For each fast marching, adding a point in the binary heap has a complexity of the order $\mathcal{O}(\log(M))$ as well as the deleting procedure. Thus, the N fast marching complexity is of order $(NM)\mathcal{O}(\log(M))$ whereas with the multiple label fast marching method, we use only one heap sort that contains NM points inducing a complexity of $(NM)\mathcal{O}(\log(NM))$. Which is worse than using N fast marchings but the proposed algorithm has the ability to stop propagation when the label map L_2 is defined, which alleviates the computational time. Finally, the cpu time difference is very small.

3 Collisions strategy

As discussed before, it is crucial to develop a collision model to avoid contacts between cells. In this section, we present the collision model that consists in a short range repulsive force taking into account the interactions between the closest cells. This short range repulsive force is inspired by the collision model introduced in [4].

3.1 Collision model using several level set functions

The collision model [4] was developed within a level set framework in the context of fluid/rigid bodies interaction. The level set decomposition is used in such a way that each body interface is captured by one level set function. Consider N bodies immersed in a fluid and denote $F_{j,i}$ the force applied by the body Ω_j on the body Ω_i and ϕ_i the level set function which captures the boundary Γ_i of the body Ω_i. The distance of a point x of Ω_i to the body Ω_j is given by $\phi_j(x)$ and the direction of the force $F_{j,i}$ is obtained directly by $\nabla\phi_j$.
Moreover, to localize the interface Γ_i we use a cut off function regularized on a thickness ε on each part of the interface:

$$\forall x \in \Omega,\ F_{j,i}(x) = \frac{k}{\varepsilon}\zeta_\varepsilon(\phi_i(x))\frac{\nabla\phi_j(x)}{\phi_j(x)}\exp(-\frac{\phi_j(x)}{\varepsilon_b})$$

Consequently, we obtain the following collision model:

$$\forall x \in \Omega,\ F_{\text{col}}(x) = \sum_{\substack{i,j=1 \\ i\neq j}}^{N} \frac{k}{\varepsilon}\zeta_\varepsilon(\phi_i(x))\frac{\nabla\phi_j(x)}{\phi_j(x)}\exp(-\frac{\phi_j(x)}{\varepsilon_b}) \tag{7}$$

where k is a repulsive constant proportional to the square of the relative velocities of the corresponding bodies just before collision. The coefficient ε represents the half thickness of the interface on which we apply the repulsive force and ε_b represents the rebound coefficient. In practice we set $\varepsilon_b = \varepsilon$. The interaction forces tend to zero out of a cut-off radius reducing the number of interacting neighbours. This collision model accounts for all possible interactions between the N bodies. Consequently for N bodies captured by N level set functions, N^2 computations of the repulsive forces are required, which represents a huge computational effort.

3.2 Present collision model using two distance functions

We present here the repulsive force used in this work to prevent contacts between cells. To reduce the high computational cost of (7) we propose a reformulation that depends only on the two distance functions φ_1 and φ_2. The following short range repulsive force accounts for the interaction between the closest cells at all points:

$$\forall x \in \Omega,\ F_{\text{col}}^{\text{lab}}(x) = \frac{k}{\varepsilon}\zeta_\varepsilon\left(\varphi_1(x)\right)\frac{\nabla\varphi_2(x)}{\varphi_2(x)}\exp\left(-\frac{\varphi_2(x)}{\varepsilon_b}\right) \tag{8}$$

This force has its support on a subset $\Gamma_\varepsilon = \{x \in \Omega, \varphi_1(x) \leq \varepsilon\}$. As proved in proposition 1, if two cells are at a distance greater than few ε, the force becomes small. In section 6, numerical tests provide evidence that, in that case, this force does not change the dynamics of the bodies. As φ_2 is the distance to the second closest object at all points of the fluid/structures domain, if a cell is surrounded by other cells the interaction of the other cells are taken into account on different part of its interface. For example, on Figure 1 the label L_2 indicates that the repulsive forces acting on the blue cell (top left) is on one part of the interface coming from the yellow cell (top right) and on the other part coming from the green cell (bottom). The advantage of this formulation is that we get rid of the sum in (7) leading to a considerable saving of the computational cost. In addition there is only one repulsive force for an arbitrary number of objects.

3.3 Comparison of the two collision models

In this subsection, we compare the two collision models introduced above in two-dimensions. We consider the case of N circular rigid bodies having the same radius R. The following result gives an estimate of the difference introduced by using the label force (8) instead of the collision model (7). As expected, the difference is smaller for large bodies and/or small ε.

Proposition 1. *Assuming that the N bodies are disks, such as:*

$$\forall x \in \Omega, \quad \varphi_2(x) \geq 2\varepsilon$$

then:

$$\|F_{col} - F_{col}^{lab}\|_{L^1} \leq N(N-2)\left(\frac{2\pi k}{\alpha} + \mathcal{O}\left(\frac{\varepsilon}{R}\right)\right)\exp\left(-\frac{\alpha(R+\varepsilon)}{\varepsilon}\right)$$

where $\alpha = \frac{\sqrt{13}-3}{2} \approx 0.3$.

Proof. Let $L_i(x)$ denote the i^{th} closest object to x and $\phi_{L_i(x)}$ the distance function associated to the object $\Omega_{L_i(x)}$, it comes:

$$\forall x \in \Omega, \forall i \in \{1, ..., N\}, \quad \phi_{L_i(x)}(x) = d(x, \Gamma_{L_i(x)}).$$

Using the assumption on the distance between disks, since the support of the cut-off functions do not intersect, it holds:

$$\forall x \in \Omega, \quad \zeta_\varepsilon\left(\varphi_1(x)\right) = \sum_{j=1}^{N} \zeta_\varepsilon(\phi_j(x)),$$

Then, the model (7) can be written using these functions:

$$\forall x \in \Omega, \quad F_{col}(x) = \sum_{i=2}^{N} \frac{k}{\varepsilon}\zeta_\varepsilon(\varphi_1(x))\frac{\nabla\phi_{L_i(x)}(x)}{\phi_{L_i(x)}(x)}\exp\left(-\frac{\phi_{L_i(x)}(x)}{\varepsilon}\right).$$

As defined in subsection 2.3.2, we have $\varphi_2 = \phi_{L_2}$ and using (8) leads to:

$$\forall x \in \Omega, \quad F_{\text{col}}(x) - F_{\text{col}}^{\text{lab}}(x) = \sum_{i=3}^{N} \frac{k}{\varepsilon}\zeta_\varepsilon(\varphi_1(x))\frac{\nabla\phi_{L_i(x)}(x)}{\phi_{L_i(x)}(x)}\exp(-\frac{\phi_{L_i(x)}(x)}{\varepsilon}). \tag{9}$$

Considering that $\forall i \in \{2, .., N\}, \phi_{L_i}$ are distance functions we get:

$$\|F_{\text{col}} - F_{\text{col}}^{\text{lab}}\|_{L^1} \leq \|\frac{k}{\varepsilon}\zeta_\varepsilon(\varphi_1)\|_{L^1(\Omega)}\|\sum_{i=3}^{N}\frac{1}{\phi_{L_i}}\exp(-\frac{\phi_{L_i}}{\varepsilon})\|_{L^\infty(\Omega)}.$$

Moreover by definition we have:

$$\forall x \in \Omega, \forall i \in \{4, .., N\}, \quad \phi_{L_i(x)}(x) \geq \phi_{L_3(x)}(x).$$

leading to :

$$\|F_{\text{col}} - F_{\text{col}}^{\text{lab}}\|_{L^1} \leq (N-2)\|\frac{k}{\varepsilon}\zeta_\varepsilon(\varphi_1)\|_{L^1(\Omega)}\|\frac{1}{\phi_{L_3}}\exp(-\frac{\phi_{L_3}}{\varepsilon})\|_{L^\infty(\Omega)}.$$

It is a classical result from level-set theory that $\frac{1}{\varepsilon}\zeta_\varepsilon(\varphi_1(x))$ is an approximation of the length of the zero level-set of φ_1. Namely, in the case of N disks, we can prove that

$$\|\frac{1}{\varepsilon}\zeta_\varepsilon(\varphi_1)\|_{L^1(\Omega)} = N(2\pi R + \mathcal{O}(\varepsilon)),$$

Therefore,

$$\|F_{\text{col}} - F_{\text{col}}^{\text{lab}}\|_{L^1(\Omega)} \leq N(N-2)(2\pi Rk)\|\frac{1}{\phi_{L_3}}\exp(-\frac{\phi_{L_3}}{\varepsilon})\|_{L^\infty}$$

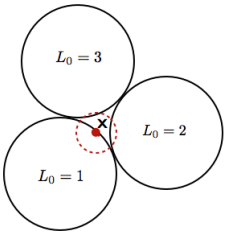


Figure 3: Configuration of three bodies. The circles represented are of radius $R + \varepsilon$.

We are looking for a lower bound of ϕ_{L_3} depending on the radius R. To that aim, we consider the worst case of three disks which are enlarged by ε and are in contact like shown in Figure 3. This configuration gives the minimum distance required between two disks. Taking a point x on the boundary of one disk (without loss of generality, Γ_1) as shown in Figure 3, we compute the minimal distance of x to the second closest disk. Once again, the worst case is obtained in the situation where x is as in Figure 3. Through a simple calculation, we obtain as distance:

$$d = \frac{\sqrt{13} - 3}{2} R$$

that is a lower bound for ϕ_{L_3}. As $r \to \frac{1}{r}\exp(-\frac{r}{\varepsilon})$ is decreasing, we obtained the announced estimation. □

Let us point out to the reader that this difference tends to zero when R tends to ∞ or when ε tends to zero. Consequently, we can adjust ε depending upon R such as this difference becomes negligible. Numerically, we take an ε that depends on the discretization space step, thus the grid mesh size is selected in order to lower the difference.
Moreover, the influence of the repulsive force imposed by the first closest cell at all points $F_{\text{col}}^{\text{lab}}$, is the most influential on the dynamics of the cells as it is the largest. Therefore, numerically it is better to compare the relative difference:

$$\frac{\|F_{\text{col}} - F_{\text{col}}^{\text{lab}}\|_{L^1}}{\|F_{\text{col}}^{\text{lab}}\|_{L^1}}.$$

4 Application to rigid bodies

4.1 Flow configuration

We consider N cells evolving in an incompressible fluid denoted Ω_{N_f} and the entire domain Ω is partitioned in the same way than (3). In this case each cell $\Omega_i, 1 \leq i \leq N$ represents a rigid body, ρ_f and μ denote the constant density and the viscosity of the fluid, U and p denote the flow velocity and the pressure. The flow is governed by the viscous incompressible Navier-Stokes equations:

$$\begin{cases} \rho_f(\partial_t U + (U \cdot \nabla)U) - \nabla \cdot (\mu \nabla U) + \nabla p = 0 & \text{in } \Omega_{N_f} \times (0, T) \\ \nabla \cdot U = 0 & \text{in } \Omega_{N_f} \times (0, T) \end{cases}$$

with the condition at the interfaces of the solid bodies:

$$\forall i \in \{1, ..., N\},\ U = u_i \ on\ \Gamma_i \tag{10}$$

where u_i is the rigid motion of the body Ω_i. In order to get rid of the boundary conditions (10) on the solid boundaries, we use a penalization method ([1, 25, 4]). This method consists in adding a penalization term in the Navier-Stokes equations to impose the rigid motion inside the solid and to solve the boundary value problem inside the whole domain Ω including the bodies. So, there is no need for an adapted mesh to the geometry of the bodies and a Cartesian mesh on a box domain can be used. If we denote by χ_i the characteristic function of the body Ω_i and by λ the penalization parameter we obtain the following model:

$$\begin{cases} \rho(\partial_t U + (U \cdot \nabla)U) - \nabla \cdot (\mu \nabla U) + \nabla p = \rho g + \lambda(\sum_{i=1}^{N} \chi_i(u_i - U)) + F_{\text{col}} + F_{\text{wall}} & \text{in } \Omega_T = \Omega \times [0, T] \\ \nabla \cdot U = 0 & \text{in } \Omega_T \end{cases} \tag{11}$$

where g is the gravity force, F_{col} is the collision force (8) and F_{wall} is a repulsive force carry on by the solid boundaries of the domain Ω on the rigid bodies. The rigid velocity u_i is obtained by averaging translation and

angular velocities over the solid body Ω_i ([25]):

$$u_i = u_i^t + u_i^\theta = \frac{1}{\mid \Omega_i \mid} \int_{\Omega_i} \rho \chi_i u dx + \left(J_i^{-1} \int_{\Omega_i} \rho \chi_i u \times (x - x_i^g) dx \right) \times (x - x_i^g) \tag{12}$$

where J_i is the inertial matrix of the body Ω_i and x_i^g its center . As the rigid bodies move with the fluid, the displacement of χ_i is:

$$\forall x \in \Omega, \forall i \in \{1, ..., N\}, \quad \partial_t \chi_i + U \cdot \nabla \chi_i = 0. \qquad \text{or} \qquad \partial_t \chi_i + u_i \cdot \nabla \chi_i = 0.$$

Finally, denoting by ρ_i the density of the immersed bodies Ω_i, we obtain the following density function:

$$\rho = \rho_f + \sum_{i=1}^{N} (\rho_f - \rho_i) \chi_i.$$

Let us note that the coefficient $\lambda >> 1$ means that the regions occupied by the solid bodies are considered as porous media with a very small permeability.

4.2 The penalization model using several level set functions

The characteristic function χ_i can also be deduced considering a level set function ϕ_i that captures the interface Γ_i. This level set function evolves as:

$$\partial_t \phi_i + u_i \cdot \nabla \phi_i = 0$$

and using the regularized Heaviside function χ_i can be smoothed as:

$$\chi_i^\varepsilon = 1 - H_\varepsilon(\phi_i).$$

So the model becomes:

$$\begin{cases} \rho(\partial_t U + (U \cdot \nabla)U) - \nabla \cdot (\mu \nabla U) + \nabla p = \rho g + \lambda(\sum_{i=1}^{N} \chi_i^\varepsilon (u_i - U)) + F_{\text{col}} + F_{\text{wall}} & \text{in } \Omega_T \\ \nabla \cdot U = 0 & \text{in } \Omega_T \\ \partial_t \phi_i + u_i \cdot \nabla \phi_i = 0 & \text{in } \Omega_T \end{cases} \tag{13}$$

where

$$\forall x \in \Omega, \ F_{col}(x) = \sum_{\substack{i,j=1 \\ i \neq j}}^{N} \frac{k}{\varepsilon} \zeta_\varepsilon(\phi_i(x)) \frac{\nabla \phi_j(x)}{\phi_j(x)} \exp(-\frac{\phi_j(x)}{\varepsilon_1}).$$

This system is dependent on the number of structures causing a high computational cost if one wishes to simulate a large number of bodies.

4.3 The proposed penalization model

Using our method, we can define the following penalized model

$$\begin{cases} \rho(\partial_t U + (U \cdot \nabla)U) - \nabla \cdot (\mu \nabla U) + \nabla p = \rho g + \lambda(\chi_{L_0^\varepsilon}(u_{L_0^\varepsilon} - U)) + F_{\text{col}}^{\text{lab}} + F_{\text{wall}} & \text{in } \Omega_T \\ \nabla \cdot U = 0 & \text{in } \Omega_T \\ \partial_t \phi + u \cdot \nabla \phi = 0 & \text{in } \Omega_T \end{cases} \tag{14}$$

where $\chi_{L_0^\varepsilon}$ is the characteristic function of the region defined by the label map L_0^ε which is an extension to the distance ε of the label L_0:

$$\forall x \in \Omega, \forall i \in \{1, ..., N\}, \quad L_0^\varepsilon(x) = \begin{cases} i & \text{if} \quad d_s(x, \Gamma_i)) \leq \varepsilon \\ N_f & \text{otherwise} \end{cases} \tag{15}$$

where $d_s(x, \Gamma)$ is the signed distance function from x to Γ.
The label map L_0^ε can also be defined using the label maps definition as:

$$\forall x \in \Omega, \quad L_0^\varepsilon(x) = \begin{cases} L_0(x) & \text{if} \quad (L_0(x) \neq N_f) \\ L_1(x) & \text{if} \quad ((L_0(x) = N_f) \text{ and } (\phi \leq \varepsilon)) \\ N_f & \text{otherwise} \end{cases} \tag{16}$$

We can define a characteristic functions of the solid bodies:

$$\forall x \in \Omega, \forall y \in \Omega, \quad \chi_{L_0^\varepsilon(x)}(y) = 1 - H(\phi_{L_0^\varepsilon(x)(y)})$$

In the case of the algorithm 5.1, the level set functions ϕ_i will be defined as:

$$\forall x \in \Omega, \forall i \in \{1, ..., N\}, \quad \phi_i(x) = \begin{cases} \phi(x) & \text{if} \quad ((L_0(x) = i) \text{ or } (L_1(x) = i)) \\ \phi_2(x) & \text{otherwise} \end{cases} \tag{17}$$

And like for the penalization model (13) above, for each $x \in \Omega$, $y \to u_{L_0^\varepsilon(x)}(y)$ is the rigid velocity of the solid body $\Omega_{L_0^\varepsilon(x)}$ obtained by averaging the translation and angular velocities over the solid. Setting

$$\forall x \in \Omega, \quad \mid \Omega_{L_0^\varepsilon(x)} \mid = \int_{\Omega_{L_0^\varepsilon(x)}} \rho(z)dz = \int_\Omega \rho(z)\chi_{L_0^\varepsilon(x)}(z)dz$$

we obtain the following formulation which is equivalent to (15). $\forall x \in \Omega$, $\forall y \in \Omega$,

$$u_{L_0^\varepsilon(x)}(y) = \frac{1}{\mid \Omega_{L_0^\varepsilon} \mid} \int_{\Omega_{L_0^\varepsilon(x)}} \rho_x(z)\chi_{L_0^\varepsilon(x)}(z)u(z)dz + \left(J^{-1}_{L_0^\varepsilon(x)} \int_{\Omega_{L_0^\varepsilon(x)}} \rho_x(z)\chi_{L_0^\varepsilon(x)}(z)u(z) \times (y - x^g_{L_0^\varepsilon(y)})dz \right) \times (y - x^g_{L_0^\varepsilon(y)}).$$

where $J_{L_0^\varepsilon(x)}$ and $x^g_{L_0^\varepsilon(x)}$ denote the inertial matrix and center of gravity of solid $\Omega_{L_0^\varepsilon(x)}$. Denoting by $\rho_{L_0^\varepsilon(x)}$ the density of the body $\Omega_{L_0^\varepsilon(x)}$ we obtain the following density function:

$$\rho_x = \rho_f(1 - \chi_{L_0^\varepsilon(x)}) + \chi_{L_0^\varepsilon(x)}\rho_{L_0^\varepsilon(x)}$$

On the implementation side, all rigid velocities are computed incrementally, involving only one iteration on the mesh grid. This penalisation model therefore becomes completely independent on the number of bodies. Thanks to the label maps, we have suppressed the dependence on the number of bodies in the repulsive force and in the penalization term. This is a very desirable model to simulate a large number of interacting cells. In the case of circular rigid bodies, the level set equation is replaced in the previous model (14) by the transport of the gravity center of the N bodies and a reconstruction of the associated level set function is performed:

$$\partial_t x_i^g = u_i^t, \quad 1 \leq i \leq N$$

5 Numerical implementation of the fluid/structure models

This part is devoted to the numerical implementation of the system (14) with a change on the transport of the level set function.
The system is discretized by a finite difference method on a staggered grid where the pressure and the level set function are located at the center of the mesh cells and the velocity at the center of the sides in two-dimensions and of the faces in three-dimensions. Consequently the divergence free is computed at the pressure point, enforcing the volume constraint very accurately. The Navier-Stokes equations are solved using an incremental projection method of Chorin type. First, we compute an intermediate state u^* from:

$$u^* = u^n - \Delta t(u^n \cdot \nabla)u^n + \frac{\Delta t \mu}{\rho^n}\Delta u^n + \Delta t g - \frac{\Delta t \nabla p^n}{\rho^n}. \tag{18}$$

Then we solve the pressure from the equation:

$$\nabla \cdot (\frac{\nabla p^{n+1}}{\rho^n}) = \frac{div(u^*)}{\Delta t} + \nabla \cdot (\frac{\nabla p^n}{\rho^n}) \tag{19}$$

so that the velocity:

$$\overline{u} = u^* - \frac{\Delta t}{\rho_0}(\nabla p^{n+1} - \nabla p^n) \tag{20}$$

is divergence free. The pressure equation(19) can be solved directly by a conjugate gradient algorithm or approximated with a relaxation procedure as follows:

$$\Delta p^{n+1} = \frac{\rho_0}{\Delta t} div(u^*) + \Delta p^n$$

setting $1/\rho^n = 1/\rho_0 - (1/\rho_0 - 1/\rho^n)$ in order to get a Poisson equation much faster to solve. In that case, the step $\overline{u}$ satisfies Navier-Stokes equations with a modification of the pressure term $(1/\rho^n - 1/\rho_0)\nabla p^n + 1/\rho_0 \nabla p^{n+1}$ instead of $1/\rho^n \nabla p^{n+1}$. It remains to choose ρ_0 in order to have the best approximation. We have tested three different values $\rho_0 = \rho_f$, $\rho_0 = \rho_{L_0}$ and $\rho_0 = (\rho_{L_0} + \rho_f)/2$. The Poisson solver is based on a classical 5 or 7 points second order stencil according to the dimension, the viscous terms are discretized by a second order central scheme and the convection term is discretized by a 5^{th} order WENO scheme. To take into account the solid bodies we use the method proposed in [30]. An implicit treatment of the penalization term is achieved in order to use larger penalization coefficients λ and therefore the interface boundary condition is satisfied with better accuracy. This algorithm appears to be optimal in the case of spherical rigid bodies but we first present it in the general case before going to the spherical rigid bodies case.

5.1 Case of general rigid bodies

In the general case the algorithm performs the following steps:

1. Compute and add the repulsive force $u_{\text{col}} = \overline{u} + \Delta t F^{\text{lab}}_{\text{col}}$,

2. Compute the translation velocity $u_t^{L_0}$ of the body Ω_{L_0}, $u^t_{L_0} = \dfrac{\int_\Omega \rho^n u_{\text{col}} \cdot \chi_{L_0^n} dx}{\int_\Omega \rho^n \chi_{L_0^n} dx}$,

3. Compute the rotational velocity w^{L_0} of the body Ω_{L_0}

$$w^{L_0} = J^{-1}_{L_0} \int_\Omega \rho^n r^n \times u_{\text{col}} \cdot \chi_{L_0^n} dx \times r^n, \text{ where } r^n = (x - x^g_{L_0^n})$$

4. Compute the rigid velocity u_{L_0} of the body Ω_{L_0}, $u_{L_0} = u^t_{L_0} + u^t_{L_0}$

5. Correct the velocity using an implicit treatment of the penalization term

$$\frac{u^{n+1} - u_{col}}{\Delta t} = \frac{1}{\lambda} \chi_{L_0}(u_{L_0} - u^{n+1}),$$

6. Transport the solid bodies with the fluid velocity, $\phi^{n+1} = \phi^n - \Delta t u^{n+1} \cdot \nabla \phi^n$,

7. Redefine the distance function $\varphi_1^{n+1} = |\phi^{n+1}|$,

   Redefine L_0^{n+1} using ϕ^{n+1}

   Perform a multi label fast marching method using a 1st order numerical scheme [28].

5.2 Case of spherical rigid bodies

In the case of a rigid body captured by one level set function, it is more efficient to displace the body by transporting the level set function with the rigid velocity of the corresponding body. Indeed, the rigid velocity is more regular than the fluid velocity and thus the redistanciation procedure is not necessary.
In our model, the rigid velocities u_{L_0} are described locally in a neighbourhood of the body Ω_{L_0}. Consequently, it is not possible to apply this procedure. An alternative algorithm in the case of spherical rigid bodies is to transport the center of gravity $x^g_{L_0}$ with the rigid motion u_{L_0} and then to reconstruct the N level set functions. This algorithm is faster than performing a multi label redistanciation.
Then, the steps 6 and 7 of the previous algorithm are modified as follows:

6. Transport the solid bodies using the centers: $x_g^{n+1}(i) = x_g^n(i) + \Delta t u_t^i$, $\phi_i^{n+1}(x) = \phi^0(x_g^{n+1}(i))$

7. Redefine the label maps and distance functions:

$$\forall x \in \Omega, \forall i \in \{1, ..., N+1\}, \quad \begin{cases} L_0^{n+1}(x) = i \text{ if } \phi_i^{n+1} \leq 0 \\ L_1^{n+1}(x) = \underset{j \neq L_0^{n+1}(x)}{\arg\min} \ \phi_j^{n+1} \\ L_2^{n+1}(x) = \underset{j \notin \{L_0^{n+1}(x), L_1^{n+1}(x)\}}{\arg\min} \ \phi_j^{n+1} \end{cases}$$

$$\forall x \in \Omega, \quad \begin{cases} \varphi_1^{n+1}(x) &= d(x, \Gamma_{L_1^{n+1}(x)}) \\ \varphi_2^{n+1}(x) &= d(x, \Gamma_{L_2^{n+1}(x)}) \end{cases}$$

To find the best value for ρ_0, we perform a numerical test with six rigid disks falling under gravity in two dimensions. The Figure 4 shows that the results obtained with $\rho_0 = \rho_f$ gives the worst approximation whereas the two other values are very close to the approximation of the exact solution computed with the conjugate gradient. In the following we set $\rho_0 = (\rho_{L_0} + \rho_f)/2$.

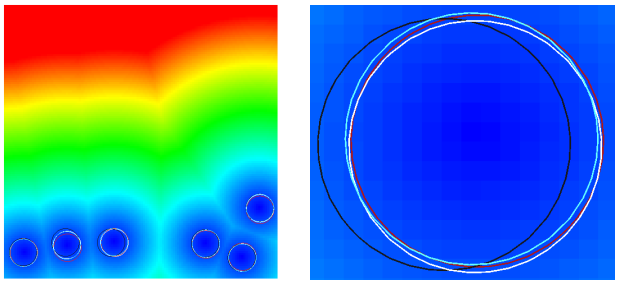


Figure 4: Flow field for six rigid disks falling under gravity (left) and a zoom of the location of the interface for different values of ρ_0 (right). The conjugate gradient method is in red, $\rho_0 = \rho_f$ in black, $\rho_0 = \rho_{L_0}$ in white and $\rho_0 = (\rho_{L_0} + \rho_f)/2$ in blue. The background color shows the level set amplitude.

5.3 Comparison of the two algorithms

In this case of circular rigid bodies, we compare the CPU time of both algorithms. In algorithm 5.1, the transport of the solid bodies with the help of one level set function is achieved using a WENO5 scheme whereas in algorithm 5.2 it is performed by direct explicit transport of the center of the bodies. For a reasonable number of cells this procedure is very fast. The redefinition of the label maps is performed either with a time consuming multi label fast marching or using N signed distance functions that are updated directly thanks to the N gravity centers. In consequence, the CPU time is much lower as can be seen in table 1 and is 10 times faster for a number of cells $N = 200$. Of course, this is true only in the case of circular rigid disks and spheres.

Number of disks	Algorithm 5.1 CPU time	Algorithm 5.2 CPU time
2	1.3	0.2
5	1.3	0.23
25	1.3	0.4
50	1.7	0.6
100	4.3	0.9
200	23.6	2.4

Table 1: Computational time of both algorithms.

6 Numerical illustrations

In this section, we present the numerical results obtained with the proposed model and give some comparisons with existing methods of the literature. In a first part, we present a qualitative grid convergence. Then we compare our model (14) to the model (13). In a third part, we give a qualitative comparison of our model with those proposed in [20]. Finally, some simulations of dense suspensions of circular rigid bodies in two and three dimensions are presented.

For all of the simulations presented in this paper, the computational domain Ω is a square of size $[0, 1]^2$ or a cube of size $[0, 1]^3$. In order to compare with the results in [4] and in [20] the dynamic viscosity μ is set to 0.01, the density of the fluid is set to $\rho_f = 1$, and the density of the rigid bodies is the same for all the bodies $\rho_s = 2$.

Let $k = (k_x, k_y, k_z)^t$ be the repulsive coefficient between bodies and $k^{\text{wall}} = (k_x^{\text{wall}}, k_y^{\text{wall}}, k_z^{\text{wall}})^t$ be the repulsive coefficient exerted by the walls, the value of the components depends on the amplitude of the force.

6.1 Grid convergence

To study the grid convergence, we use four different grids (G_1, G_2, G_3,G_4) which contain respectively : (128×128), (256×256), (512×512) and (1024×1024) cells on a uniform mesh. The coefficient ε is set to Δx^{G_1} where Δx^{G_1} denotes the mesh size corresponding to the coarsest grid (128×128). We take as test case the sedimentation of 25 circular rigid bodies having the same radius $R = 0.025 + \varepsilon$. The gravity force g is set to -980 and the repulsive coefficients on the cells and on the walls are:

$$k_x = -g/10, \quad k_y = -g/10, \quad k_x^{\text{wall}} = -g/40, \quad k_y^{\text{wall}} = -g.$$

The repulsive forces are applied on a ring around the interface of thickness $1.5\Delta x^{G_1}$.
In Figure 5 the results obtained with the four different grids are presented, the white line corresponds to the numerical size of the rigid particles ($\phi = \varepsilon$). We can see that the dynamic of the rigid bodies and the interaction between them is quite similar for the different resolutions. The collisions are avoided thanks to the repulsive force. We observe the same phenomenon of kissing and tumbling of the bodies.
By $t = 0.48$, all simulations have reached static equilibrium which represents different local minimum of the sedimentation of the 25 disks. We observe that the two fine simulations are very close to each other until time $t = 0.15$ and keep a symmetrical distribution. At the end of the simulation, a packing of the bodies is formed at the bottom of the computational domain and is composed of three layers. Each layer contains the same number of structures for the four different grids: eleven bodies on the first one, ten bodies on the second one and four on the last one. The distribution of the four bodies of the third layer is different for the various grids but is much closer on the two finest simulations. Thus a qualitative grid convergence is achieved.

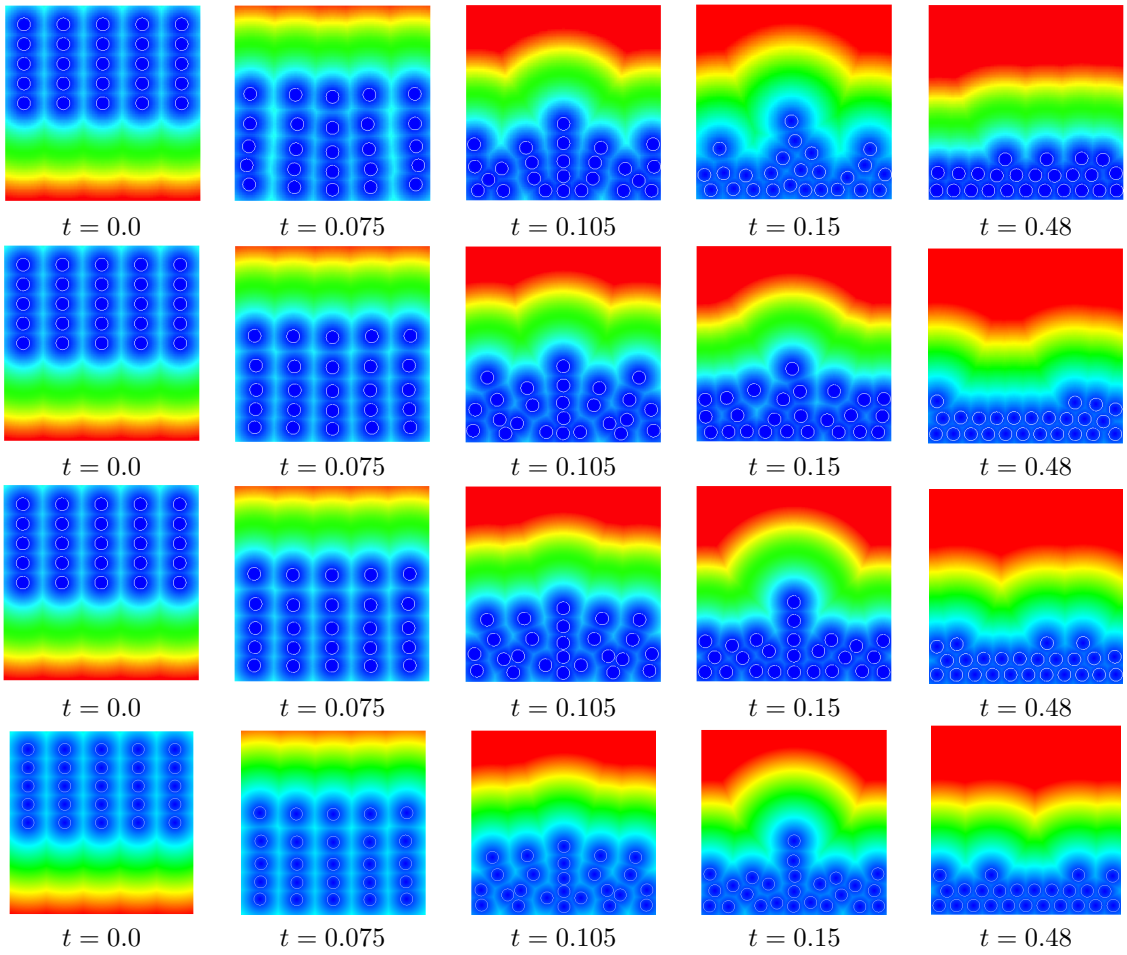


Figure 5: Study of the grid convergence with a test case of 25 rigid disks falling under gravity. From top to bottom, grid 128×128, grid 256×256, grid 512×512 and grid 1024×1024. The background color shows the level set amplitude.

6.2 Comparison of the method with level set decomposition

We give here a comparison of our penalization model (14) and the penalization model that uses a level set decomposition (13)

6.2.1 Computational time using the algorithm (5.2) for rigid disks

We first give a comparison of the computational time in the case of N rigid disks. The algorithm associated to our model is given in section 5.2. Instead of transporting one level set function and performing the multi label fast marching method, we transport the gravity centers of the N rigid structures and reconstruct their associated level set functions. Then, the label and distance functions are reinitialized by using their definition. It is obvious that this algorithm depends on the number of cells because of the N level set functions and their associated center of gravity. However, this part of the algorithm is very fast. This algorithm will allow us to compare the saving computational time which is induced by changing the existing collision model (7) and the penalization term. We average the computational time on the ten first iterations.
The averaged CPU time of our algorithm (table 2) is compared to the method using N level set functions (table 1), according to the number of cells. As noticed before the collision model (7) computes N^2 repulsive forces which induced a high computational cost as shown in the second column of table 1 whereas in the present algorithm, the CPU time of the collision model is constant as it does not depend on the number of cells. The CPU time of the penalization model is larger in (13) because it depends on the number of cells. Indeed, N rigid velocities must be computed to get the right velocity of each cell. In the present algorithm, we must add the label redefinition that depends almost linearly on the number of cells and so is quite cheap. For a low number of cells the CPU time of both methods is close but increasing this number from 2 to 400 cells induces a total CPU time 8000 larger for the model (13) whereas with the present model the CPU time is only 25 times larger. So our method is around 340 times faster for 400 cells.

Number of disks	Collision model (7) CPU time	Penalization model (13) CPU time	Total CPU time
2	0.02	0.06	0.2
5	0.17	0.16	0.48
10	0.72	0.35	1.24
25	4.87	0.88	6
50	19.25	1.75	21.5
100	80.8	3.9	85.3
400	1583.4	19.75	1605.3

Table 2: Averaged CPU time using the N level set decomposition

Number of disks	Collision model (8) CPU time	Penalization model (14) CPU time	Label redefinition CPU time	Total CPU time
2	0.015	0.05	0.008	0.2
5	0.015	0.06	0.014	0.23
10	0.015	0.09	0.02	0.25
25	0.016	0.18	0.08	0.4
50	0.016	0.3	0.16	0.6
100	0.016	0.56	0.23	0.9
400	0.016	2.52	2.06	4.7

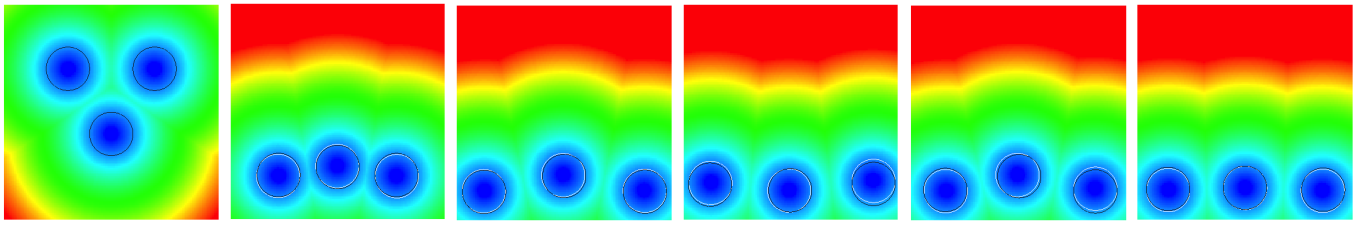
Table 3: Averaged CPU time using the algorithm of section 5.2

6.2.2 Numerical comparison of the two collision models

To highlight the differences between the two collision models we first focus on a a test case with three circular rigid bodies falling on each other. The simulations are performed on a grid of size (128×128) corresponding to a space step $\Delta x = 7.8125.10^{-2}$ in order to better see the difference between the two models. The bodies have the same radius $R = 0.1$ and the thickness of the interface is $\varepsilon = 2\Delta x$. The repulsive coefficients are:

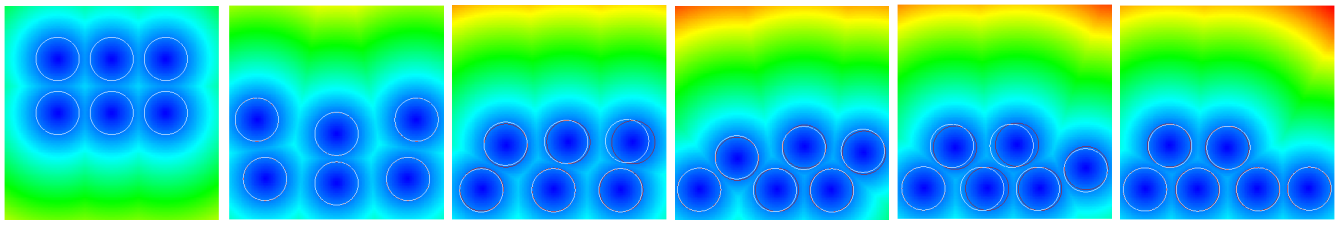
$$k_x = -g/10, \;\; k_y = -g/10, \;\; k_x^{\text{wall}} = -g/40, \;\; k_y^{\text{wall}} = -g.$$

On Figure 6, black and white lines represents the interfaces of the three different objects. The black line stands for the collision model (7) and the white line for the collision model (8) corresponding also to the colors that represent the values of the level set function. We can see that the bodies have the same behaviour, as expected, because on the one hand the radius is large and on the other hand the forces applied on the bodies are very similar. So the difference is small. The Figure 7 shows the results obtained with 6 rigid bodies. Here we can see again that the behaviour is very similar for a large radius. The last test concerns the same configuration with six smaller disks with radius $R = 0.03$ leading to a higher difference in the dynamics even if the final state is the same. The difference between the two models is stronger when the number of body is larger or when the force coefficients are higher. The bottom plot in Figure 9 shows the vorticity inside the fluid domain, when the bodies reached the bottom they move to the right and induce a strong positive vortex that has a strong influence on the dynamics of the bodies in its turn. We observe that the vorticity increases when the bodies are close to each other as the repulsive force is higher. At time $t = 0.5$ the vorticity vanishes because the six rigid bodies have reached the static equilibrium.



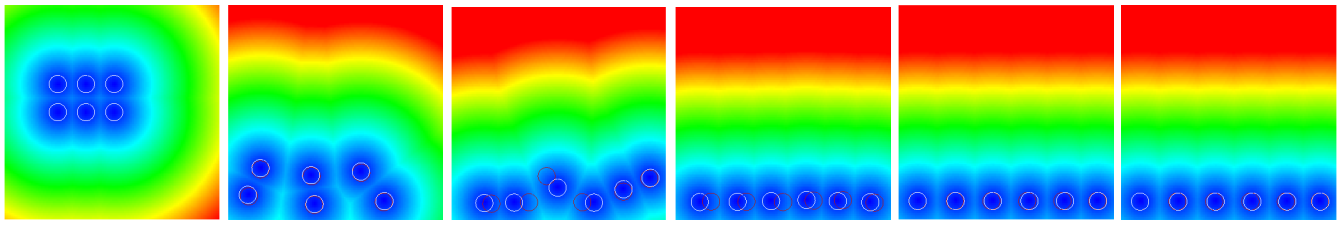
(a) $t = 0.0$ (b) $t = 0.08$ (c) $t = 0.1$ (d) $t = 0.12$ (e) $t = 0.15$ (f) $t = 0.23$

Figure 6: Comparison of the two collision models for three disks of radius $R = 0.1$. The background colors show the level set amplitude.



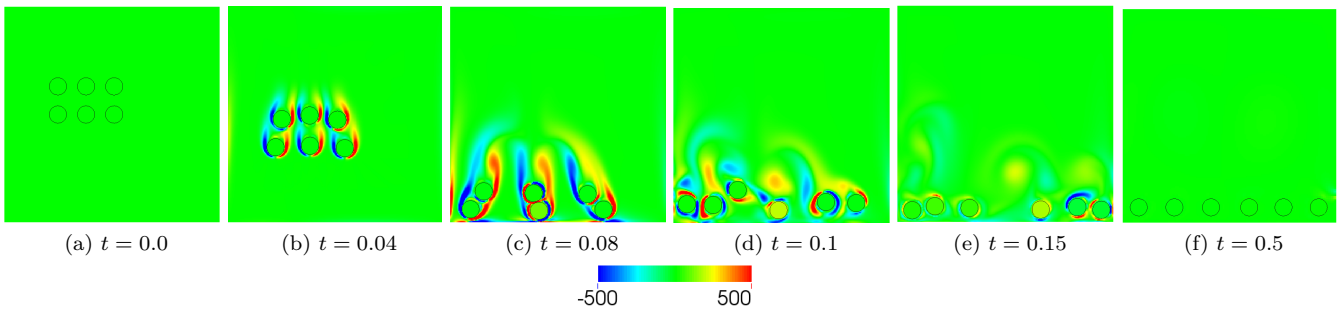
(a) $t = 0.0$ (b) $t = 0.08$ (c) $t = 0.15$ (d) $t = 0.25$ (e) $t = 0.38$ (f) $t = 0.50$

Figure 7: Comparison of the two collision models for six disks of radius $R = 0.1$.The background colors show the level set amplitude.



(a) $t = 0.0$ (b) $t = 0.08$ (c) $t = 0.15$ (d) $t = 0.25$ (e) $t = 0.38$ (f) $t = 0.5$

Figure 8: Comparison of the two collision models for six disks of radius $R = 0.03$. The background colors show the level set amplitude.



(a) $t = 0.0$ (b) $t = 0.04$ (c) $t = 0.08$ (d) $t = 0.1$ (e) $t = 0.15$ (f) $t = 0.5$

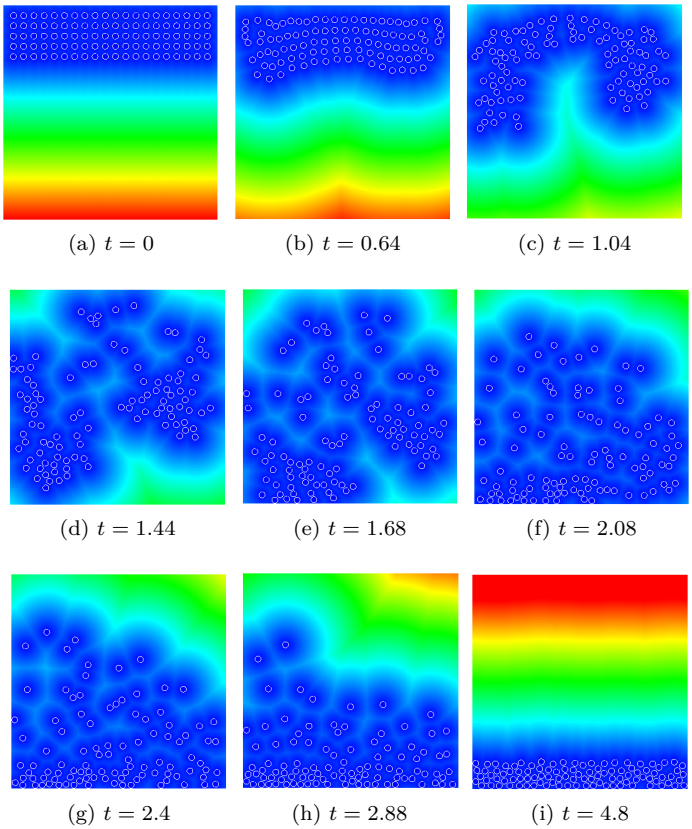
Figure 9: Six disks falling under gravity, colors represents the values of the vorticity field.

6.3 Comparison with the model introduced in [20]

In this part, we compare qualitatively our model to an existing method proposed in [20]. In that model, the solid bodies are taking into account by penalizing the strain tensor to enforce the rigid body motion (see [19]). A scheme for inelastic collisions is implemented imposing a minimal distance between bodies and therefore avoiding contacts (see [23] for more details). Moreover, the contacts with the four walls are also handled in the same way. The test case is the sedimentation of 100 rigid particles of radius $R = 0.01$ subject to the gravity force $g = -70$. The corresponding repulsive coefficient are:

$$k_x = -g/7, \;\; k_y = -g/7, \;\; k_x^{\text{wall}} = -g/28, \;\; k_y^{\text{wall}} = -g.$$

The Figure 10 shows the results obtained with the FreeFem code implemented by A.lefebvre (see [20]) on a mesh with x elements. Using our model, the simulations are performed on a grid of size (512×512), the half interface thickness ε is set to the mesh size Δx and the results are shown in Figure 11. Despite a symmetrical configuration the results in Figure 10 are unsymmetric from the beginning whereas our results stay symmetric until time $t = 1.44$. However, we can see that the dynamics is globally the same and that the static equilibrium is reached at the same time $t = 4.8$.

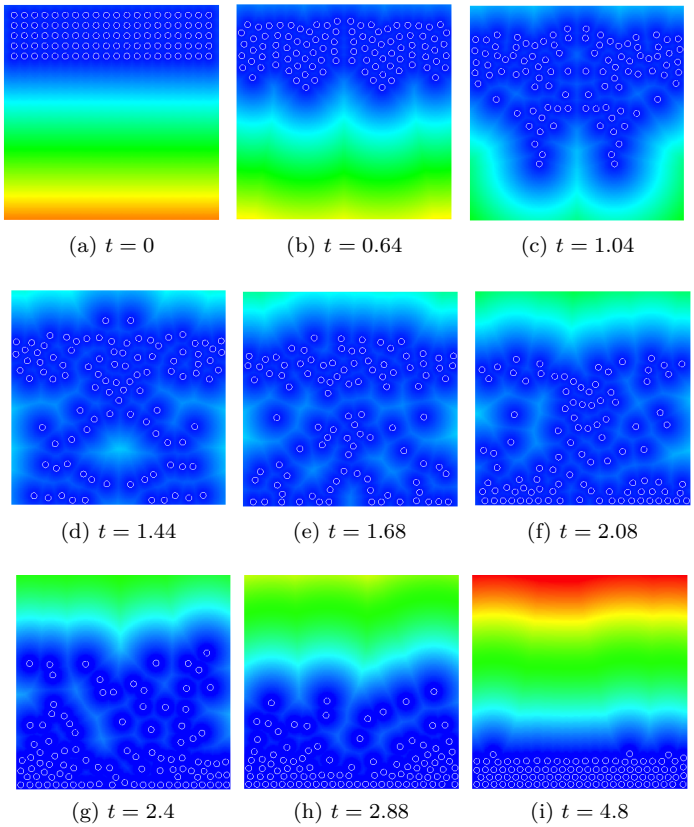


(a) $t = 0$ (b) $t = 0.64$ (c) $t = 1.04$

(d) $t = 1.44$ (e) $t = 1.68$ (f) $t = 2.08$

(g) $t = 2.4$ (h) $t = 2.88$ (i) $t = 4.8$

Figure 10: Simulation of 100 rigid particles submitted to gravity obtained with the FreeFem code [20]



(a) $t = 0$ (b) $t = 0.64$ (c) $t = 1.04$

(d) $t = 1.44$ (e) $t = 1.68$ (f) $t = 2.08$

(g) $t = 2.4$ (h) $t = 2.88$ (i) $t = 4.8$

Figure 11: Simulation of 100 rigid particles submitted to gravity obtained with our model. The background colors show the level set amplitude.

6.4 Dense suspensions of rigid bodies in 2D and 3D

In this part, we present some results of dense suspensions of rigid bodies subject to gravity which was performed using our numerical model. The first simulations deal with the sedimentation of 400 rigid bodies of radius $R = 0.01$ in the two dimensional case. The simulations are performed on a grid of size (512×512) and the half thickness of the interface is $\varepsilon = 1.5\Delta x$. The white line shows the real numerical size of the particles corresponding to the isoline $\phi = \varepsilon$. The coefficient of gravity g is set to -980. The repulsive coefficients are:

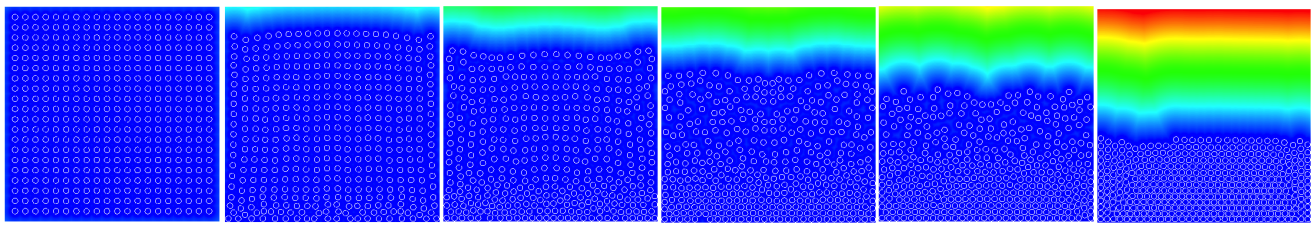
$$k_x = -g/10, \quad k_y = -g/10, \quad k_x^{\text{wall}} = -g/40, \quad k_y^{\text{wall}} = -g.$$

The 400 bodies fall down symmetrically to reach a dense repartition at the bottom as can be seen in the Figure 12.

The second simulation addresses the 3D case. Figures 13 and 14 show the simulation of 200 rigid spheres of radius $R = 0.01$ falling under gravity for two different grids of size 64^3 and 128^3. The half thickness of the interface is $\varepsilon = 2\Delta x$. The coefficient of gravity g is set to -980. The repulsive coefficients are:

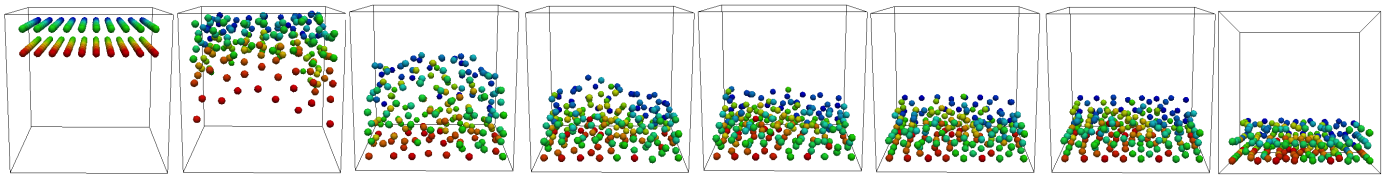
$$k_x = -g/10, \quad k_y = -g/10, \; k_z = -g/10, \quad k_x^{\text{wall}} = -g/40, \quad k_y^{\text{wall}} = -g/40, \; k_z^{\text{wall}} = -g.$$

At initial step, there are two slices of 100 bodies at a distance $d = 0.1$ (distance of two closest bodies' centers). Consequently, on the coarser mesh there is only one full mesh cell between the two numerical slices. Indeed, the numerical radius is $R + \varepsilon \approx 0.041$ and so the repulsive forces are active, whereas on the finer mesh there are on average 6 mesh cells between the two numerical slices as $R + \varepsilon \approx 0.026$. In that case the repulsive forces are negligible. The interactions between bodies occur at once on the coarse mesh while they start after $t = 1.5$ on the fine mesh. As a consequence, the equilibrium state is reached much faster on the fine grid, $t = 2.4$ instead of $t = 8.9$ for the coarse resolution.



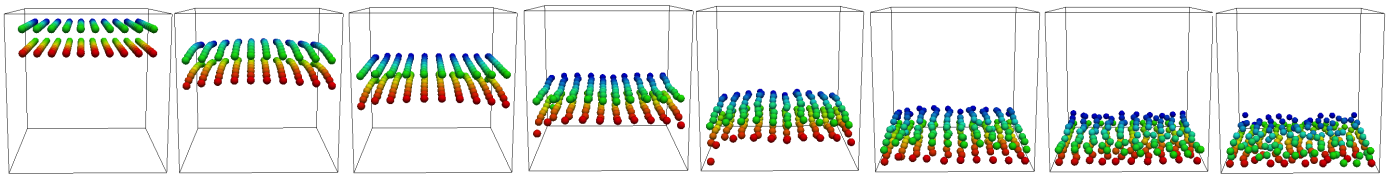
(a) $t = 0.0$ (b) $t = 0.75$ (c) $t = 1.5$ (d) $t = 2.25$ (e) $t = 3.0$ (f) $t = 6.9$

Figure 12: Simulation of 400 rigid disks submitted to gravity (the white line corresponds to the level line $\phi = \varepsilon$). The background colors show the level set amplitude.



(a) $t = 0.0$ (b) $t = 0.5$ (c) $t = 1.5$ (d) $t = 2.0$ (e) $t = 2.5$ (f) $t = 4.0$ (g) $t = 5.0$ (h) $t = 8.9$

Figure 13: Simulation of 200 rigid spheres subject to gravity (grid resolution size 64^3). The background colors show the level set amplitude.



(a) $t = 0.0$ (b) $t = 0.5$ (c) $t = 0.8$ (d) $t = 1.2$ (e) $t = 1.5$ (f) $t = 1.8$ (g) $t = 2.0$ (h) $t = 2.4$

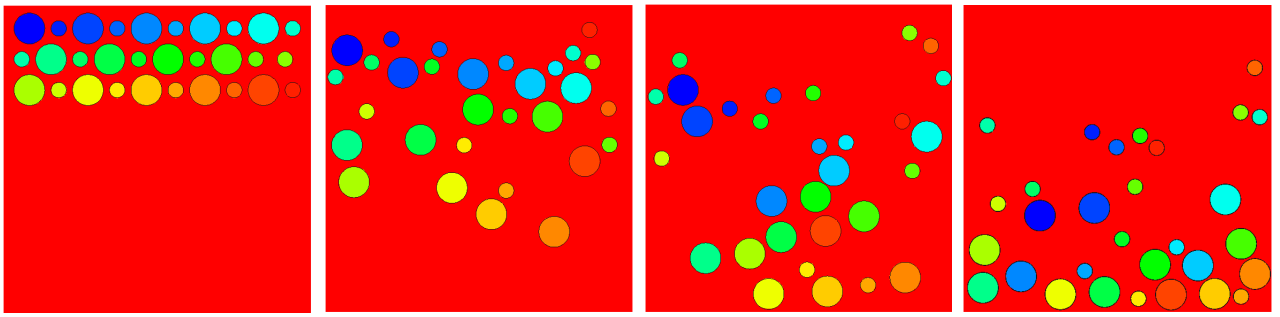
Figure 14: Simulation of 200 rigid spheres subject to gravity (grid resolution size 128^3). The background colors show the level set amplitude.

6.5 Dynamics of rigid bodies of various radii

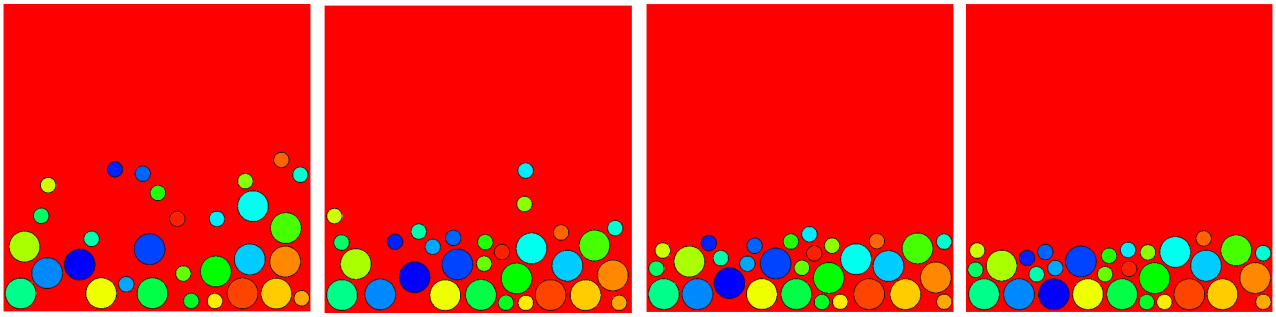
In this section is shown the dynamics of rigid bodies of various diameter. The simulations deal with the sedimentation of 30 rigid bodies of radii $R = 0.05$ and $R = 0.025$ in two dimensions. The simulations are performed on a grid of size (512×512) and the half thickness of the interface is $\varepsilon = 1.5\Delta x$. The coefficient of gravity g is set to -980. The repulsive coefficients are:

$$k_x = -g/10, \quad k_y = -g/10, \quad k_x^{\text{wall}} = -g/40, \quad k_y^{\text{wall}} = -g.$$

The results are represented in Figure 15, the colors represent the different values of the label map L_0. As in section 2.3.1 the first body is dark blue and the fluid is red. It appears that the repulsive forces are well taken into account even if the difference between the size of the bodies is important. Indeed, there is no merging of small and big bodies although, due to the distance function φ_2, the force of a big cell on a small one is effective on the whole boundary ring of the small cell whereas the force of a small cell on a big one is effective only on a part of the boundary ring.
In Figure 16 is plotted the vorticity field inside the fluid. At first the vorticity is created by the falling of the bodies whereas, later, the vortices convect the bodies. In particular this can be seen at time $t = 0.15$ for the three small cells (colors: light blue, orange and light green) at the top right of the figure. Indeed, the orange cell goes up from time $t = 0.1$ to time $t = 0.15$ driven by the fluid flow structure in this part of the domain.

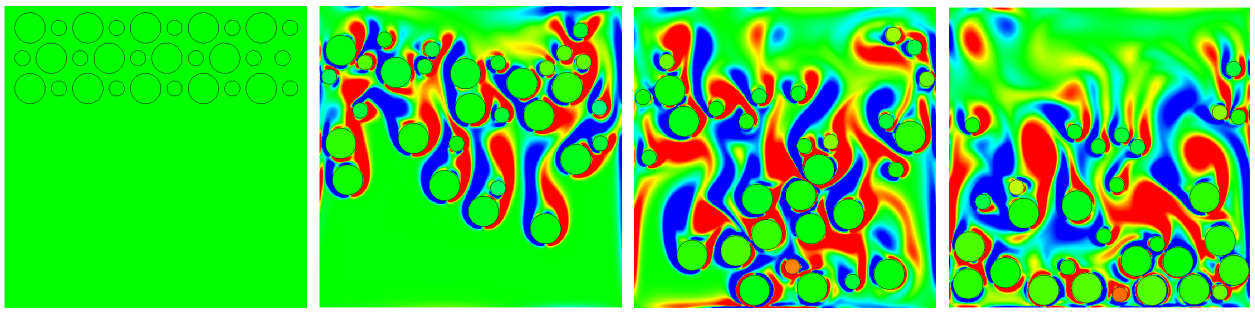


(a) $t = 0$ (b) $t = 0.1$ (c) $t = 0.15$ (d) $t = 0.2$

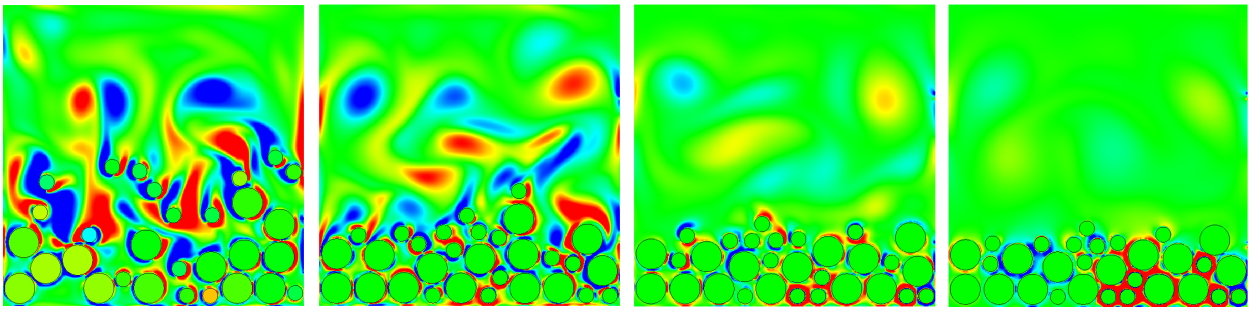


(e) $t = 0.25$ (f) $t = 0.3$ (g) $t = 0.4$ (h) $t = 0.6$

Figure 15: Simulation of 30 rigid bodies of different radii ($R = 0.05$ or $R = 0.025$) falling under gravity. The colors indicate the values of the label map L_0 from dark blue for the first body to dark orange for the 30^{th} body and red for the fluid that is the 31^{th} object.



(a) $t = 0$ (b) $t = 0.1$ (c) $t = 0.15$ (d) $t = 0.2$



(e) $t = 0.25$ (f) $t = 0.3$ (g) $t = 0.4$ (h) $t = 0.6$

Figure 16: Simulation of 30 rigid bodies of different radii ($R = 0.05$ or $R = 0.025$) falling under gravity. The colors indicate the vorticity level from dark blue for -200 and dark red for 200.

7 Conclusions

In this work, we introduced a new model to simulate efficiently a large number of interacting cells immersed in a fluid. This model involved three label maps and two distance functions which allow to locate the bodies and their closer neighbours in the domain. A collision model depending on the distance between the closest cells is proposed. This model which is totally independent on the number of bodies, is compared both theoretically and numerically to the model introduced in [4]. An application to rigid structures is presented with a penalisation model that only depends on five advected field functions.
Numerical results are in good agreement with the results of the literature at least qualitatively. Compared to a model which is totally dependent on the number of cells, our model substancially reduces the CPU time. A numerical test on cells of various radii shows that the collision model is efficient even when the strength of the force is very different.
In future work, this model will be applied to elastic bodies.

References

[1] P. Angot, C.-H Bruneau, and P. Fabrie. A penalization method to take into account obstacles in incompressible viscous flows. *Numerische Mathematik*, 81(4):497–520, 1999.

[2] J. A Bogovic, J. L Prince, and P-L Bazin. A multiple object geometric deformable model for image segmentation. *Computer Vision and Image Understanding*, 117(2):145–157, 2013.

[3] C. Conca, A. Osses, and J. Planchard. Added mass and damping in fluid-structure interaction. *Computer methods in applied mechanics and engineering*, 146(3):387–405, 1997.

[4] M. Coquerelle and G.-H. Cottet. A vortex level set method for the two-way coupling of an incompressible fluid with colliding rigid bodies. *Journal of Computational Physics*, 227(21):9121–9137, 2008.

[5] G.-H Cottet and E. Maitre. A level-set formulation of immersed boundary methods for fluid–structure interaction problems. *Comptes Rendus Mathematique*, 338(7):581–586, 2004.

[6] G.-H Cottet and E. Maitre. A level set method for fluid-structure interactions with immersed surfaces. *Mathematical models and methods in applied sciences*, 16(03):415–438, 2006.

[7] J Donea, S Giuliani, and JP Halleux. An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions. *Computer methods in applied mechanics and engineering*, 33(1):689–723, 1982.

[8] M. A. Fernández. Coupling schemes for incompressible fluid-structure interaction: implicit, semi-implicit and explicit. *SeMA Journal*, 55(1):59–108, 2011.

[9] M. A. Fernández. Incremental displacement-correction schemes for incompressible fluid-structure interaction. *Numerische Mathematik*, 123(1):21–65, 2013.

[10] M. A Fernández, J.-F Gerbeau, and C. Grandmont. A projection algorithm for fluid–structure interaction problems with strong added-mass effect. *Comptes Rendus Mathematique*, 342(4):279–284, 2006.

[11] M. A. Fernández, M. Landajuela, and M. Vidrascu. Fully decoupled time-marching schemes for incompressible fluid/thin-walled structure interaction. *Journal of Computational Physics*, 297:156–181, 2015.

[12] M. A. Fernández, J. Mullaert, and M. Vidrascu. Generalized robin–neumann explicit coupling schemes for incompressible fluid-structure interaction: Stability analysis and numerics. *International Journal for Numerical Methods in Engineering*, 101(3):199–229, 2015.

[13] M.A. Fernández, J.-F. Gerbeau, and C. Grandmont. A projection semi-implicit scheme for the coupling of an elastic structure with an incompressible fluid. *International Journal for Numerical Methods in Engineering*, 69(4):794–821, 2007.

[14] R. Glowinski, T-W Pan, Todd I Hesla, and Daniel D Joseph. A distributed lagrange multiplier/fictitious domain method for particulate flows. *International Journal of Multiphase Flow*, 25(5):755–794, 1999.

[15] C. Grandmont and Y. Maday. Fluid-structure interaction: a theoretical point of view. *Revue européenne des éléments finis*, 9(6-7):633–653, 2000.

[16] M. Hillairet. Lack of collision between solid bodies in a 2d incompressible viscous flow. *Communications in Partial Differential Equations*, 32(9):1345–1371, 2007.

[17] J. Hron and S. Turek. *A monolithic FEM/multigrid solver for an ALE formulation of fluid-structure interaction with applications in biomechanics*. Springer, 2006.

[18] Howard H. Hu. Direct simulation of flows of solid-liquid mixtures. *International Journal of Multiphase Flow*, 22(2):335–352, 1996.

[19] J. Janela, A. Lefebvre, and B. Maury. A penalty method for the simulation of fluid-rigid body interaction. In *ESAIM: Proceedings*, volume 14, pages 115–123. EDP Sciences, 2005.

[20] A. Lefebvre. Fluid-particle simulations with freefem++. In *ESAIM: Proceedings*, volume 18, pages 120–132. EDP Sciences, 2007.

[21] B. Maury. A many-body lubrication model. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, 325(9):1053–1058, 1997.

[22] B. Maury. Direct simulations of 2d fluid-particle flows in biperiodic domains. *Journal of computational physics*, 156(2):325–351, 1999.

[23] B. Maury. A time-stepping scheme for inelastic collisions. *Numerische Mathematik*, 102(4):649–679, 2006.

[24] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *Journal of computational physics*, 79(1):12–49, 1988.

[25] NA Patankar. A formulation for fast computations of rigid particulate flows. *Center for Turbulence Research Annual Research Briefs*, 2001:185–196, 2001.

[26] C. S Peskin. The immersed boundary method. *Acta numerica*, 11:479–517, 2002.

[27] Th Richter and Th Wick. Finite elements for fluid–structure interaction in ale and fully eulerian coordinates. *Computer Methods in Applied Mechanics and Engineering*, 199(41):2633–2642, 2010.

[28] E. Rouy and A. Tourin. A viscosity solutions approach to shape-from-shading. *SIAM Journal on Numerical Analysis*, 29(3):867–884, 1992.

[29] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[30] N. Sharma and N. A. Patankar. A fast computation technique for the direct numerical simulation of rigid particulate flows. *Journal of Computational Physics*, 205(2):439–457, 2005.

[31] E. Sifakis and G. Tziritas. Moving object localisation using a multi-label fast marching algorithm. *Signal Processing: Image Communication*, 16(10):963–976, 2001.

[32] M. Y. Wang and X. Wang. "color" level sets: a multi-phase method for structural topology optimization with multiple materials. *Computer Methods in Applied Mechanics and Engineering*, 193(6):469–496, 2004.

[33] T. Wick. Fluid-structure interactions using different mesh motion techniques. *Computers & Structures*, 89(13):1456–1467, 2011.