Advances in Engineering Software 70 (2014) 51–62

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

Finite Element/Fictitious Domain programming for flows with particles made simple

Marcos Lage [a,*], Lhaylla Crissaff [b], Hélio Lopes [c], Márcio da Silveira Carvalho [d]

[a] Instituto de Computação, Universidade Federal Fluminense, Brazil
[b] Instituto de Matemática, Universidade Federal Fluminense, Brazil
[c] Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Brazil
[d] Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro, Brazil

ARTICLE INFO

Article history:
Received 28 November 2012
Received in revised form 25 September 2013
Accepted 9 January 2014
Available online 7 February 2014

Keywords:
Particulate flows
Numerical simulation
Finite element method
Fictitious Domain
Topological data structures
Lagrange multipliers

ABSTRACT

Flows with suspended particles is a challenging task and important in many applications such as sedimentation, rheology and fluidized suspensions. The coupling between the suspending liquid flow and the particles' motion is the central point in the complete understanding of the phenomena that occur in these applications. Finite Element/Fictitious Domain is an important class of method used to solve this problem. In this work we propose a simple object oriented implementation for simulations of flows with suspended particles in the plane using the Fictitious Domain method together with Lagrange multipliers to solve the Navier–Stokes and rigid body equations with a fully implicitly and fully coupled Finite Element approach. To have an efficient implementation for Fictitious Domain/Finite Element method, we introduce a new topological data structure that is concise in terms of storage and very suitable for searching the elements of the mesh intersected by the particles.

1. Introduction

Flows with suspended rigid particles occur in many different applications, from sedimentation problems to the manufacturing of ordered monolayer of micro and nano particles. The evolution of the particles position is a central point in the complete understanding of the flow of these suspensions. One way of analyzing the flow is to use a moving grid to discretize the Navier–Stokes equations that wraps around all suspended particles. This method requires remeshing and is not very efficient in the case of many particles. An alternative is the *Fictitious Domain* approach that was first proposed by Glowinski et al. [1], latter improved by Goano et al. [2] and recently extended to handle particle flotation by Lage et al. [3]. The Navier–Stokes equations are solved on a fixed mesh and the particles move over it. This approach avoids the need to remesh around the rigid particles, solving the entire problem on a single fixed mesh. The Navier–Stokes equations are solved on the entire domain, but inside each particle, the flow is constrained to be a rigid body motion using Lagrange multipliers.

The implementation of particle flows methods based on Fictitious Domain/Finite Element is quite complex, since the degrees of freedom associated with each individual particle (linear and angular velocities) cannot be written in terms of elemental degrees of freedom. Moreover, the governing equations changes as the overlap of particles over finite elements varies during the flow. Therefore, an efficient implementation requires a topological data structure to optimize the searching procedure of the mesh elements intersected by the suspended particles in the fluid flow simulation. This detection is fundamental to identify where the Lagrange multipliers are active or not in the Fictitious Domain/Finite Element method [2].

Contributions: In this work, we present an object-oriented implementation for the 2D Fictitious Domain/Finite Element method using non-structured meshes with triangular and quadrangular elements. The use of object-oriented programming (OOP) has been recognized as an useful coding technique for scientific computing [4–7] and in particular for Finite Element Methods [8–16]. This implementation uses important concepts of OOP, such as modularization, inheritance and encapsulation. As a consequence we provide a simple interface to the data structures and algorithms required to simulate particle flows.

To have an efficient implementation for the 2D Fictitious Domain/Finite Element method using Lagrange multipliers, we also propose a new topological data-structure, called Compact Tri-Quad Representation, that gives a suitable balance between time and memory complexity for modeling a hybrid 2D mesh. Other data structures have been proposed in the literature in order to reduce the memory use on Finite Element Method implementations [17].

* Corresponding author. Tel.: +55 2126295643.
E-mail addresses: mlage@ic.uff.brr (M. Lage), lhaylla@vm.uff.br (L. Crissaff), lopes@inf.puc-rio.br (H. Lopes), msc@puc-rio.br (M. da Silveira Carvalho).

0965-9978/$ - see front matter © 2014 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.advengsoft.2014.01.003

The Compact Tri-Quad Representation's main characteristic is the conciseness in terms of storage without loosing its efficiency to answer adjacency and incidence relations queries. To answer such queries among the topological entities of the mesh (faces, edges and vertices), this data structure uses integer tables (implemented as containers) and integer formulas to implicitly represent some adjacency and incidence relations.

Paper outline: The remainder of this work is organized as follows. Section 2 describes the Ficticious Domain formulation for fluid flow simulations with suspended particles. Section 3 presents the Compact Tri-Quad Representation data structure. Section 4 proposes the object oriented framework for a simple implementation for the 2D Fictitious Domain/Finite Element method. Section 5 shows the results. Finally, Section 6 concludes this work by making some remarks and suggestions for future developments.

2. Fictitious Domain formulation of flows with particles

Notation: Consider a two-dimensional bounded domain Ω with external boundary $\partial\Omega$, which is filled with Newtonian incompressible fluids and suspended solid particles. Whenever the context is clear, we call *fluid* one or more immiscible, Newtonian and incompressible fluid phases filling the simulation domain Ω.

We denote $\Omega_f = \bigcup_{f_i=1}^{n_f} \Omega_{f_i}$ the region of Ω occupied by n_f fluid phases $f_i \in \{1 \dots n_f\}$ with densities ρ_{f_i} and viscosities μ_{f_i}, and we denote $\Omega_p = \bigcup_{p_i=1}^{n_p} \Omega_{p_i}$ the region of Ω covered by n_p rigid particles $p_i \in \{1 \dots n_p\}$ with densities ρ_{p_i} and radius R_{p_i}. We also represent the interface between fluid and particles by $\partial\Omega_p = \bigcup_{i=1}^{n_p} \partial\Omega_{p_i}$, where $\partial\Omega_{p_i}$ represents the boundary of the particle p_i and we denote the interface between two fluid phases f_i and f_j by $\partial\Omega_{f_{ij}}$. Finally, we observe that $\Omega = \Omega_f \bigcup \Omega_p$. Fig. 1 sketches a typical simulation scenario with a two dimensional domain filled by two fluid phases and one suspended solid particle.

Formulation review: In this paper, we propose a programming approach to simulate flows with suspended rigid body particles using the formulation presented by Lage et al. [3]. Here, we briefly review the differential formulation of the problem. Let us define the velocity field $\vec{u}_p$ to be a rigid body velocity inside each particle p_i and zero in the fluid region Ω_f, i.e.:

$$\vec{u}_p = \begin{cases} \vec{U}_{p_i} + \omega_{p_i} \times (\vec{x} - \vec{X}_{p_i}) & \text{in } \Omega_{p_i} \text{ with } p_i \in (1 \dots n) \\ 0 & \text{in } \Omega_f \end{cases} \quad (1)$$

The integral momentum equation for $\vec{u}_p$ restricted to Ω_{p_i} can be written as:

$$\int_{\Omega_{p_i}} \rho_{p_i} \frac{\mathrm{D}\vec{u}_p}{\mathrm{D}t} d\Omega_{p_i} = \int_{\Omega_{p_i}} \rho_{p_i} \vec{g} d\Omega_{p_i} + \int_{\partial\Omega_{p_i}} \vec{n}_{p_i} \cdot \boldsymbol{\sigma}_f \; d\partial\Omega_{p_i} + \vec{b} \quad (2)$$

where $\vec{n}_{p_i}$ is the outward normal to $\partial\Omega_{p_i}$ and $\vec{b}$ represents the capillary, repulsion or any other body force acting on the particle.

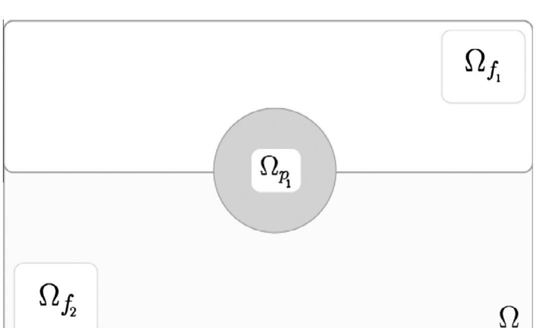

Fig. 1. Sketch of a simulation scenario: two immiscible fluid phases Ω_{f_1} and Ω_{f_2} filling a two-dimensional box Ω and one suspended particle covering the region Ω_{p_1}.

Assuming that the liquid is Newtonian, the stress tensor $\boldsymbol{\sigma}_f$ can be extended over the entire domain Ω. Such extension can always be done if we define $\vec{u}$ and p to be extensions over Ω of the velocity and pressure fields.

Using this extended stress tensor we can apply the divergence theorem and rewrite Eq. (2) as:

$$\int_{\Omega_{p_i}} \rho_{p_i} \frac{\mathrm{D}\vec{u}_p}{\mathrm{D}t} d\Omega_{p_i} = \int_{\Omega_{p_i}} \rho_{p_i} \vec{g} d\Omega_{p_i} + \int_{\Omega_{p_i}} \nabla \cdot \boldsymbol{\sigma} \; d\Omega_{p_i} + \vec{b}$$

Now, if we adopt the following notation:

$$\vec{F} = \begin{cases} -\rho_f \frac{\mathrm{D}\vec{u}}{\mathrm{D}t} + \mu \triangle \vec{u} & \text{in } \Omega_{p_i} \text{ with } p_i \in (1 \dots n) \\ 0 & \text{in } \Omega_f \end{cases}$$

together with an additional constraint to the extended velocity field $\vec{u}$ that imposes $\vec{u} = \vec{U}_{p_i} + \omega_{p_i} \times (\vec{x} - \vec{X}_{p_i})$ in Ω_{p_i} and the fact the particle's shape is circular, we can write the equation for the particle's velocity $\vec{U}_{p_i}$:

$$\int_{\Omega_{p_i}} (\rho_{p_i} - \rho_f) \frac{\partial \vec{U}_{p_i}}{\partial t} d\Omega_{p_i} = \int_{\Omega_{p_i}} \left\{ \rho_{p_i} \vec{g} - \nabla p + \vec{F} \right\} d\Omega_{p_i} + \vec{b}$$

We can recover the angular velocity ω_{p_i} assuming the no-slip boundary condition along the surface of particle $p_i, \vec{u} = \vec{U}_{p_i} + \omega_{p_i} \times (\vec{x} - \vec{X}_{p_i})$ in $\partial\Omega_{p_i}$:

$$\int_{\Omega_{p_i}} \omega_{p_i} d\Omega_{p_i} = \frac{1}{2} \int_{\Omega_{p_i}} \nabla \times (\vec{u} - \vec{U}_{p_i}) \; d\Omega_{p_i}$$

Using the force $\vec{F}$, the extended velocity $\vec{u}$, pressure p and stress tensor $\boldsymbol{\sigma}$ fields, we can write the momentum equation:

$$\rho \frac{\mathrm{D}\vec{u}}{\mathrm{D}t} = \nabla \cdot \boldsymbol{\sigma} + \rho \vec{g} - \vec{F} \text{ in } \Omega$$

We now define a global Lagrange multiplier $\vec{l}$ that is related to $\vec{F}$ through the following boundary value problem:

$$\vec{F} = -\alpha \vec{l} + \mu \triangle \vec{l} \text{ in } \Omega \quad (3)$$

$$\vec{l} = 0 \text{ on } \partial\Omega \quad (4)$$

where α is a positive constant parameter.

The problem defined by Eqs. (3) and (4) is a well posed for $\vec{F}$ and it is more efficient to use its unique solution to impose the rigid-body constraint on the extended velocity field $\vec{u}$ (see Goano et al. [2]). The complete formulation of the flow with suspended particles using the Fictitious Domain method is:

$$\begin{array}{ll} \rho \frac{\mathrm{D}\vec{u}}{\mathrm{D}t} = \nabla \cdot \boldsymbol{\sigma} + \rho \vec{g} + \alpha \vec{l} - \mu \triangle \vec{l} & \text{in } \Omega \\ \nabla \cdot \vec{u} = 0 & \text{in } \Omega \\ \int_{\Omega_{p_i}} (\rho_{p_i} - \rho_f) \frac{\partial \vec{U}_{p_i}}{\partial t} d\Omega_{p_i} = \int_{\Omega_{p_i}} \left\{ \rho_{p_i} \vec{g} - \nabla p - \alpha \vec{l} + \mu \triangle \vec{l} \right\} d\Omega_{p_i} + \vec{b} & \text{in } \Omega_{p_i} \\ \int_{\Omega_{p_i}} \omega_{p_i} d\Omega_{p_i} = \frac{1}{2} \int_{\Omega_{p_i}} \nabla \times (\vec{u} - \vec{U}_{p_i}) \; d\Omega_{p_i} & \text{in } \Omega_{p_i} \end{array} \quad (5)$$

In addition to the system of Eq. (5), the Lagrange multiplier in the fluid domain, the rigid body constraint inside the particle domain and the evolution of the particles' position must also be in the complete formulation:

$$\begin{array}{ll} \vec{l} = 0 & \text{in } \Omega_f \\ \vec{u} = \vec{U}_{p_i} + \omega_{p_i} \times (\vec{x} - \vec{X}_{p_i}) & \text{in } \Omega_{p_i} \\ \frac{\partial \vec{X}_{p_i}}{\partial t} = \vec{U}_{p_i} & \text{for } p_i \in (1 \dots n_p) \end{array}$$

3. *CTQR* Topological data structure

To efficiently handle with the topological relationship between particles and mesh elements, which is the basic query needed to implement fully coupled particle flow simulations based on