Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXXI, págs. 151-164 (artículo completo)
Alberto Cardona, Paul H. Kohan, Ricardo D. Quinteros, Mario A. Storti (Eds.)
Salta, Argentina, 13-16 Noviembre 2012

PARTICLE TRANSPORT IN LAMINAR/TURBULENT FLOWS

Juan M. Gimenez[a], Damián Ramajo[a,b] and Norberto M. Nigro[a,b]

[a]*International Center for Computational Methods in Engineering (CIMEC), INTEC-UNL/CONICET, Guemes 3450, Santa Fe, Argentina, http://www.cimec.org.ar*

[b]*Facultad de Ingeniería y Ciencias Hídricas - Universidad Nacional del Litoral. Ciudad Universitaria. Paraje "El Pozo". Santa Fe. Argentina. http://www.fich.unl.edu.ar*

Keywords: particle, tracking, turbulence, CFD.

Abstract. The implementation of a Lagrangian particle transport model in non-homogeneous turbulent flow is presented. The model proposes one-way interaction between the continuous and the discrete phases where the behavior of the continuous one is previously solved with other CFD software. Particle dynamics include four forces terms: buoyancy, inertial, drag and added mass. Also, a Discrete Random Walk (DRW) formulation is added to model the changes on the trajectory of the particles due to turbulence.

The temporal integration is carried out using a Runge-Kutta-Felhberg (RKF) integrator, and a novel analytical integrator is used to solve particle-wall collisions. To manipulate the time-step on each particle, a three-layer filter is developed: an user layer, a RKF layer and a physical layer determined by the particle relaxation time. Also, it is proposed an efficient implementation that uses shared memory techniques to parallelize the execution.

To validate the developed code, laminar and turbulent academic tests are presented. Finally, experimental test data from a pilot plant are reproduced using the in house code along with a commercial CFD software, showing a good agreement with data and a radical improvement in time computing comparing with the CFD software.

1 INTRODUCTION AND MOTIVATION

Particle transport simulation has found uses in many scientific fields and industrial applications. Atmospheric dispersion of pollutants, sediment transport in rivers, drug delivery in human airways, nuclear fission products transport are examples where an accurate description of particle transport is of great practical importance.

Simulating this phenomenon is of special interest in turbulent flow systems. Turbulence can be simulated using Direct Numerical Simulation (DNS), but this option is computationally cost prohibitive for high Reynolds numbers. Alternatively, Large Eddy Simulation (LES) techniques are much more computationally efficient as they solve the large scale flow structures (most of the kinetic energy) but employ a sub-grid turbulence model for the small scales (Batcherol, 1953).

Current commercial CFD codes has limited capabilities for lagrangian simulation. For example, the massively used ANSYS-CFX package has not implemented a random walk model for LES turbulence modeling. In addition, coalescence models are only available for euler-euler multiphase modeling. Finally, the computing times required become excessive for industrial problems. Then, it is necessary developing an own implementation, with enough versatility and control, to solve engineering problems including those mentioned features.

In section 2 physical and numerical models are explained in deep. Coalescence phenomena are not yet considered, leaving it for future works. Code validation in front to academic tests are showed in section 3. In section 4 experimental data from a pilot plant facility of a oil-water gravity separation tank are reproduced by means of the in house code and a CFD commercial package. Finally, section 5 aboard the most relevant conclusions.

2 PHYSICAL AND NUMERICAL MODELS

2.1 Turbulent Flow Model

The numerical simulation of particle transport in a fluid flow requires modeling the continuous phase (the driven phase) and the discrete phase, the particles (the advected phase).

If the concentration of particles is high, the particle-particle interaction and its effect on the fluid (four-way coupling) must be modeled. For intermediate concentrations, particle interaction may be neglected (two-way coupling). For low concentrations, the fluid flow is not considerably influenced by the particle flow (one-way coupling) (Hryb et al., 2009).

According to Elghobashi (Elgobashi, 1994), the volume fraction of particles (α^p) defines the type of interaction:

- $\alpha^p < 10^{-6}$: one-way coupling
- $10^{-6} < \alpha^p < 10^{-3}$: two-way coupling
- $10^{-3} < \alpha^p$: four-way coupling

In the present work, a dilute concentration is assumed ($\alpha^p < 10^{-6}$), the flow does not depend on the particle dynamics, therefore it can be solved in an uncoupled way. This allows calculating the fluid and the particle flow in separated stages. First, the fluid is calculated using CFD software and then the particle transport is simulated by the code developed in this work. This strategy is called *Eulerian-Lagrangian one-way coupling*, which only requires that the continuous phase states may be calculated in the particle position to couple the models.

The model of the continuous phase assumes viscous incompressible flow and constant fluid properties (density, viscosity). Turbulence is modeled by Large Eddy Simulation (LES).

The governing equations for LES are obtained by filtering the time-dependent Navier-Stokes equations. The filtering process filters out the eddies whose scales are smaller than the grid spacing used in computations.

$$\nabla \cdot \mathbf{v}^f = 0 \tag{1}$$

$$\rho^f \frac{d\mathbf{v}^f}{dt} = \rho^f \frac{\partial \mathbf{v}^f}{\partial t} + \rho^f \mathbf{v}^f \cdot \nabla \mathbf{v}^f = -\nabla p + \nabla \cdot [(\mu^f + \mu^t)(\nabla \mathbf{v}^f + \nabla \mathbf{v}^{fT})] + \rho^f \mathbf{g} \tag{2}$$

$$\frac{\mu^t}{\rho^f} = (Cs\Delta)^2 |(\nabla \mathbf{v}^f + \nabla \mathbf{v}^{fT})| \tag{3}$$

with the Smagorinsky constant $Cs = 0.18$. Equation 3 is the Smagorinsky subgrid-scale model for LES, where the approximation $\nu_{SGS} \propto l\ q_{SGS}$ is used, being l the length scale of the unresolved motion (usually the grid size Δ) and q_{SGS} the velocity of the unresolved motion (ANSYS, 2010).

2.2 Particle transport Model

Particle transport modeling is a type of multiphase model, where particles are tracked through the flow in a Lagrangian way. The tracking is carried out by a set of ordinary differential equations in time for each particle, which consists on equations for position and velocity. These equations are then integrated using the fourth-order Runge-Kutta-Fehlberg (RKF) method, which solves not only an initial value problem (IVP), but also a system of IVP, adapting the time-step to control the solution error(Burden and Faires, 2003).

For particle transport the second order differential equation $\frac{d^2\mathbf{x}^p}{dt^2} = \mathbf{a}^p$ must be solved. It can be converted into a first order differential equations system doing:

$$\frac{d\mathbf{v}^p}{dt} = \mathbf{a}^p \tag{4}$$

$$\frac{d\mathbf{x}^p}{dt} = \mathbf{v}^p \tag{5}$$

$$\mathbf{x}(t=0) = \mathbf{x}_0$$

$$\mathbf{v}(t=0) = \mathbf{v}_0$$

Following the Newton law, the acceleration of the particle can be calculated according to(Hryb et al., 2009):

$$\rho^p \frac{d\mathbf{v}^p}{dt} = \mathbf{F}_i + \mathbf{F}_b + \mathbf{F}_d + \mathbf{F}_m \tag{6}$$

Four forces acting on the particle are taking into account:

- $\mathbf{F}_i = \rho^f \frac{d\mathbf{v}^f}{dt}$ is the inertial force which depends on local fluid properties.
- $\mathbf{F}_b = (\rho^p - \rho^f)\mathbf{g}$ is the buoyancy force due to the action of gravity acceleration.
- $\mathbf{F}_d = \frac{3}{4}\frac{\rho^f}{d^p} C_D (\mathbf{v}^f - \mathbf{v}^p)|\mathbf{v}^f - \mathbf{v}^p|$ is the drag force due to the action of the fluid opposing to the particle trajectory.

- $\mathbf{F}_m = -\frac{\rho^f}{2}\frac{d(\mathbf{v}^p - \mathbf{v}^f)}{dt}$ is the added mass force due to the fact that the fluid near the particle is also being accelerated.

The drag coefficient depends on the particle Reynolds number $Re^p = \rho^f d^p |\mathbf{v}^f - \mathbf{v}^p|/\mu^f$ and can be estimated according to experimental correlations:

$$C_D = \frac{24}{Re^p} \qquad Re^p < 0.1 \tag{7}$$

$$C_D = \frac{24}{Re^p} + \frac{3}{\sqrt{Re^p}} + 0.34 \qquad 0.1 < Re^p < 1000 \tag{8}$$

$$C_D = 0.445 \qquad Re^p > 1000 \tag{9}$$

So, to solve 4 and 5 using RKF, the function $\mathbf{a}^p$ can be expressed like

$$\mathbf{a}^p(t,\mathbf{x},\mathbf{v}) = \frac{1}{\rho^p + \frac{\rho^f}{2}}\{\frac{3}{2}(\nabla\cdot[(\mu^f+\mu^t)(\nabla\mathbf{v}^f + \nabla\mathbf{v}^{fT})] - \nabla p) + (\rho^p - \rho^f)\mathbf{g} + \frac{3}{4}\frac{\rho^f}{d^p}C_D(\mathbf{v}^f - \mathbf{v}^p)|\mathbf{v}^f - \mathbf{v}^p|\} \tag{10}$$

2.3 Discrete Random Walk

In laminar flows, the path of a particle is deterministic (there is a unique path for a particle injected at a given location in the flow). In turbulent tracking, changes in the trajectory of the particles due the turbulence fluctuations are taking account using a discrete random-walk model. As a result of this, two particles with the same initial state can follow separate trajectories.

Once solved RKF and found dt, each particle position is updated according to(Oksendal, 2000)

$$\mathbf{x}^p_{new} = \mathbf{x}^p + \nabla(\frac{\mu^t}{\sigma\rho^f})dt + \mathbf{w}\sqrt{2\,dt\frac{\mu^t}{\sigma\rho^f}} \tag{11}$$

where $\mathbf{w}$ is a random gaussian variable with values between $[-1, 1]$ and mean zero and $\sigma = 0.7$ is selected from Tominaga(Tominaga and Stathopoulos, 2007).

2.4 Geometry Discretization and Implementation Details

In this work, equations 1, 2 and 3 are solved using the software ANSYS-CFX® v13.0 which uses an element-based Finite Volume Method. This solver requires a FVM mesh (typically an hybrid mesh) to calculate the state of the continuous phase on cell-centers. After that, the Lagrangian stage must be done, where the particles trajectory are calculated. If it is considered steady flow this process must be done only once. On the other hand, for unsteady flows, N states of the continuous phase must be calculated and then, for particle transport two options are available: a) transport the particles for each state and then average its trajectories, b) average the flow states and calculate the particles trajectory on the averaged flow.

The implementation developed by the authors is written in C++ using the Object Oriented Paradigm (OOP) and paralleling the execution over shared memory with the library OpenMP. This code is an add-on of the library PFEM2 previously presented by the authors (Gimenez and Nigro, 2011).

The code manages only Finite Element (FEM) triangular (2d) or tetrahedral (3d) meshes, then mesh conversions must be done, keeping one-to-one the relationship between the points of the FVM mesh and the nodes of the FEM mesh to preserve the quality of the continuous phase representation. Cell-centered states also must be projected to nodal values using averaging algorithms.

To update its position and velocity, each particle must know the element in the mesh where it is placed. Identifying the element, the nodal values ($\mathbf{v}^f$, p and their derivatives) are interpolated in the particle position $\mathbf{x}^p$. With this information equations 4 and 5 can be solved.

2.5 Particle-Wall Collisions

In confined flows, particle-wall collisions become important. This section explains how the algorithm manages that event.

Normally, to find the new particle position and velocity, the RKF algorithm calculates evolution of the variables evaluating in different positions and using different velocities. After that, it calculates an average of those evolutions and updates the unknown variable.

To detect when the particle is near from any wall, elements which contain one or more nodes over the boundary are identified. When the algorithm RKF tries to calculate particle states evolution on one of these elements, it is swapped to an analytical integrator for that particle in the current time-step. This analytical integration has the feature of managing the collision with the walls, something that RKF can not do because there is a discontinuity in the velocity.

The analytical integrator tries to advance the entire time-step following a straight line (particle acceleration is considered null inside boundary elements). If it detects that the new particle position is in another element or outside the geometry (any area coordinate is negative), it searches the time of the crossing or the collision, using line-segment intersection in 2d (Equation 12) or line-plane intersection in 3d (Equation System 13).

$$(1-\alpha)\mathbf{x}_0 + \alpha\mathbf{x}_1 = (1-\beta)\mathbf{P}_0 + \beta\mathbf{P}_1 \tag{12}$$

$$(1-\alpha)\mathbf{x}_0 + \alpha\mathbf{x}_1 = (1-\beta-\gamma)\mathbf{P}_0 + \beta\mathbf{P}_1 + \gamma\mathbf{P}_2 \tag{13}$$

Once it is solved the equations system 12 or 13, the exit time is calculated following 14.

$$t_{exit} = \alpha\ dt \tag{14}$$

If the particle collisions with any wall, it is calculated the normal to the surface $\mathbf{n}$ in the exit point, and the velocity of the particle is updated following

$$v_{t1}^p = c_t\ v_{t1}^p \tag{15}$$

$$v_{t2}^p = c_t\ v_{t2}^p \tag{16}$$

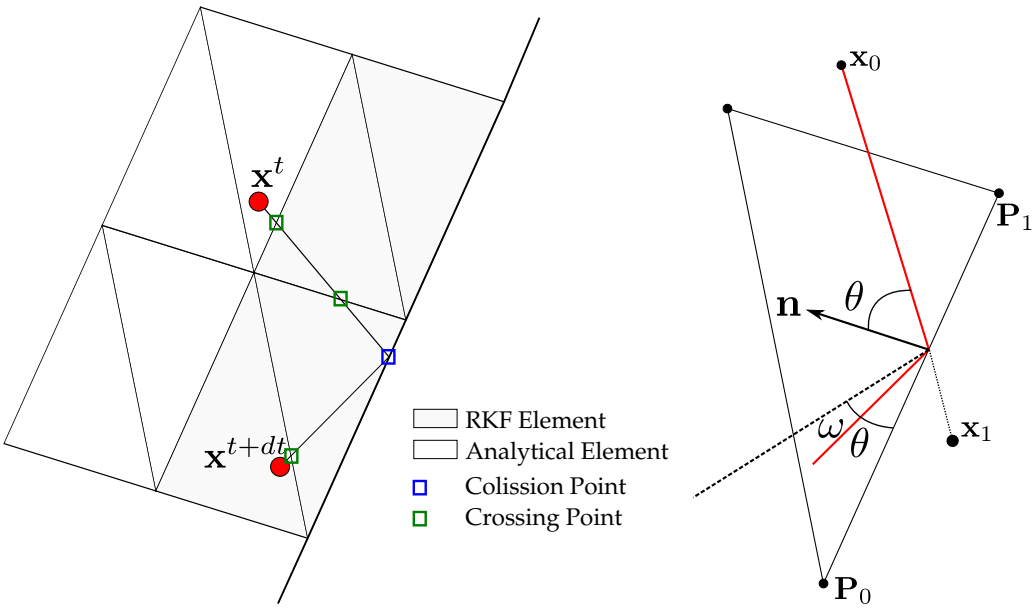
$$v_n^p = c_n\ v_n^p \tag{17}$$

where $v_n^p = \mathbf{v}^p \cdot \mathbf{n}$, $v_{t1}^p = \mathbf{v}^p \cdot \mathbf{t_1}$, $v_{t2}^p = \mathbf{v}^p \cdot \mathbf{t_2}$, and c_t and c_n are the tangential reflection coefficient and normal reflection coefficient respectively. Those coefficients allow to simulate conditions like wall rugosity (if they are chosen randomly), or free-surface (choosing $c_n = 0$).

On the other hand, if the calculated exit point is a crossing one, the particle enters into a new element and keeps its velocity. Also, in outlet flow boundaries, if a collision is detected and the

surface element was identified as *out_ele* (it belongs to outlet), the algorithm detects that the particle has left the domain, removes it, and stores that event.

Finally, the algorithm continues until completing the time-step. Figure 1a shows a particle trajectory in one time-step, whereas the Figure 1b presents how the algorithm solves the collision of the particle with the wall.



(a) (b)

Figure 1: 1a: Trajectory of the particle with the analytical algorithm. 1b: Treatment of the collision with the wall, the incidence angle θ is changed to $\theta + \omega$ after collision.

2.6 Selecting time-step

A correct time-step selection becomes very important when high efficiency on computing times is wanted.

The first layer of selection of the time-step (Δt) is decided by the user. Typically, the user chooses some Δt_{user} to save particle data, but this time is usually too long and the algorithm may obtain wrong results. RKF allows to control the time-step truncating the local error (usually $\Delta t_{RKF} < \Delta t_{user}$). Although RKF has control of the time-step warrantying some level of error, sometimes this Δt_{RKF} is also excessive to represent the dynamics of the problem. Because of that, the *particle response time* or *particle relaxation time* (τ^p) is used as initial time-step to carry out the integration. It characterizes the time required for a particle to adjust or *relax* its velocity to a new forces condition. It is an indication of the ability of the particle to quickly adjust to a new environment or condition. It depends on the mass and mechanical mobility of the particle, and is not affected by the external forces acting on the particle.

This value is calculated like:

$$\tau^p = \frac{\rho^p d^{p2}}{18\mu^f}, \qquad Re^p < 0.5 \tag{18}$$

$$\tau^p = \frac{4}{3}\frac{\rho^p d^p}{C_D|\mathbf{v}^f - \mathbf{v}^p|}, \qquad Re^p > 0.5 \tag{19}$$

Summarizing, the Δt selection algorithm is the next:

Algorithm 1 - Time Selection Algorithm

$T = 0$
while $T < T_{final}$
  $t = 0$
  $dt = \tau^p$
  while $t < \Delta t_{user}$
    do RKF
    if $(e_{rkf} < tol)$
      $t = t + dt$
    $\delta = 0.84(\frac{tol}{Rerror})^{0.25}$
    if($\delta <= 0.1$)
      $dt = 0.1 * dt$;
    else if($delta >= 4$)
      $dt = 4 * dt$;
    else
      $dt = \delta * dt$;
    if($dt + t > \Delta t_{user}$)
      $dt = \Delta t_{user} - t$;
  end while
  $T = T + \Delta t_{user}$
end while

3 VALIDATIONS

3.1 Wooden Sphere in Water

To validate the implementation of the model, the trajectory of a wooden sphere in water is analyzed following the example presented in (Hryb et al., 2009) where the analytical solution is presented. Test conditions are:

- $\mathbf{v}_o^p = -8[m/s]$ (along **g** direction)
- $d^p = 0.05[m]$
- $\rho^p = 700[kg/m^3]$
- $\rho^f = 1000[kg/m^3]$

Comparison between numerical results obtained with the in house code with analytical results is reported in the Figure 2.

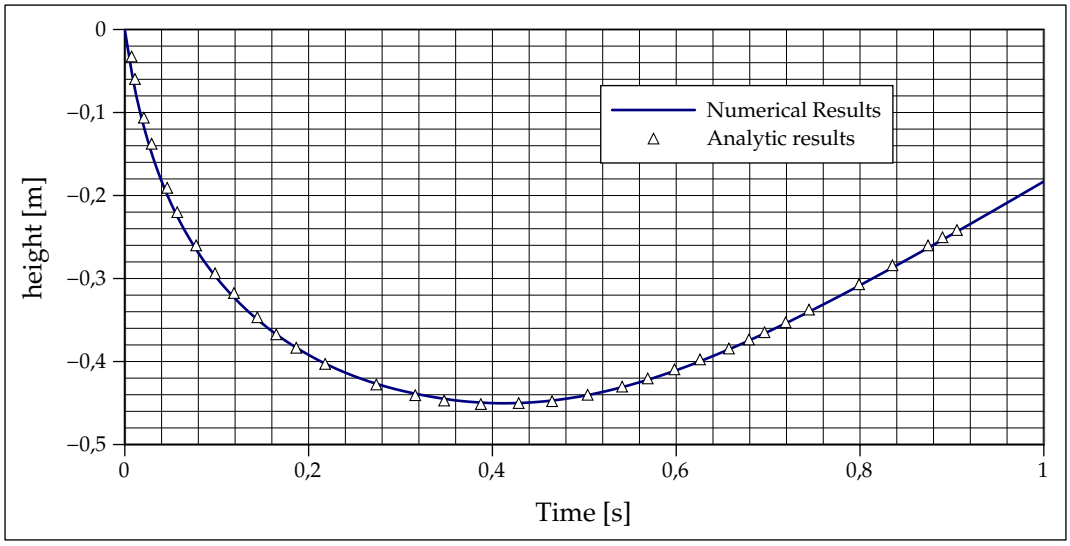


Figure 2: Trajectory of a wood sphere in water.

3.2 Particles into a Cylindrical Couette Flow

Another test to validate the implementation is to analyze the trajectory of particles in a laminar flow into a rotating cylinder. Figure 3 presents the case.

$$p(r,z) = |\mathbf{g}|(h-z) + \tfrac{1}{2}\rho r^2\omega^2$$

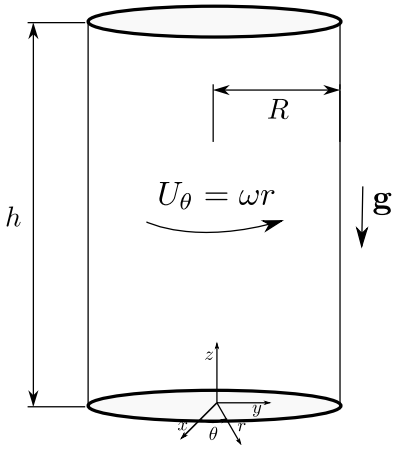


Figure 3: Description of the test. $h = 5[m]$, $R = 0.5[m]$

Acceleration along z axis is constant due to the forces acting in this direction are constant ($F_z = F_b - F_w$ where F_b is the buoyancy force and F_w is the weight). Terminal velocity is assumed to be reached instantaneously with the value $V_t = \sqrt{\frac{4|\mathbf{g}|d^p}{3Cd}(\frac{\rho^p-\rho^f}{\rho^f})}$.

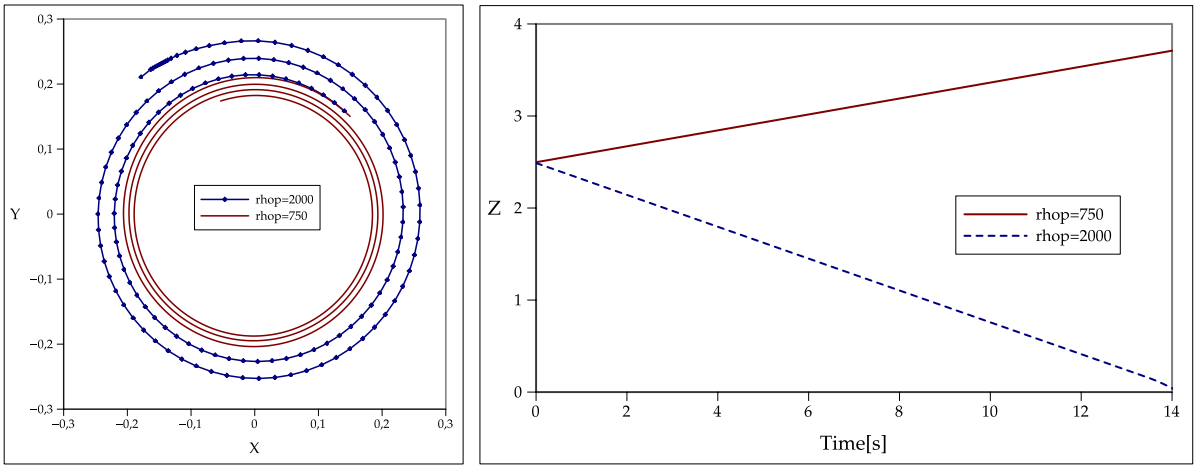
Along radial direction the forces acting on the particle depend on r ($F_r = F_p + F_{cp} + F_D$ where F_p is the pressure gradient force, F_{cp} is the centripetal force and F_D is the drag force). It can be demonstrated that instantaneous velocity along radial direction has the expression: $V_t(r) = \sqrt{\frac{4\omega^2 r d^p}{3Cd}(\frac{\rho^p-\rho^f}{\rho^f})}$.

Two tests are presented. Fields p and U_θ presented in the Figure 3 are used, and physical parameters $d^p = 0.001[m]$, $\mu^f = 10^{-5}[Pa\ s]$, $\rho^f = 1000[\frac{kg}{m^3}]$, $\mathbf{g} = -10[\frac{m}{s^2}]\mathbf{z}$ giving a Reynolds number greater than 1000 (it can be used $Cd = 0.445$). The Table 1 summarizes the results obtained where a relative good agreement is achieved.

$\rho^p[\frac{kg}{m^3}]$	Analytical V_t (z-axis)	Numerical V_t(z-axis)	Relative Error
2000	$-0.17309[\frac{m}{s}]$	$-0.1731[\frac{m}{s}]$	0.005%
750	$0.087[\frac{m}{s}]$	$0.0865[\frac{m}{s}]$	0.025%

Table 1: Comparison between analytical and numerical terminal velocities.

Figure 4 shows the trajectories of the particles. If the density of the particle is smaller than the density of the fluid the particle floats and it is centrifuged, otherwise it sediments and its trajectory radius diminishes.



(a) (b)

Figure 4: Trajectory of the particles. (a): XY plane. (b): height.

4 REPRODUCING EXPERIMENTAL DATA IN A PILOT PLANT

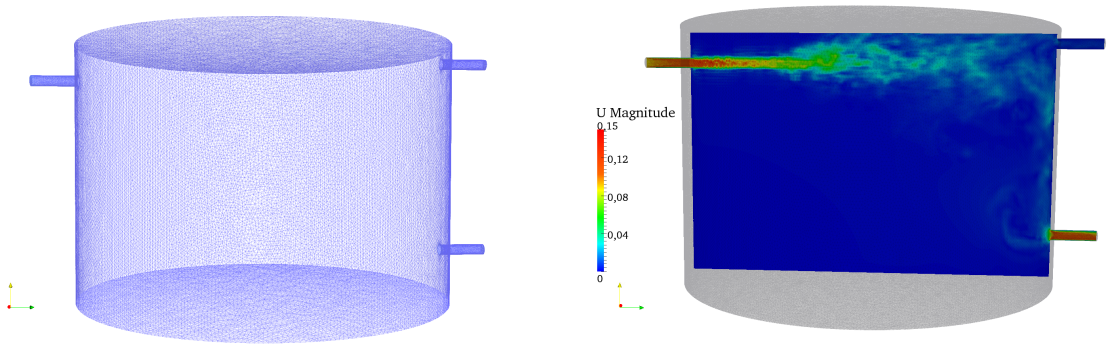
In secondary oil extraction big quantities of water are employed, and it is called *process water*. This process water is injected into oil wells to help to extract crude by drag and buoyancy forces. The process water must not be thrown again in the environment, and this is why it must be processed to separate the crude that it contains. This process is carried out with buoyancy equipment called *skimmer*, which basically consists of a big tank with internals where the water-oil mixture enters and the crude is separated by buoyancy and collected on the free-surface while the water is removed from the bottom.

A key parameter to determinate the efficiency of this equipment is the called *characteristic residence time*, which is calculated with $t_r^{eff} = V_{eff}/Q$, where V_{eff} is the tank effective volume and Q is the inlet flow rate (if t_r is big the efficiency grows because the crude has more time to float by buoyancy). However, for experimental cases, this t_r^{eff} is far from the theoretical t_r because there are *dead zones* (where the fluid has very small velocities) and *recirculation zones*. This reduces V_{eff} and diminishes t_r^{eff}.

In the subsection 4.1 numerical simulations are compared with experimental results obtained from a pilot skimmer built to evaluate some gravity separation technologies.

4.1 Skimmer computational model

Figure 5a presents the geometry which models the studied skimmer. The dimensions of the skimmer are $0.8[m]$ of radius and $1.06[m]$ of height, then $V = 2.13[m^3]$. The flow states ($\mathbf{v}$, p and μ^t) were obtained using a mesh with more than two millions of hexahedrons, impossing $Q = 500[l/h] = 0.5[m^3/h]$. Figure 5b shows the magnitude of the velocity field in a slice. Note that the outlet flow is distributed 95% through the bottom pipe and 5% through the upper pipe.



(a) (b)

Figure 5: Skimmer computational mesh and a slice with the magnitude of the velocity calculated by CFD software. The skimmer has one inlet pipe at left, and two outlet pipes at right.

4.2 Residence Times of a Saline Pulse

The upper pipe is closed in this set of experiments, but this is not a problem because its influence on the data obtained is small. In the Figure 6 a set of experimental data measuring the conductivity of the outflow is presented. Two big peaks appear around $t = 170[s]$ and $t = 350[s]$. Another peak is found around $t = 550[s]$.

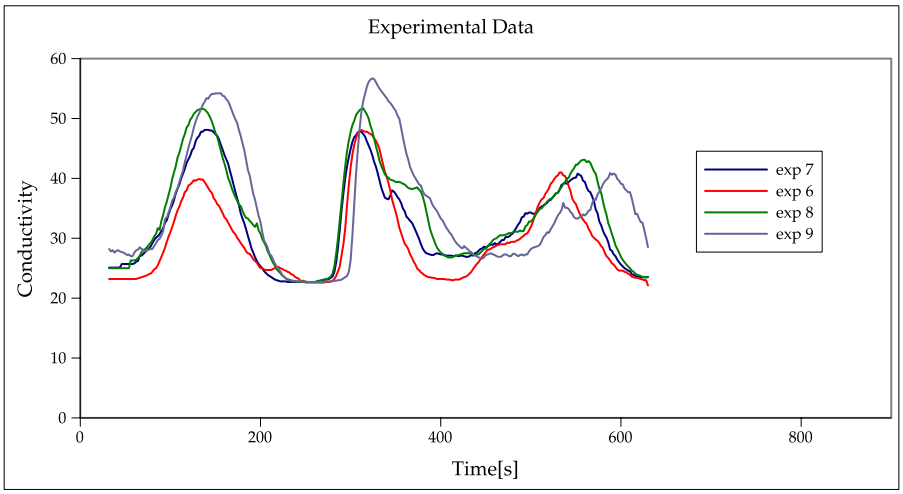


Figure 6: Conductivity measured in the bottom pipe.

The computational simulation of the residence times of a saline pulse are carried out by particle transport. A set of around 15000 particles are seeded on random positions in the inlet pipe and is supposed steady flow for the continuous phase. Physical parameters are:

- $\Delta t_{user} = 20[s]$
- $\mu^f = 8.89\ 10^{-4}[Pa\ s]$
- $\rho^f = 1000[Kg/m^3]$
- $\rho^p = 1000[Kg/m^3]$
- $d^p = 5\ 10^{-6}[m]$

The results obtained with the in house code (using a mesh with seven millions of tetrahedrons) are compared with those obtained with the CFD software (using a mesh with two and a half millions of hexahedrons) and they are presented in the Figure 7. Note that both meshes have the same number of nodes or points.

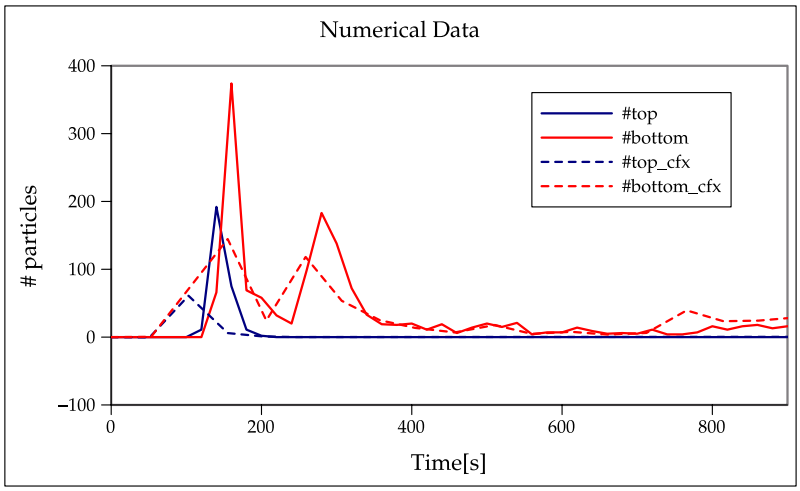
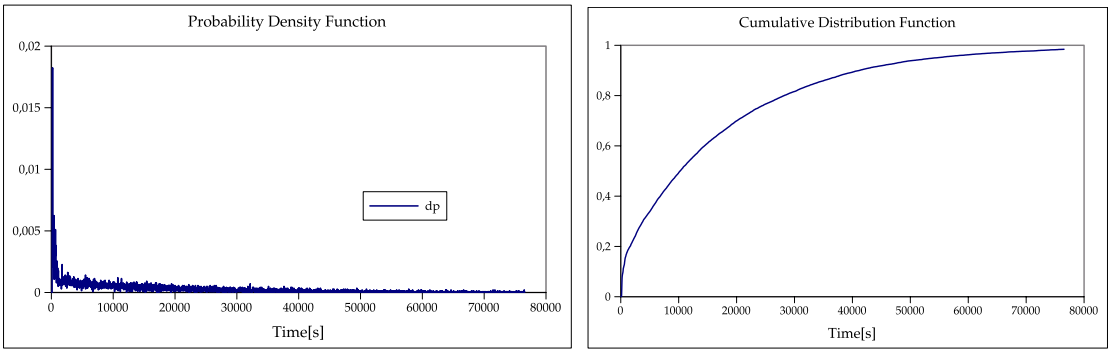


Figure 7: Number of particles detected on outlet pipes at each time. #top and #bottom are the number of particles detected with the in house code and #top-cfx and #bottom-cfx with ANSYS-CFX software.

Both the CFD software and in-house code reproduce the first two peaks observed on experimental results. However, the third peak is not clearly reproduced on both calculations.

A extended simulation was done to estimate the effective residence time and compare it with the theoretical residence time $t_r = V/Q = 15345[s] \approx 4h15'$. In the simulation, 14316 particles were seeded and it was executed until the 99% of the particles had left the skimmer (approximately $5V/Q$).

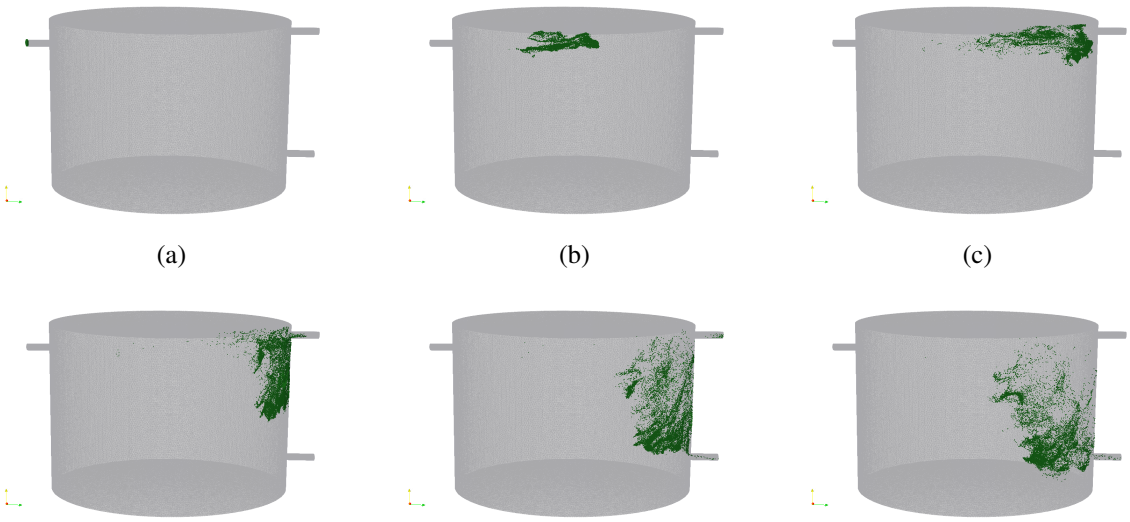
The probability density function $dp(t)$ is calculated as $dp(t) = \frac{p_{acum}(t)}{\int_0^\infty p_{acum}(t)dt}$, then the expected value of dp can be calculated as $t_{mean} = \int_0^\infty t\ dp(t)dt$. Using these expressions, t_r^{eff} was calculated, obtaining $t_r^{eff} = 15246[s]$. This result has an error of about 1%. Figures 8a and 8b show the calculated statistical functions.



(a) (b)

Figure 8: Statistics functions for the residence times.

Finally, Figure 9 shows some snapshots with the particles positions in the skimmer at different times.



(a) (b) (c)

(d) (e) (f)

Figure 9: Snapshots with particles position at different times. Times (from left to right) are: $0[s]$, $20[s]$, $60[s]$, $100[s]$, $140[s]$ and $180[s]$.

4.3 Computing Times

The in house code allows to parallelize the execution using shared memory with the library OpenMP.

Tests of the in house code were done with a *Intel i7-2600k* processor (on 4 cores) and the CFD Software test was done with a *Intel i7-3930k* processor (on 5 cores). Performance tests[1] for Intel i7-3930k give a factor 13.55 while for Intel i7-2600k they give 8.89, this means that 3930k is $1.52x$ faster than 2600k (normalized time takes into account this factor).

Both cases simulated $380[s]$ of real time, tracking around 15000 oil particles ($\rho^p = 900[\frac{Kg}{m^3}]$) in the skimmer presented above.

[1] http:\www.cpubenchmark.net

- In house code:
  CPU Time: $1322[s]$ - Normalized Time: $\mathbf{1322}[\mathbf{s}]$
- ANSYS-CFX:
  CPU Time: $25200[s]$ - Normalized Time: $\mathbf{38304}[\mathbf{s}]$

This result shows that the in-house code is $30\times$ faster.

5 CONCLUSIONS

Particle transport has been developed using the Eulerian-Lagrangian one-way coupling approach and considering a DRW for LES.

In the current implementation, the time integration is carried out by a Runge-Kutta integration with time adaptability (Runge-Kutta-Felhberg). Also, a novel algorithm to manage particle-wall collisions is proposed, and a time-step selector is presented to achieve high performance on computing times.

The implementation, coded in OOP-C++, was validated with an analytical solution and with experimental results obtained from a pilot plant. Comparing with commercial software, the computing times achieve a significant improvement and numerical results are similar to those obtained with it.

Having a in-house code allows to modify it and include new physical models, such as the coalescence phenomena, particle-particle collisions or random walk on velocity.

6 ACKNOWLEDGMENTS

The authors want to thank CONICET, Universidad Nacional del Litoral (CAI+D Tipo II 65-333 (2009)) and ANPCyT-FONCyT (grants PICT 1645 BID (2008)) for their financial support.

NOMENCLATURE

$\mathbf{a}^p$	Particle acceleration
α^p	Particle volume fraction
$\mathbf{g}$	Gravity aceleration
μ^f	Fluid viscosity
μ^t	Turbulent viscosity
ρ^f	Fluid density
ρ^p	Particle density
σ	Turbulent Schmidt number
τ^p	Particle Relaxation Time
$\mathbf{v}^f$	Time average fluid velocity
$\mathbf{v}^p$	Particle velocity
$\mathbf{w}$	Normal Gaussian normal variable
C_D	Drag Coefficient
c_n	Normal Reflection Coefficient
c_t	Tangential Reflection Coefficient
d^p	Particle diameter
p	Time average pressure
Re^p	Particle Reynolds number

REFERENCES

ANSYS. *ANSYS CFX-Solver Theory Guide*. ANSYS, Inc., 13th edition, 2010.

Batcherol G.K. *The theory of homogeneous turbulence*. Cambridge University Press., 1953.

Burden R.L. and Faires J.D. *Numerical Analysis*. Math Learning, 7th edition, 2003. ISBN 970-686-134-3.

Elgobashi S. On predicting particle-laden turbulent flows. *Applied Scientific Research*, 52:309–329, 1994.

Gimenez J.M. and Nigro N. Parallel implementation of the particle finite element method. *ENIEF*, XXX:3021–3032, 2011.

Hryb D., Cardozo M., Ferro S., and Goldschmidt M. Particle transport in turbulent flow using both lagrangian and eulerian formulations. *International Cummunications in Heat and Mass Transfer*, 36:451–457, 2009.

Oksendal B. *Stochastic Diferential Equations: An introduction with applications*. Springer-Verlag., 5th edition, 2000.

Tominaga Y. and Stathopoulos T. Turbulent schmidt numbers for cfd analysis with various types of flowfield. *Atmospheric Environment*, 41:8091–8099, 2007.