Open Access

Simulation of Magnetic Fluid Applied to Plastic Sorting

G. Houzeaux*,[1], C. Samaniego[1], H. Calmet[1], R. Aubry[1], M. Vázquez[1] and P. Rem[2]

[1] *Computer Applications in Science and Engineering Department, Barcelona Supercomputing Center (BSC-CNS), Edificio NEXUS I, Campus Nord UPC Gran Capitán 2-4, 08034 Barcelona, Spain*

[2] *Department of Design & Construction, Delft University, Stevinweg 1, 2628 CN Delft, The Netherlands*

Abstract: The efficient large-scale recycling of plastic waste is of increasing interest from an ecological and economic point of view but it represents a goal that has yet to be achieved by the recycling industry. European project W2Plastics (FP7) aims at a fundamental change of the present status of plastics recycling by applying the Magnetic Density Separation (MDS) technology as well as the Ultrasound Imaging Technology to develop a separation device for the recycling of polyolefin's from complex wastes, i.e., wastes such as Waste from Electric and Electronic Equipment (WEEE), household waste and Automotive Shredder Residue (ASR).

The sorting of plastics in W2Plastics is based on the use of a magfluid, magnetized water, to stratify the different plastics according to their densities and to collect them at different depths in an efficient way. One component of this project consists in simulating the particle paths into the water flow inside the separation device in order to understand the separation mechanisms and to optimize its configuration.

The separation device is divided into three sections. The plastic particles enter the device with a strong turbulent mixing, necessary for an initial good separation between particles. The first section is the laminator section that tries to make the flow laminar in the shortest distance as possible. The second section is the separator section in which the particles are separated according to their densities, under the magnetic field. These particles are flowing to the last section, the collector section where each kind of plastics should flow at a different heights.

From the numerical point of view, there are three main challenges. First, the Navier-Stokes solver should be robust and fast enough. This is accomplished using parallelized schemes together with efficient algebraic solvers, running on thousands of processors. The second challenge is the plastic particles tracking and their interactions with the flow. This will be carried out using a fixed grid method in order to avoid remeshing at each time step. Finally, the turbulent flow will be captured using a Variational Multiscale Method method, and the inlet turbulence will be treated using a synthetic turbulence generation.

Keywords: Navier-stokes equations, incompressible flow, parallelization, plastic sorting, magnetic fluid.

1. INTRODUCTION

1.1. Density Separation Applied to Plastics

We are interested in this work in the recycling of polyolefin waste. Polyolefin fractions are often end fractions resulting from the recycling of cars, waste from electric and electronic equipment (WEEE) and packaging waste. Sink-float, with water as the medium, is the usual process that creates such mixtures of PP and PE as a float fraction. Typical PP:PE ratio range from 70:30 for car scrap to 25:75 for packaging waste. Such mixtures are not suitable for high quality products. For this, the grade of PP and PE should be better than 97%. Several separation techniques are known to achieve this level of purity. Among others, let us mention some common techniques with advantages and drawbacks summed up in [1]. Electrostatic separation; techniques based on the fact that PP and PE have different of melting points; techniques based on near infra red spectroscopy; a sink-float process can give an effective separation with both high grade and recovery if the difference in densities between the materials is large enough.

An alternative is the inverse magnetic density separator (IMDS). This separator achieves a lower apparent density than water by the combination of a gradient magnetic field and a magnetic liquid. The magnetic liquid contains nm-sized ferrite particles and is water based. Magnetic liquid can be used without the economic and environmental problems of organic liquids. With the IMDS it is also possible to use two different cut densities in one step. The present papers propose to describe a numerical algorithm capable of simulation an IMDS.

1.2. Objectives of the Simulation

The sorting of plastics in W2Plastics is based on the use of a magfluid, magnetized water, to stratify different particles according to their density and to collect them at different depths in an efficient way. The particles are made of products and residues. The products are the plastics we want to sort, that is the PP and PE. The residues are any

*Address correspondence to this author at the Computer Applications in Science and Engineering Department, Barcelona Supercomputing Center (BSC-CNS), Edificio NEXUS I, Campus Nord UPC Gran Capit´an 2-4, 08034 Barcelona, Spain; Tel: + 34 93 405 42 88; Fax: +34 93 413 77 21; E-mail: guillaume.houzeaux@bsc.es

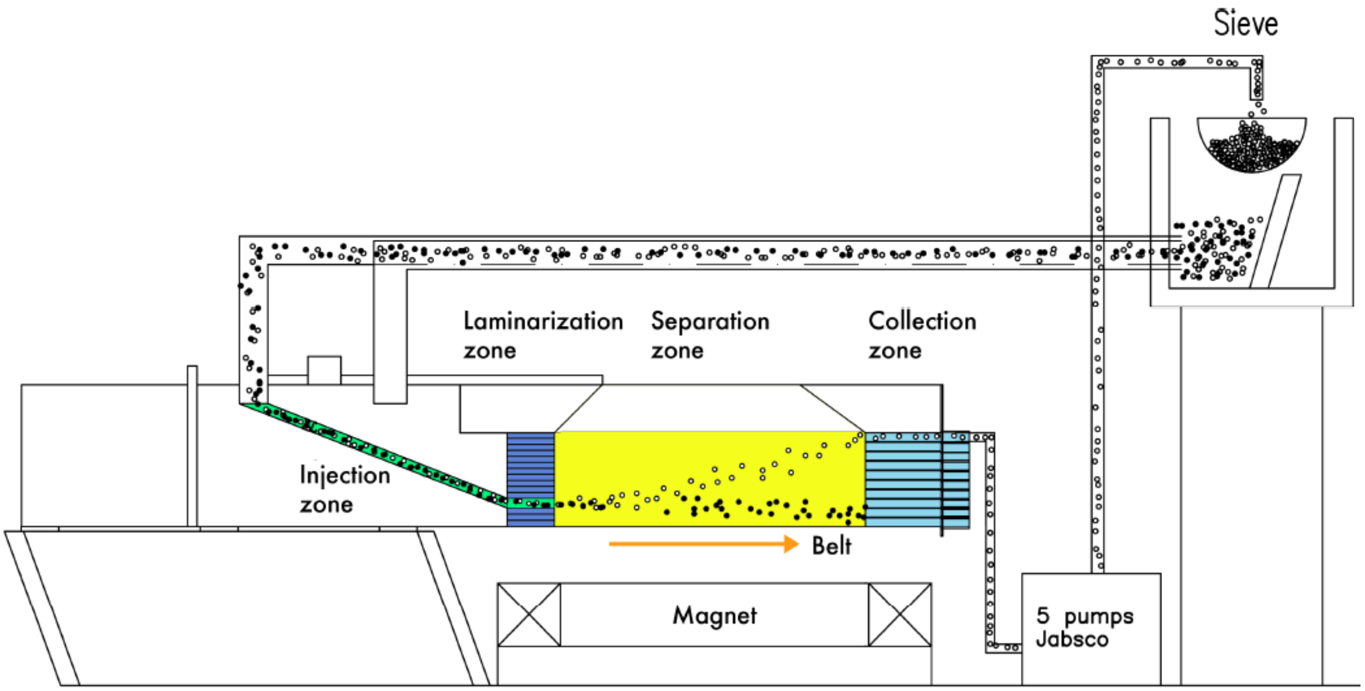

Fig. (1). Prototype.

other material, which can be floating or sinking. The apparatus is shown in (Fig. **1**).

In this introduction we present the different zones of the separator which will be of interest to develop the numerical code. The separation process is based on the separation capacity of the magfluid. Magfluid with particles are injected in the injection zone, the flow is laminarized in the laminarization zone, particle are separated in the separation zone and finally collected in the collection zone. Finally, the belt removes the heavy residues.

The magfluid. The magnetized water is composed of water and magnetite, that is a magnetic mineral. This volume of this mineral should be high enough to create the desired stratification effect on plastics but not too dense to remain a fluid. In fact, if too many magnetites are present in the water, their polarization would lead to a magnetic binding resulting in the formation of solid pockets. To avoid this, they are synthetically surrounded by a surfactant to prevent the magnetite from getting too close from each other. When magnetized, this magnetized water exhibits a non-hydrostatic behavior at rest so that any components floating in it will fell a force depending on its density. See section 2.2.

Particle injection: injection zone. The injection of the particles is a key point. A strong mixing is required at the channel entrance in order to obtain a good separation of plastic particles. If this is not achieved, then groups of bound particles could be transported through the channel without having a chance to be separated. The mixing is obtained upstream using a specific technique (like a mixer, etc.). Usually, the volume of particles which provides a good mixing at injection is around 20% of the total volume. Obviously, this mobility requirement contradicts the collection requirement that the flow should be as laminar as possible to separate smoothly. So the apparatus needs a laminarization zone which should be as short as possible for practical and cost reason.

Flow laminarization: laminarization zone. The role of the laminarization zone is to low down the mixing needed at the particle injection in order to make effective the separation zone in the shortest distance as possible. One proposal consists in dividing this zone into two parts. The first part transports the mixed particle from the injection zone; this is the injection channel. The second part provides a laminar and smooth flow which should attenuate the mixing when put in contact to the first part; this is the laminator, shown in (Fig. **2**). This second part is made of pipes which diameter must be small enough to obtain a small Reynolds number and laminar flow at their outflow, and large enough to avoid too high pressure losses associated with too low outflow velocities. The crucial point to study is how the laminar shear layer is able to dissipate the turbulent eddies coming from the injection channel.

Separation of particles: separation zone. The separation zone involves laminar or slightly turbulent flow to avoid mixing between particles. We in fact know that one property of turbulent flows is the enhance mixing through momentum transfer between fluid layers. Its only requirement is that it is long enough to enable all product particles to reach their equilibrium depth and short enough for practical and cost reasons.

Collection of particles: collection zone. The magfluid enables to distribute the plastic particles according to their density across a given fluid depth. In order to collect them, on possibility is to transport them to a separator device (channels located at different heights). This transport must be smooth enough to prevent any mixing between particles of different densities and is provided by the separation zone.

Another requirement of the collection is that the incoming flow at the channel entrance should also be

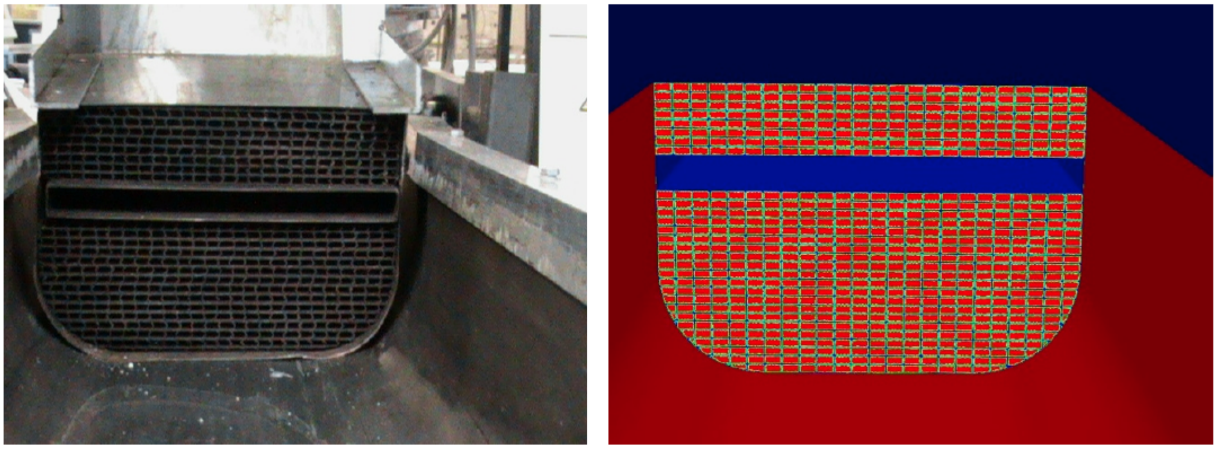

Fig. (2). Laminator. (left) experimental, (right) numerical.

smooth. In fact, when approaching the channel, the streamwise velocity is decreased. Due to mass conservation, this implies a vertical motion of the fluid. This vertical motion is not desirable as it could transport some particles to the wrong collection channel.

Belt: residual removing. A belt at the bottom of the channel is necessary to remove residues. In fact, without this belt, sinking residue would fall into the boundary layer and could not be transported out of the channel.

The numerical model presented in the following will take into account all the physical aspects described here. Some approximations will nevertheless have to be made: for example, the injection zone will not be simulated but its effect will surely be taken into account *via* specific boundary conditions. The free surface at the separation zone will neither be taken into account (see Fig. **3**). The final objective of the simulation is to understand the mechanisms involved in the separation, the interaction of the particles between themselves and the flow, the efficiency of the laminarization zone, etc. Ultimately, this understanding will be helful to optimize the channel and test new configurations.

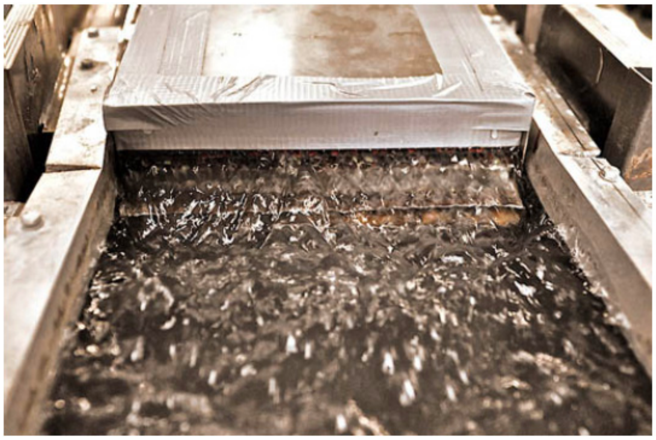

Fig. (3). Free surface at separation zone.

2. MAGFLUID: PHYSICAL PROBLEM

2.1. Governing Equations

In this section, we briefly describe the numerical method used to solve the Navier-Stokes equations for transient incompressible flows (viz. [2]). A deeper description of this method can be found in [3]. Let μ be the viscosity of the fluid, and ρ its constant density. The problem is stated as follows: find the velocity $\boldsymbol{u}$ and mechanical pressure p in a domain Ω such that they satisfy in a time interval

$$\rho\frac{D\boldsymbol{u}}{Dt} - \nabla\cdot[2\mu\varepsilon(\boldsymbol{u})] + \nabla p = p\boldsymbol{f},$$

$$\nabla\cdot\boldsymbol{u} = 0,$$

together with initial and boundary conditions. $D\boldsymbol{u}/Dt$ is the total derivative of the velocity and $\varepsilon(\boldsymbol{u})$ is the velocity strain rate such that

$$\frac{D\boldsymbol{u}}{Dt} := \frac{\partial\boldsymbol{u}}{\partial t} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u},$$

$$\varepsilon(\boldsymbol{u}) := \frac{1}{2}\ (\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^t).$$

The force term $\boldsymbol{f}$ includes the gravity and the magnetic force:

$$\rho\boldsymbol{f} = \rho\boldsymbol{g} + M\nabla B,$$

where B is the magnetic field and M is the magnetization of the fluid.

2.2. Apparent Density

A nice metaphor to understand the effects of the magnetic field consists in introducing the concept of apparent density. Define $\boldsymbol{g} = g\hat{\boldsymbol{g}}$ where g is the norm of the gravity acceleration and $\hat{\boldsymbol{g}}$ the gravity unit vector. If ∇B is aligned with the gravity, we have $\nabla B = (\nabla B\cdot\hat{\boldsymbol{g}})\hat{\boldsymbol{g}}$. Thus we can rewrite the force term as

$$\rho\boldsymbol{f} = \left(\rho + M\frac{\nabla B\cdot\hat{\boldsymbol{g}}}{g}\right)\boldsymbol{g},$$

From this we can define an *apparent density* ρ_{app} such that

$$\rho_{\text{app}} := \rho + M\frac{\nabla B\cdot\hat{\boldsymbol{g}}}{g},$$

This apparent density will affect the force exerted on the particles and will therefore depend of the magnetic field distribution.

From a numerical point of view, it is convenient to redefine the pressure by including the gravity force and magnetic force into a modified pressure (see section on boundary conditions 2.3). In fact we observe that we can redefine the pressure as

$$p^* = p - pg\int \hat{\boldsymbol{g}} - MB,$$

$$\rho\frac{D\boldsymbol{u}}{Dt} - \nabla\cdot[2\mu\varepsilon(\boldsymbol{u})] + \nabla p^* = 0,$$

$$\nabla\cdot\boldsymbol{u} = 0,$$

Therefore, we conclude that, as the gravity force, the magnetic field does not participate to the motion of the fluid, but only affects the pressure distribution. The steady state solution is solution of ∇p^* so that Steady state:

$$p = pg\int \hat{\boldsymbol{g}} - MB,$$

up to a constant.

Note that if we eventually solve for p^*, both the buoyancy and magnetic forces will have to be explicitly taken into account when performing the force balance on the particles, as the fluid pressure does not include it.

2.3. Boundary Conditions

We need boundary conditions for the inflow, for the walls, for the belt and for the outflow. For the walls and the belt, the velocity is prescribed to zero for the former case and to the belt velocity for the latter one.

For outflows, we prescribed the traction to zero where the traction is defined as

Traction := $\boldsymbol{\sigma}\cdot\boldsymbol{n}$

with $\boldsymbol{\sigma}$ being the stress tensor

$$\boldsymbol{\sigma} = -p^*\boldsymbol{n} + 2\mu\varepsilon(\boldsymbol{u})\cdot\boldsymbol{n}$$

For a uniform flow we have $\varepsilon(\boldsymbol{u}) = 0$; for flows with high Reynold numbers, we have $\varepsilon(\boldsymbol{u}) \approx 0$. For these two cases, the prescription of zero traction is therefore (almost) equivalent to prescribing p^* to zero, that is to the steady state (hydro-magnetic static) pressure. From now on and for the sake of clarity, the asterisk superscript will be omitted.

In order to save CPU time, the computational domain does not include the injection zone. So the inflow is located at the injection channel and should therefore include the flow structures generated upstream. That is one cannot impose a constant inflow velocity. In order to take into account the incoming turbulence of the flow, a special strategy based on the synthetic turbulence concept is used [4]:

> *The Synthetic Eddy Method (SEM), is based on the classical view of turbulence as a superposition of coherent structures...*

3. MAGFLUID: NUMERICAL METHOD

We will describe in this section the main ingredients of the numerical strategy: the stabilization based on a Variational MultiScale (VMS) method, namely the orthogonal subgrid scale (OSS) method, the subgrid scale (SGS) tracking in convection and time, the Navier-Stokes solver based on the iterative solution of the pressure Schur complement, and the parallelization of the code. The time discretization is based on the classical trapezoidal rule of first and second order and the linearization is carried out using the Picard method, so no special attention will be paid these two aspects. Let us define $\boldsymbol{a}$ the convection velocity taken from a previous time step and $\partial_t\boldsymbol{u} = (\boldsymbol{u} - \boldsymbol{u}^n)/(\theta\delta t)$, where $\boldsymbol{u}$ is the solution computed at time step $n + \theta$. We choose $\theta = 1$ for first order Euler scheme and $\theta = 1/2$ for the Crank-Nicolson second order scheme.

3.1. Weak Form

Let us start with the weak form. Let $\boldsymbol{v}$ and q be the test functions We introduce the bilinear form $B([\boldsymbol{u},p], [\boldsymbol{v},q])$ such that

$$B([\boldsymbol{u},p], [\boldsymbol{v},q]) := (\rho(\boldsymbol{a}\cdot\nabla)\boldsymbol{u}, \boldsymbol{v}) + (2\mu\varepsilon(\boldsymbol{u}), \varepsilon(\boldsymbol{v})) - (p, \nabla\cdot\boldsymbol{v}) + (q, \nabla\cdot\boldsymbol{u}) \quad (1)$$

The weak form can be written in a compact form as follows: find $[\boldsymbol{u},p]$ in appropriate space such that

$$(p\partial_t\boldsymbol{u}, \boldsymbol{v}) + B([\boldsymbol{u},p], [\boldsymbol{v},q]) = 0$$

for all $[\boldsymbol{v},q]$ in appropriate space. The weak form of the momentum equation can be recovered by simply taking $q = 0$ while that of the continuity equation can be found by taking $\boldsymbol{v} = 0$.

3.2. Stabilized Formulation

The stabilization technique is based on a VMS method, namely the the split OSS method introduced in [5]. This method has similarities with old high order artificial viscosity methods but it is set in the context of a VMS method; this method introduces the SGS as the key for stabilization in a natural way. Let $\boldsymbol{a}$ the convection velocity known from a previous iteration (coming from the Picard linearization method).

The way to devise the split OSS method consists in the following steps:

- Split velocity and pressure into grid and subgrid scale components, $\boldsymbol{u}$ and $\boldsymbol{u}'$ for the velocity and p and p' for the pressure.
- Introduce this splitting in the weak form (1).
- Solve for the SGS equation by approximating the differential momentum and continuity operators by algebraic ones (τ_1 and τ_2).
- Require that the SGS is in the orthogonal space.

Eventually, the *split OSS method* consists in adding three stabilization terms to the Galerkin formulation and reads

$$(p\partial_t\boldsymbol{u}, \boldsymbol{v}) + B([\boldsymbol{u},p], [\boldsymbol{v},q]) + (\rho\boldsymbol{a}\cdot\nabla\boldsymbol{v}, \tau_1\rho\boldsymbol{a}\cdot\nabla\boldsymbol{u} - \Pi_{\text{adv}}) + (\nabla q, \tau_1\nabla p - \Pi_{\text{pre}} + (\nabla\cdot\boldsymbol{v}, \tau_2\nabla\cdot\boldsymbol{u} - \Pi_{\text{div}}) = 0 \quad (2)$$

where

$$\Pi_{\text{adv}} = \Pi\,(\tau_1\rho\boldsymbol{a}\cdot\nabla\boldsymbol{u})$$

$$\Pi_{\text{pre}} = \Pi\,(\tau_1\nabla p)$$

$$\Pi_{\text{div}} = \Pi\,(\tau_2\nabla\cdot\boldsymbol{u})$$

The operator Π is the L_2 projection, obtained by solving

$$\int_\Omega \Pi(u)v\, d\Omega = \int_\Omega uv\, d\Omega.$$

The first additional term Π_{adv} stabilizes the convection; for smooth velocity field, we have $\Pi_{\text{adv}} \approx \tau_1 \rho \boldsymbol{a} \cdot \nabla \boldsymbol{u}$ and the convection does not need to be stabilized. The second term Π_{pre} is the pressure stabilization which enables the use of equal order interpolation. The third term Π_{div} enforces the mass conservation in the momentum equation that was penalized by the pressure stabilization in the continuity equation.

Parameters τ_1 and τ_2 are the so-called stabilization parameters, that is the algebraic approximate inverses of the momentum and continuity equations, respectively [6, 7]. In this work we take [8]:

$$\tau_1 = \left(4\frac{\mu}{h^2} + 2p\frac{|\boldsymbol{a}|}{h}\right)^{-1}$$
$$\tau_2 = h^2 \tau_1^{-1},$$

where h is the element characteristic length.

Including the linearization and time integration the final system to solve is:

$$(\rho \boldsymbol{u}/(\theta\delta t), \boldsymbol{v}) + \mathrm{B}([\boldsymbol{u}, p]), [\boldsymbol{v}, q] + (\rho \boldsymbol{a} \cdot \nabla \boldsymbol{v}, \tau_1 \rho \boldsymbol{a} \cdot \nabla \boldsymbol{u})$$
$$+ (\nabla q, \tau_1 \nabla p) + (\nabla \cdot \boldsymbol{v}, \tau_2 \nabla \cdot \boldsymbol{u}) \tag{3}$$
$$= (\rho \boldsymbol{u}^n/(\theta\delta t), \boldsymbol{v}) + (\rho \boldsymbol{a} \cdot \nabla \boldsymbol{v}, \Pi_{\text{adv}}) + (\nabla q, \Pi_{\text{pre}}) + (\nabla \cdot \boldsymbol{v}, \Pi_{\text{div}})$$

where all terms on the RHS are known from a previous time step or linearization iteration.

3.3. Subgrid Scale Modeling

In addition to the OSS stabilization, the tracking in convection and time of the subgrid scale is taken into account for two purposes: it has been shown that it gives more accurate results [7] and it consists of a mathematical LES model [9]. Therefore, using this strategy, no LES model has to be considered.

The velocity subgrid scale is computed by solving the following non-linear transient equation:

$$p\partial_t \boldsymbol{u}' + \tau_1 - 1\boldsymbol{u}' = -\tau_1 \rho \boldsymbol{a} \cdot \nabla \boldsymbol{u} + \Pi_{\text{adv}},$$

in the following way

$$\left(\frac{\rho}{\theta\delta t} + \tau_1^{-1}\right)\boldsymbol{u}' = \frac{\rho}{\theta\delta t}\boldsymbol{u}'^{\text{n}} - \tau_1 \rho \boldsymbol{a} \cdot \nabla \boldsymbol{u} + \Pi_{\text{adv}}$$

where now the convection velocity includes the velocity SGS: $\boldsymbol{a} \Leftarrow \boldsymbol{a} + \boldsymbol{u}'$ and where the time discretization is carried out using the same trapezoidal rule as the one used for the velocity is used. Note that the non-linearity comes from different terms: τ_1, $\boldsymbol{a}$ and the projection Π_{adv} which includes τ_1 and $\boldsymbol{a}$. In practice, the subgrid scale (which is discontinuous) is computed at any Gauss point in any element.

3.4. Pressure Schur Complement Solver

At each time step, the linearized system

$$\begin{pmatrix} \mathbf{A}_{uu} & \mathbf{A}_{up} \\ \mathbf{A}_{pu} & \mathbf{A}_{pp} \end{pmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} \begin{bmatrix} \mathbf{b}_u \\ \mathbf{b}_p \end{bmatrix}$$

is solved, where **u** and **p** are velocity and pressure unknowns. Four sub-matrices then arise. Matrix $\mathbf{A}_{uu}$ includes the Galerkin as well as the stabilization terms. Matrix $\mathbf{A}_{up}$ includes the Galerkin pressure term. Matrix $\mathbf{A}_{pu}$ includes the velocity divergence operator as well as the part of the pressure stabilization involving the velocity in the momentum residual. Finally, matrix $\mathbf{A}_{pp}$ includes only the pressure stabilization. When this system is solved in one shot using either a direct solver or an iterative solver with preconditioning, the resulting scheme is referred as a *monolithic scheme*. The strategy used here transforms the solution process in a *fractional scheme*.

The method consists in solving the pressure Schur complement by using the Orthomin(1) iterative solver introduced in [10]. It is shown in Algorithm 1.

Algorithm 1. Momentum Preserving Orthomin(1) Iteration

```
1. Solve momentum eqn $\mathbf{A}_{uu}\mathbf{u}^{k+1} = \mathbf{b}_u - \mathbf{A}_{up}\mathbf{p}^k$
2. Compute Schur complement residual
     $\mathbf{r}^k = [\mathbf{b}_p - \mathbf{A}_{pu}\mathbf{u}^{k+1}] - \mathbf{A}_{pp}\mathbf{p}^k$
3. Solve continuity eqn $\mathbf{Qz} = \mathbf{r}^k$
4. Solve momentum eqn $\mathbf{A}_{uu}\mathbf{v} = \mathbf{A}_{up}\mathbf{z}$
5. Compute $\mathbf{x} = \mathbf{A}_{pp}\mathbf{z} - \mathbf{A}_{pu}\mathbf{v}$
6. Compute $\alpha = <\mathbf{r}^k, \mathbf{x}> / <\mathbf{x}, \mathbf{x}>$
7. Update velocity and pressure
     $\begin{cases} \mathbf{p}^{k+1} = \mathbf{p}^k + \alpha\mathbf{z} \\ \mathbf{u}^{k+2} = \mathbf{u}^{k+1} - \alpha\mathbf{v} \end{cases}$
```

The Orthomin(1) algorithm requires two solves of the momentum equations and one for the continuity equation. The system resulting from the momentum equations is unsymmetrical for which GMRES or Bi-CGSTAB algebraic solvers can be used. The system resulting from the continuity equation is symmetric (See Alg. 3.4). A deflated Conjugate gradient [11-13] is used together with a linelet preconditioner [14] in the boundary layers. Matrix **Q** is the preconditioner of the pressure Schur complement based on the weak form of the Uzawa's operator; any details can be found in [10].

Fig. (**4**) shows some convergence histories of the simulation of the hemodynamics in a brain for a 19M tetrahedra mesh. The results are taken from [10]. It compares the momentum preserving Orthomin(1) with the continuity preserving version (not presented in this work) using different safety factors γ for the time step. We observe that the continuity preserving method converges even with very large safety factors. In fact for the other two methods, the continuity residual remains blocked at unity when using the highest safety factor. For $\gamma = 100$, momentum and continuity residuals fall below 2 orders of magnitude in 100 iterations.

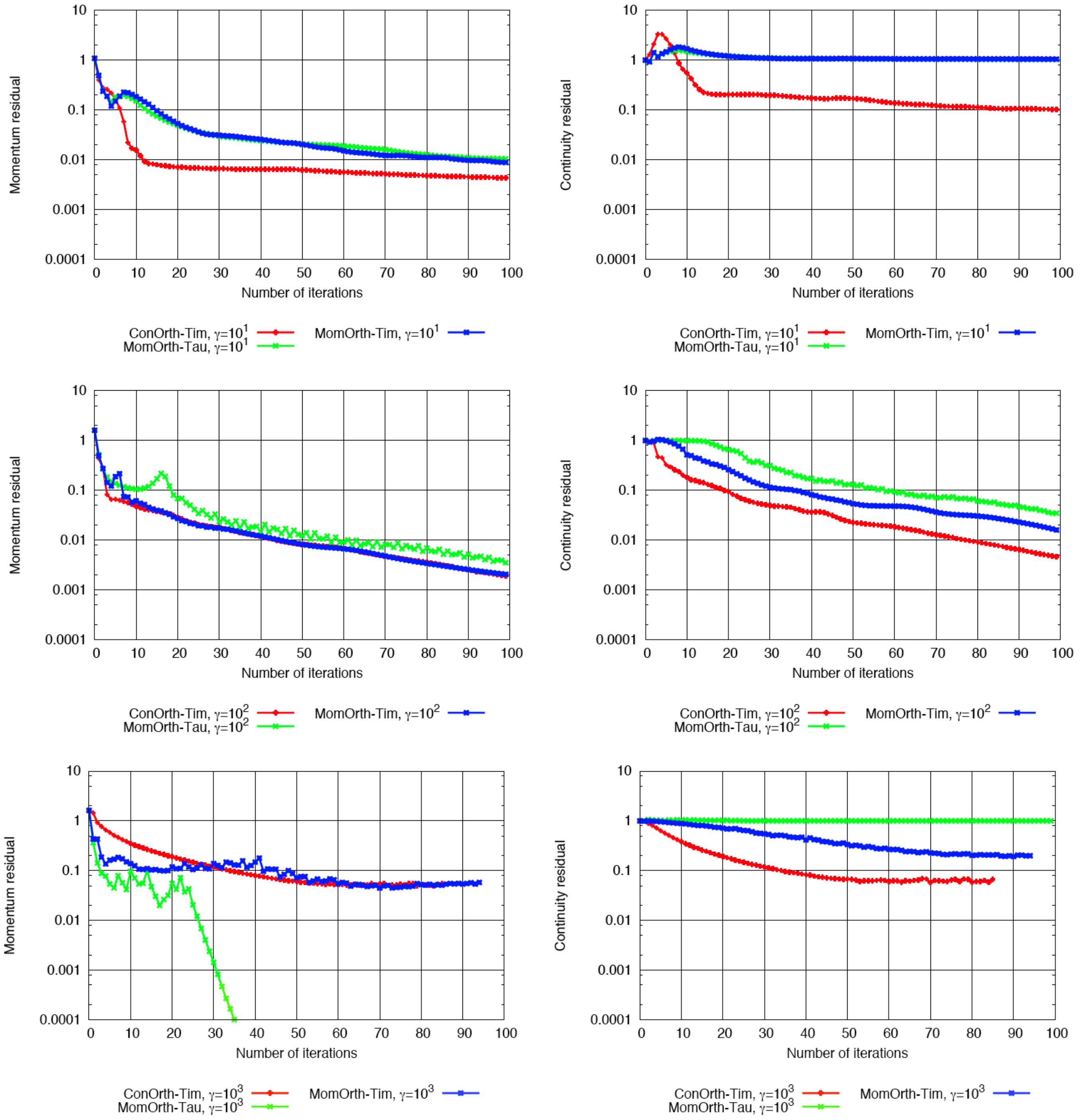

Fig. (4). Typical convergence history of the Orthomin(1) solver. From (Top) to (Bot.) $\gamma = 10^1$, $\gamma = 10^2$, $\gamma = 10^3$. (Left) Momentum residual. (Right) Continuity residual.

3.5. Solution Procedure

The final algorithm is shown in Algorithm 2.

3.6. Parallelization

The Master reads the mesh, performs its partition and dumps some output files (for example of the convergence residuals). The slaves build the local element matrices (LHS) and right-hand side (RHS) and are in charge of the resulting system solution in parallel. Two strategies are possible:

- For small problems (say <15 millions of elements): The Master creates the mesh partition, sends each of the subdomains and supplementary data to the corresponding slaves and launches the simulation.
- For large problems: The Master creates the mesh partition, writes individual restart files for each of the slaves containing its corresponding subdomain and supplementary data and stops the run. Then, a parallel run can start at any time with the Master commanding the Slaves to read the individual restart files. This

strategy is particularly well-suited for very large cases whose partitions are expensive to compute in terms of both CPU time and memory. Moreover, it is very likely that the memory available for one CPU in the distributed memory cluster is not enough, ant pre-process must be done in a different architecture with shared memory like an Altix cluster.

Algorithm 2. Solution Procedure

```
Initial conditions
for Time step do
  while Not converged do
    Assemble matrices and RHS (Eq. (3))
    Solve one Orthomin(1) iteration (Alg. 1)
    Compute velocity subgrid scale $\boldsymbol{u}'$ (Eq. (4))
    Compute projections $\Pi_{adv}$, $\Pi_{pre}$, $\Pi_{div}$, (Eq. (2))
  end while
    Update velocity: $\mathbf{u}^{n+1} = 1/\theta\mathbf{u} + (1-1/\theta)\mathbf{u}^n$
    Update SGS: $\boldsymbol{u}'^{n+1} = 1/\theta\boldsymbol{u}' + (1-1/\theta)\boldsymbol{u}'^n$
end for
```

In a finite element implementation, only two kinds of communications are necessary between subdomains. The first type of communication consists in exchanging arrays between neighbors with MPI_Sendrecv. The strategy is the following:

1. For each slave, compute elemental LHS and RHS for each element.
2. For each slave, assemble (scatter) elemental RHS and LHS into global LHS and RHS.
3. Exchange RHS of boundary nodes, the nodes belonging to more than one subdomain, and sum the contribution.
4. The operations of an iterative solver are matrix-vector multiplications. Then, for each slave, compute matrix-vector multiplication.
5. Exchange the results on the boundary nodes, as was done for the RHS.

The second type of communication is global and of reduce type with MPI_Reduce. It is used to compute:

- The critical time step: it is the minimum over the slaves.
- The convergence residual: the sum over all the nodes of the domain. Residuals are required to check the different convergence tolerances of the scheme.
- Scalar products: they take part of iterative solvers.

All details on the parallelization strategy can be found in [3]. (Figs. **5**, **6**) show two scalability results on thousands of processors from latter reference. The first one compares the speed-up obtained on Marenostrum (MAR) with BG/L and BG/P supercomputers. The second one shows only Marenostrum results. This case first one is a favorable case where the element assembly dominates and the second one the algebraic solvers dominate the CPU time.

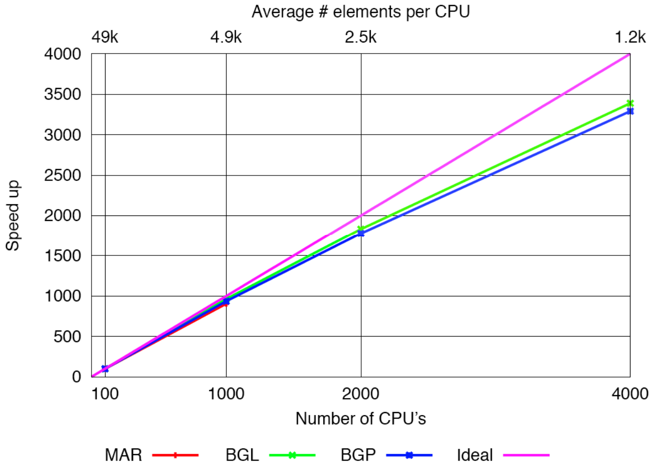

Fig. (5). Speedup for an element assembly dominated case. Comparison of MareNostrum, Blue Gene/L and Blue Gene/P.

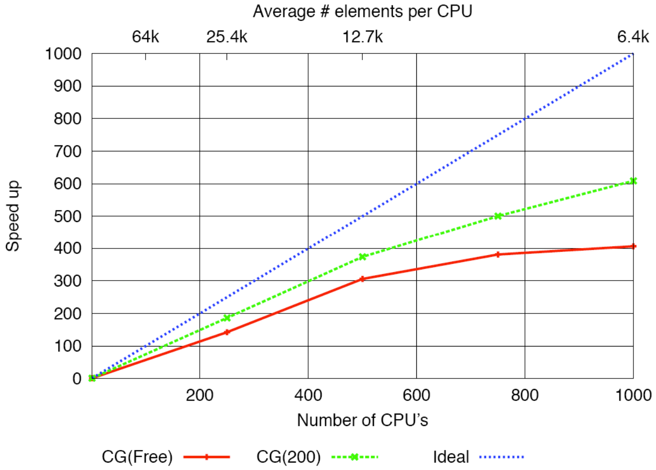

Fig. (6). Speedup for a solver dominated case. Influence of the number of iterations carried out by the CG solver.

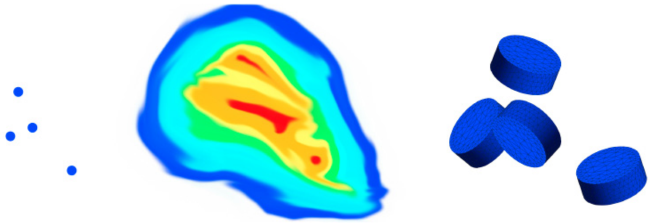

Fig. (7). Different approaches to treat the particles. (Left) Infinitesimal particles. (Mid.) Continuum approach. (Right) Real shaped particles.

4. PARTICLE TREATMENT

There exist mainly three main approaches to treat the particles transport and/or interaction, as illustrated in (Fig. 7). They are:

- **Infinitesimal particle**. The particles are transported by the flow. This is a postprocess issue as the interactions of the particles on the fluid can ``barely" be taken into account.

- **Continuum approximation**. The particles are associated to a continuous medium and one can solve for a partial differential equation describing the particle concentration. However, it is therefore impossible to take into account the dynamics of the particles.
- **Real shaped particles**. The particles are represented through a mesh describing their real shape.

The only way to consider the full dynamics of the particles is to describe their real shapes. Therefore, last option will be considered.

We treat in this section the transport of the rigid particles as well as their interactions with the flow, between each others and with the wall. These points are illustrated in Fig. (**8**) and will now be developed.

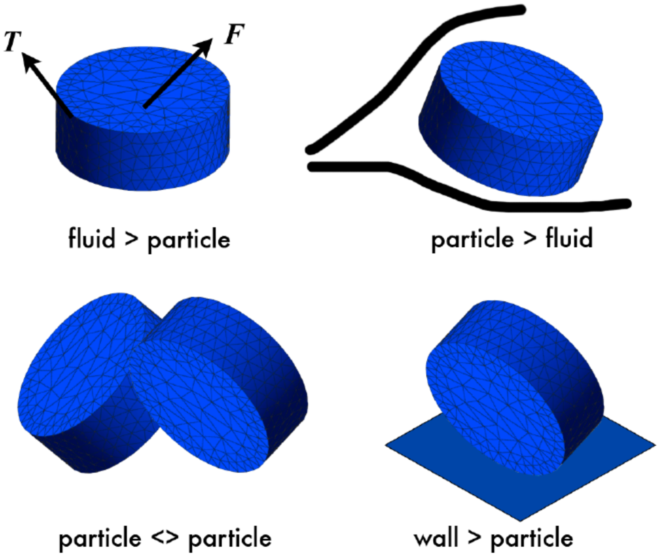

Fig. (8). Particle treatment.

4.1. Particle Properties

Let us first define the properties of the particle defined by the domain Ω_p with boundary Γ_S, shown in Table **1**.

Table 1. Particle Properties

Symbol	Definition
m	Mass (Constant)
ρ_s	Density (Constant)
I	Center of inertia with respect to center of mass
$\boldsymbol{x}, \dot{\boldsymbol{x}}, \ddot{\boldsymbol{x}}$	Linear position, velocity and acceleration of particle center of mass
$\boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}, \ddot{\boldsymbol{\alpha}}$	Angular position, velocity and acceleration of particle
$\boldsymbol{F}$	Total force acting on particle
$\boldsymbol{T}$	Total torque acting on particle wrt center of mass
n_s	Exterior normal to the particle Ω_p with components n_i

In the current numerical implementation, the particles are described by their boundaries Γ_S (boundary mesh). It is therefore convenient to re-express some properties computed as integrals over their volume into integrals over their surface (using the divergence/Gauss theorem).

The particle volume and initial position of center of gravity are computed as:

$$V = \int_{\Omega_S} d\Omega = (1/3) \int_{\Gamma_S} r_i n_i \, d\Gamma,$$

$$x_i = \int_{\Omega_S} \boldsymbol{r} \, d\Omega = (1/2) \int_{\Gamma_S} r_i^2 n_i \, d\Gamma.$$

The tensor of inertia is a symmetric tensor and is defined by

$$\boldsymbol{I} = \int_{\Omega_S} \rho_S (\| \boldsymbol{r} \|^2 \, \boldsymbol{I}_d - \boldsymbol{r} \otimes \boldsymbol{r}) \, d\Omega.$$

From this, we have for each component:

$$I_{11} = (1/3)\rho_S \int_{\Gamma_S} (r_2^3 n_2 + r_3^3 n_3) d\Gamma,$$

$$I_{22} = (1/3)\rho_S \int_{\Gamma_S} (r_1^3 n_1 + r_3^3 n_3) d\Gamma,$$

$$I_{33} = (1/3)\rho_S \int_{\Gamma_S} (r_1^3 n_1 + r_2^3 n_2) d\Gamma,$$

$$I_{12} = (1/4)\rho_S \int_{\Gamma_S} (-r_2 r_1^2 n_1 - r_1 \, r_2^2 n_2) d\Gamma,$$

$$I_{13} = (1/4)\rho_S \int_{\Gamma_S} (-r_3 r_1^2 n_1 - r_1 \, r_3^2 n_3) d\Gamma,$$

$$I_{23} = (1/4)\rho_S \int_{\Gamma_S} (-r_3 r_2^2 n_2 - r_2 \, r_3^2 n_3) d\Gamma,$$

4.2. Fluid > Particle Interaction

The motion of a rigid particle inside the fluid is described by the Euler's equations which describe its linear and angular motion:

$$\text{Linear momentum: } \boldsymbol{F} = \frac{d(m\dot{\boldsymbol{x}})}{dt},$$

$$\text{Angular momentum: } \boldsymbol{T} = \frac{d(\boldsymbol{I}\dot{\boldsymbol{\alpha}})}{dt}.$$

As we saw in Section 2.2, the fluid pressure does not include neither the gravity nor the magnetic force on the particle. The total force on the particle should therefore explicitly take them into account. We have

$$\boldsymbol{F} = \boldsymbol{F}_{\text{flu}} + \boldsymbol{F}_{\text{buo}} + \boldsymbol{F}_{\text{mag}} + \boldsymbol{F}_{\text{gra}},$$

$$\boldsymbol{F}_{\text{flu}} = \int_{\Gamma_S} \boldsymbol{\sigma} \cdot \boldsymbol{n}_s \, d\Gamma,$$

$$\boldsymbol{F}_{\text{buo}} = \int_{\Gamma_S} -\rho g \hat{\boldsymbol{g}} \boldsymbol{n}_s \, d\Gamma = -\rho V \boldsymbol{g},$$

$$\boldsymbol{F}_{\text{mag}} = \int_{\Gamma_S} -M \, B \boldsymbol{n}_s \, d\Gamma,$$

$$\boldsymbol{F}_{\text{gra}} = \rho_s V \boldsymbol{g},$$

so that

$$\boldsymbol{F} = \int_{\Gamma_s} (\sigma - M\, B\boldsymbol{I}_d) \cdot \boldsymbol{n}_s \, d\Gamma + V\boldsymbol{g}(\rho_s - \rho),$$

where $\boldsymbol{I}_d$ is the identity matrix. Gravity and buoyancy do not exert any torque on the particle so that we are left with the torques exerted by the fluid and the magnetic field:

$$\boldsymbol{T} = \int_{\Gamma_s} (\boldsymbol{r} - \boldsymbol{x}) \times (\boldsymbol{\sigma} - M\, B\boldsymbol{I}_d) \cdot \boldsymbol{n}_s.$$

The solution of the linear momentum equation (5) is straightforward. The angular momentum is more tricky as it is non-linear and time dependent, the tensor of inertia depending on the position α. A way to circumvent this is to express the angular motion in the principal axes coordinate system (notation with asterisk superscript). The resulting equations are solved using a Newmark scheme with variable time step.

4.3. Particle > Fluid Interaction

There exists different ways of treating the particle > fluid interaction, each one with advantages and drawbacks. Most common methods are illustrated in (Fig. **9**), taken from [15].

We select here three of these methods that can be applied to our problem, namely the ALE, Chimera and mebedded mesh methods. They are:

- **ALE** (Fig. **9**, Mid., Right). The mesh is updated at each time step to follow the particles, by moving nodes or by locally regenerating the mesh when this one gets too distorted. Advantage: the shape of the particle is represented up to the mesh size. Drawbacks: in a parallel context, it is a costly task as mesh partitioning should be performed to rebalance the load for each new mesh and the communication strategy should be updated.
- **Chimera** (Fig. **9**, Top, Right). Independent body fitted meshes are generated for each particles as well

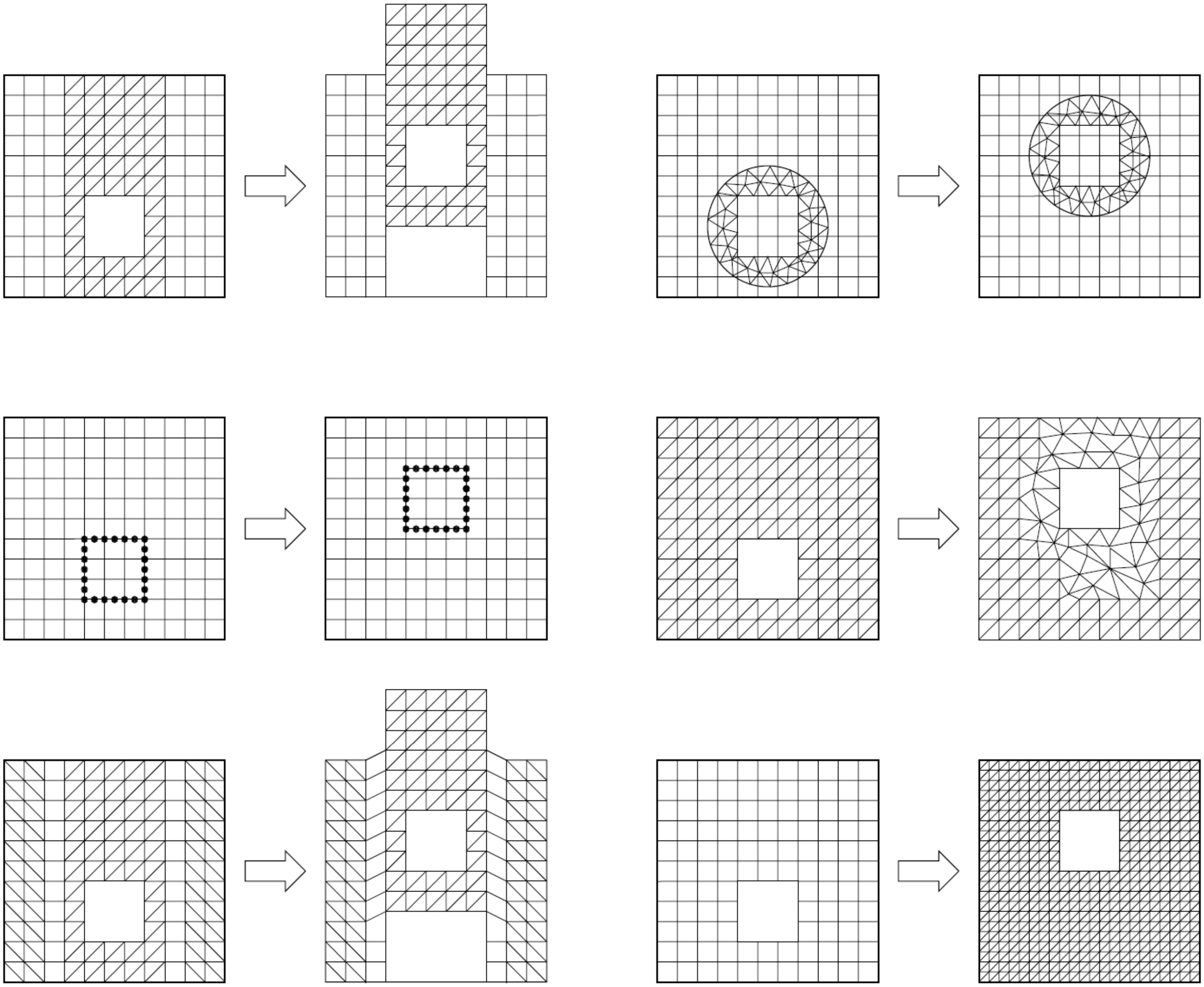

Fig. (9). Illustration of some methods to simulate flows around moving components. (Top) (Left) Multi-domain method without overlap: sliding mesh method. (Top) (Right) Multi-domain method with overlap: Chimera method. (Mid.) (Left) Embedded mesh method. (Mid.) (Right) ALE method: automatic remeshing. (Bot.) (Left) ALE method: SSMUM. (Bot.) (Right) ALE method: FMALE.

as a backgroudn mesh for the separation apparatus. Advantage: a boundary layer mesh can initially be generated around each particle and even the drag forces can be accurately computed. Drawback: in a parallel context, it is a costly task as the mesh partitioning should be performed at almost each time step, and the communication strategy should be updated as the interpolation from one mesh to another changes as well.

- **Embedded Mesh** (Fig. **9**, Mid., Left). These methods consider a bacground mesh on which boundary meshes are patched onto it. These methods can be divided into two groups. In the first group, the boundary conditions are taken into account through a force term added to the flow equations. In the second group, the boundary conditions are approximated, having previously identified the elements where the solid boundary falls. The following description is absolutely not formal. Advantage: it is well suited for parallel implementation, if no addition degrees of freedom are introduced. Drawbacks: the particle geometry is not well described from the fluid point of view and viscous forces may not be computed accurately.

For the advantages and drawbacks mentioned earlier, the embedded mesh method is the candidate. In this family, let us mention the Fictitious Domain method [16, 17], the Immersed Boundarymethod [18, 19] and the method introduced in [20]. The latter one has the advantage that no addition degree of freedom is introduced. It looks for a solution that minimizes (in a weak sense) the difference between the solution and the real boundary condition under the restriction that the solution satisfies the Navier-Stokes equations. In addition, a specific fixed mesh ALE treatment (FMALE) is need for moving objects [21].

4.4. Particle <> Fluid and Wall > Particle Interactions

To solve the problem of collision between particles *A* and *B*, we have to carry out two main tasks:

- Collision detection (pseudo-dynamic) to prevent penetrations between objects:
  - Divide time step into pseudo time steps to identify possible collision.
  - Find contact time.
  - Find contact points for vertex-face and edge-edge contacts.
- Collision response using the equations for an elastic collision:
  - Compute the contact unit normal vector:
    - * Face-vertex case: normal of the face.
    - * Edge-edge case: cross-product of the direction vectors of the two edges.
  - Write post-impulse linear and angular velocities of *A* and *B* using conservation of linear and angular momenta in term os impulse magnitude.
  - Compute relative velocity for pre and post-impulse.
  - Apply the empirical law for friction-less collisions states: the relative velocity only changes sign for dissipation-free collisions.
  - Correct linear and angular velocities.

The wall > particle interaction can be carried out in the same way by assigning to the wall infinite mass and inertia tensor components.

5. RESULTS

In this section, preliminary results are presented. They will be helful to calibrate the simulation parameters and to have an insight on the cost of the complete simulation which will include the particle transport and interactions.

A simulation to reach the steady state of the separation prototype has been carried on Marenostrum supercomputer using 64 processors (1 master + 63 slaves). This run enables

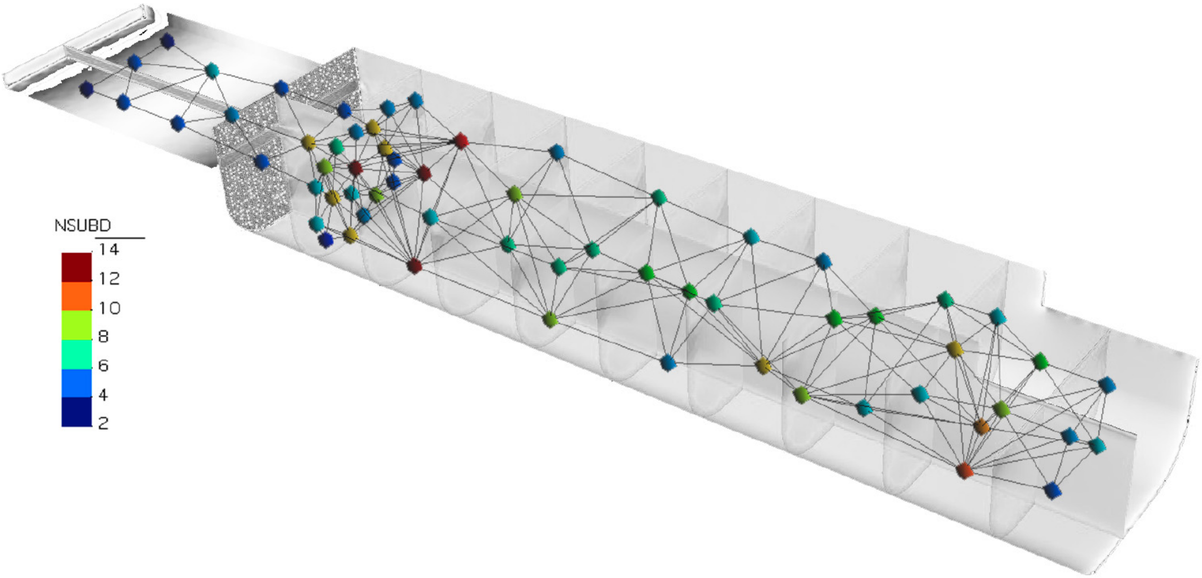

Fig. (10). Partition.

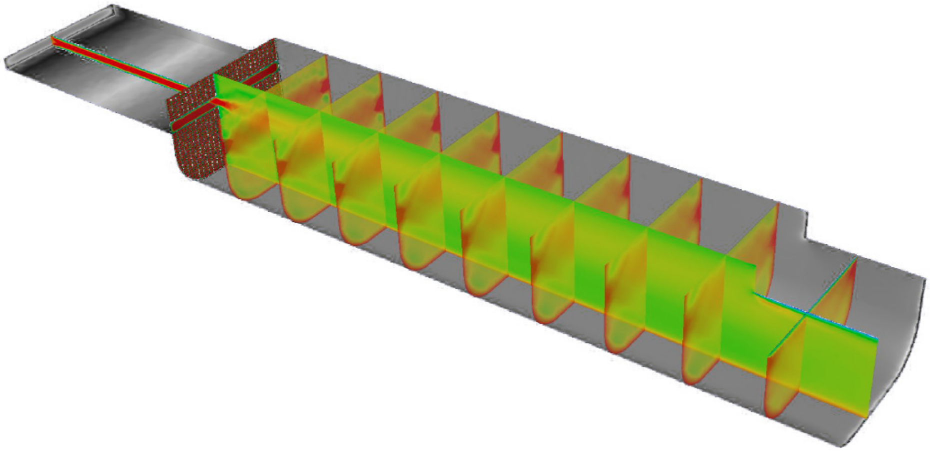

Fig. (11). Velocity contours.

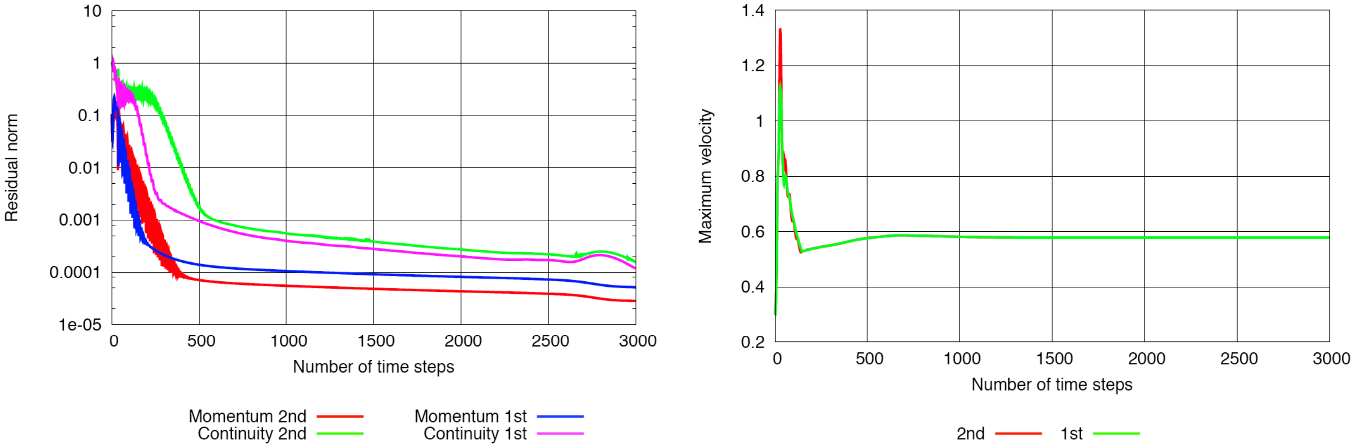

Fig. (12). Convergence. (Left) Residual norm. (Right) Maximum velocity.

to test the robustness of the algorithm and to get an insight of the CPU time would cost a real simulation. The mesh is made of approximately 3M tetrahedra elements. Only one linearization iteration is carried out per time step. Both the first order and second order time scheme will be compared.

The partition of the mesh is illustrated in (Fig. **10**) and shows the number of neighbors for each subdomains. We observe a large disparity in the number of adjacent subdomains due to the particularity of the geometry. The averaged number of elements by subdomain is around 47000. The contours of the velocity are shown in (Fig. **11**).

We now analyze the convergence of the algorithm, shown in (Fig. **12**). By observing the residual norm and maximum velocity, the steady state is reached after 1000 iterations. The first order time scheme exhibits are slightly higher rate of convergence.

Fig. (**13**) shows the number of iterations carried out by the momentum and continuity solvers. As an average, the momentum solver does 3 iterations and the continuity solver 33. Note that there are twice momentum iterations (6000) as continuity iterations (3000) as one Orthomin(1) iteration solves twice for the momentum per time step.

6. CONCLUSION

A numerical algorithm has been presented to simulate a separation by density device applied to plastics recycling. The numerical algorithm takes into account not only the flow features but also the particle transport and interactions. Preliminary results have been obtained for a steady state simulation. The convergence is fast (two orders of magnitude in momentum and continuity residuals reached in 250 iterations) and the algotihm is robust. The possibility to use a big cluster to run the parallel code is uncircumventable to be able to carry out simulation the full device in operation conditions, which are higly unsteady.

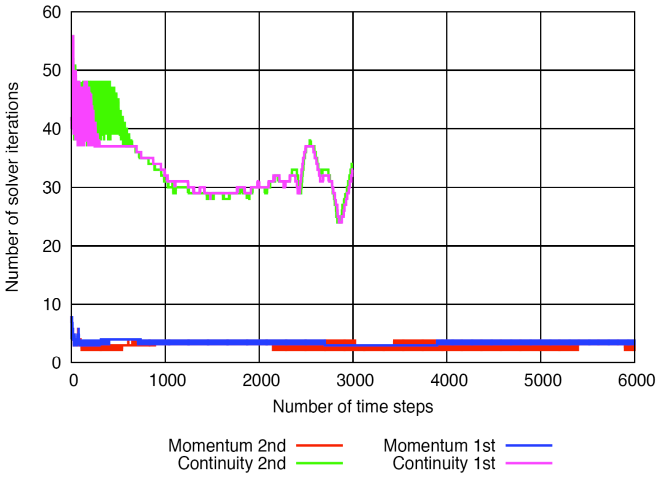

Fig. (13). Solver iterations.

Finally, Fig. (**14**) shows the CPU time elapsed at each iteration which in average is around 2 seconds.

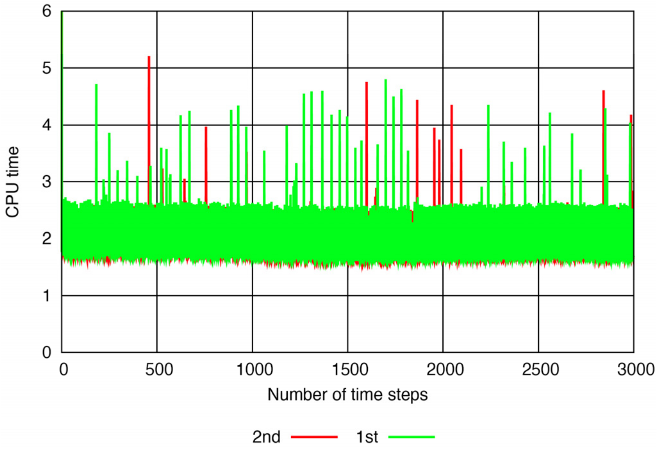

Fig. (14). CPU time.

ACKNOWLEDGEMENTS

This work has been carried out in the framework of FP7 European project W2Plastics (Grant agreement 212782). The research of Dr. Houzeaux has been partly done under a *Ramon y Cajal* contract with the Spanish *Ministerio de Educación y Ciencia*.

REFERENCES

[1] R. Aubry, F. Mut, R. Löhner and J. R. Cebral, “Deflated preconditioned conjugate gradient solvers for the Pressure-Poisson equation”, *J. Comput. Phys.*, vol. 227(24), pp. 10196-10208, December 2008.

[2] E. J. Bakker, P. C. Rema and N. Fraunholcz, “Upgrading mixed polyolefin waste with magnetic density separation”, *Waste Manage.*, vol. 29, pp. 1712-1717, May 2009.

[3] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge University Press, 1970.

[4] R. Codina, "Stabilization of incompressibility and convection through orthogonal sub-scales in finite element methods", *Comp. Meth. Appl. Mech. Eng.*, vol. 190, pp. 1579-1599, December 2000.

[5] R. Codina, “Stablized finite element approximation of transient incompressible flows using orthogonal subscales”, *Comp. Meth. Appl. Mech. Eng.*, vol. 191, pp. 4295-4321, August 2002.

[6] R. Codina and J. Baiges, “Approximate imposition of boundary conditions in immersed boundary methos”, *Int. J. Num. Meth. Eng.*, vol. 80, pp. 1379-1405, December 2009.

[7] R. Codina and J. Blasco, “Analysis of a stabilized finite element approximation of the transient convection-diffusion-reaction equation using orthogonal subscales”, *Comput. Vis. Sci.*, vol. 4, pp. 167-174, August 2002.

[8] R. Codina and G. Houzeaux, *Verification and Validation Methods for Challenging Multiphysics Problems* of Theory and Engineering Applications of Computational Methods, chapter Implementation Aspects of Coupled Problems in CFD involving time dependent domains, pages 99-123. CIMNE, 2006.

[9] R. Codina, G. Houzeaux, H. Coppola-Owen and J. Baiges, “The Fixed-Mesh ALE approach for the numerical approximation of flows in moving domains”, *J. Comput. Phys.*, vol. 228, pp. 1591-1611, March 2009.

[10] R. Glowinski, T.-W. Pan and J. Périaux, “A fictitious domain method for Dirichlet problems and Applications”, *Comp. Meth. Appl. Mech. Eng.*, vol. 111, pp. 203-303, May 1994.

[11] R. Glowinski, T.-W. Pan and J. Périaux, “Fictitious domain methods for the Dirichlet problem and its generlization to some flow problems,” In K. Morgan, E. Oñate, J. Périaux, J. Péraire and O. C. Zienkiewicz, editors, *Finite Element in Fluids, New Trends and Applications*, Eds. Barcelona: Pineridge Press, 1993, pp. 347-368 .

[12] O. Guasch and R. Codina, “A heuristic argument for the sole use of numerical stabilization with no physical LES modeling in the simulation of incompressible turbulent flows”, *J. Comput. Phys.*, vol. 199, pp. 791-801, February 2010.

[13] G. Houzeaux, R. Aubry and M. Vázquez, “Extension of fractional step techniques for incompressible flows: The preconditioned Orthomin(1) for the pressure Schur complement”, *J. Comput. Phys.*, vol. 228, pp. 6316-6332, September 2009.

[14] G. Houzeaux and J. Principe, “A variational subgrid scale model for transient incompressible flows”, *Int. J. Comp. Fluid Dyn.*, vol. 22, pp. 135-152, March 2008.

[15] G. Houzeaux, M. Vázquez, R. Aubry and J. M. Cela, “A parallel fractional incompressible navier-stokes solver for large scale supercomputers”, *J. Comput. Phys.*, vol. 228, pp. 6316-6332, November 2009.

[16] N. Jarrin, J-C. Uribe, R. Prosser and D. Laurence, “Synthetic inflow boundary conditions for wall bounded flows”, *Second Symposium on Hybrid RANS-LES Methods*, Corfu (Greece), 2007.

[17] M-C. Lai and C.S. Peskin, “An immersed boundary method with formal second-order accuracy and reduced numerical viscosity”, *J. Comput. Phys.*, vol. 160, pp. 705-719, May 2000.

[18] D. Martin and R. Löhner, “An implicit linelet-based solver for incompressible flows”, *AIAA Paper*, (AIAA-92-0668), vol. 19, pp. 6-9, January 1992.

[19] F. Mut, R. Aubry, J. Cebral, R. Löhner and G. Houzeaux, “Deflated preconditioned conjugate gradient solvers: extension and improvements”, *AIAA Conference*, Orlando (USA), 2010.

[20] R. A. Nicolaides, “Deflation of conjugate gradients with applications to boundary value problems”, *SIAM J. Numer. Anal.*, vol. 24, pp. 355-365, April 1987.

[21] C. S. Peskin, “The immersed boundary method”, *Acta Numerica*, vol. pp. 479-517, January 2002.

Received: May 27, 2010 Revised: July 14, 2010 Accepted: August 24, 2010