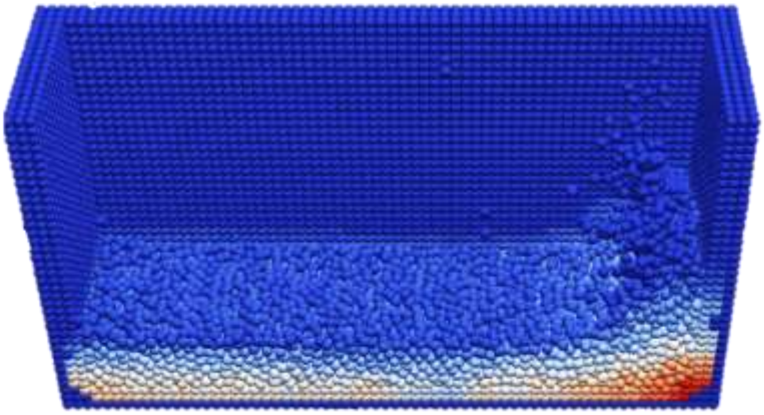


EXPLICIT-MPS METHOD

Development of accuracy and computational time

Amiot Baptiste
Ecole Polytechnique de l'Université de Nantes & Okayama University

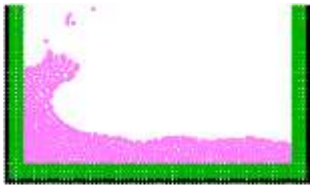
Table of contents

Overview

How is it doable to anticipate facility damages due to heavy rains or tsunamis? Japan is perhaps the most famous country where this question needs to be investigated to save inhabitant lives. To illustrate this statement, one relevant example is the Tohoku tsunami which destroyed the Fukushima power plant. Power realized was around 2 megatons of TNT, and seven years later we can still observe its effects on the japan east coast. Lately, Okayama prefecture was affected by heavy rains on July 2018 which destroyed many facilities as buildings, or bridges. In that case, overflow is less powerful but mixed with gravity action and soil erosion, water begins to be hazardous.

Picture 1 : Japan railroad destroyed due to river overflow in Hiroshima ©japantimes newspapers

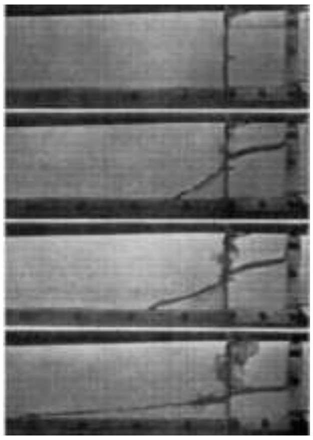
To anticipate damages, the most convenient way is to make experiments. But conditions of disasters are quite complicated to build, even in reduced scales. Values that we want to know, in terms of mechanics view, are data pressures on layout buildings (or apparatus), subjected at water flows. So that, scientists and engineers are trying now to create computing simulations which are perfectly fitting with that kind of horrific events in a way to conceive safer equipment.



Picture 3 : Dam-break model

The major difficulty for this kind of simulations concerning boundaries. In the case of floods or waves, we need to deal with the interface between liquid and gas. CFD is commonly used in fluid mechanics analysis, but free-boundaries are not well represented. Moreover, the simple fact of using a Eulerian approach involves digital noises and then efficiency is reduced. Here we choose to work on the E-MPS method, because it's using the Lagrangian approach, means that fluid is represented as a lot of particles. So that, simulation is not curbed by digital noises and does not include any meshes. Furthermore, computational time is reduced compares to grid-mesh eulerian solutions.

Thus, the aim of this internship at Okayama University was to develop a previously created E-MPS code. It has been testing on two models called: hydrostatic and dam-break problem as shown in the picture below. These models are currently used, and experimental data are available to check simulation reliability. In the future, we would like to simulate phenomenon in 3 dimensions so that we need to work on two sides. First, we need to look for the efficiency of pressure representation, and then, we want to obtain fast simulations by reducing computation time.



Picture 2 : Dam break experiment by Martin and Moyce (1952)

I. E-MPS for tsunami inquiries

What is the E-MPS method?

Features and applications

E-MPS acronym means Explicit-Moving Particle Semi-implicit, which is a direct link with the previous method called MPS: Moving Particle Semi-implicit. These two methods do not introduce any meshes, any nodes or that we call weak equations. Also, they use Lagrangian description by discretization of strong form partials equations as Navier-Stokes equations. So we directly solve them by using particle interactions and introducing “Kernel function” as a weight function that allows particles to only consider their neighbours in the calculation of interactions. It’s quite different from another solver as FEM, and one of the best argument to prefer Lagrangian MPS concerns free boundaries.

Finite Difference or Finite Elements are efficient to solve partial equation problems when model boundaries are fixed because mesh must not be recalculated at each calculation steps. When boundaries are moving, recalculating mesh is mandatory to bear efficiency, so computation time is increasing and restricted our simulations. Applications of free boundaries are prevalent for two-phasic experiments, such as phase shifts or atmosphere studies. To show how waves are running, this kind of simulations are enough because they allow studies about fluid velocity and pressure field. Tsunamis are bigger then shore waves, but they share the same movement equations so we know that E-MPS or MPS (SPH too) will work well.

What are the differences between MPS and E-MPS? This answer is giving by workflows comparisons. Even they are sharing the same flow; they solve density, location and velocity, then pressure for each particle; the last parameter is not calculated in the same manner. We also know that fluid movement is linked with pressure field, so any modifications in that processing will modify the experiment. Instead of getting the pressure field by using the Poisson equation as MPS approach, the explicit way only uses the prior calculation of density to obtain pressure field. Easier and less computation time is required but E-MPS must introduce the concept of weakly compressing fluid. Density needs to be consistent from start to end.

In most cases, this assumption is true for waves studies and using E-MPS or MPS show accurate results. The key parameter is, at this stage, the computational time required. And due to explicit workflow, E-MPS wins. It is more comfortable to run an E-MPS study then MPS because the accuracy in terms of pressure will be at least reach same results, also depending on the experiment, E-MPS will be 3 or 5 times faster. But, E-MPS needs reflection to ensure compressing inquiries.

To conclude this part, we just want to precise what kind of experiments are available with E-MPS approach. Because it is developed to simulate water flow, major experiment concerns oceanographic development as fluid-structure interaction. But also, it is commonly used when water flow is mixed with sediment, and some applications as river flow digging or landside prediction are made.

Brief state of art (bibliographic researches)

Basically, papers about E-MPS, MPS or SPH have been writing in Japanese. As a beginner in this fabulous language, it was a burdensome work to understand these papers. Nevertheless, few ones are written in English and so I mainly used them.

As we previously said, simulations were studied in coastal engineering and two major methods are still existing: SPH and MPS. SPH is used for compressible fluid problems and MPS for incompressible ones. But E-MPS introduce few aspects of compressible fluid, so research must be completed by inspecting the SPH model and use advances in E-MPS.

In October 2017, GOTOH and OKAYASU have published a tremendous paper named:" Computational wave dynamics for innovative design of coastal structures" which is the best paper referring to SPH functions applicated on MPS models at this moment. In this one, a lot of information is shown and most of them were tried in our codes. Main problems of MPS were due to pressure fluctuation, themselves caused by the Lagrangian nature of moving calculation. They designed some answers based on a higher order scheme of interpolation, but also by reducing tensile instabilities. Two of their developments provide interesting results -and are now used in our code: CMPS and HL. CMPS is a slightly different way of programming pressure gradient and allows the conservation of momentum. HL is another technic to programming Laplacian in a higher scheme and so it reduces instabilities. Then the paper is written for MPS scheme and a lot of enhancements are made to the Poisson pressure equation which is not used in E-MPS. But they are proposing some keys to solve instabilities caused by boundaries like walls, which were also tried to improve the final accuracy of our codes.

Our work on boundaries lead us to another paper wrote by MITSUME, YOSHIMURA, MUROTANI and YAMADA, from the University of Tokyo. "Explicitly represented polygon wall boundary model for the explicit MPS method" (2015), is a research about programming boundaries and especially how to deal with wall conditions. The major challenge concerning walls is to avoid using unnatural repulsive forces to coerce particles in the tank. They are solving this aim by developing virtual particles called "mirror particles" and are minimizing unnatural repulsive forces by only using it when it's mandatory, most of the time when particles reach a huge velocity. Mirror particles technic (also called ERP for Explicit Polygon Representation), was tried but due to calculation time increases, we don't use it in our new codes. But it was inspirational, and I developed some different solutions based on mirror particle programming.

To validate our codes, it is necessary to compare our results to experimental ones. First studies were done by MARTIN and MOYCE in 1952 and they proposed dimensionless results concerning dam-break model. Even if it appears to be a long time from now, these results are quite accurate. With new sensors and equipment, KOSHIZUKA, TAMAKO and OKA proposed new data which are slightly different from previous ones. Also, last research on this experimental subject was made by MATSUDA from Nagoya Institute of Technology, with a new model and bigger apparatus. His thesis focused on geotechnics and complex situation where the ground is porous and includes soil etc. It was too difficult to use this result because our goal was to obtain major improvements of time-consuming and accuracy, but in the future, it could be useful to build our codes on models provided by this thesis. In our case, I decided to compare dam-break results with experimental data of MARTIN and MOYCE work because they are not too different from KOSHIZUKA and al. ones, and our previous codes checked with their results.

Last paper I need to quote is the one proposed by KOSHIZUKA, SHIBATA and MUROTANI. “Introduction to particle methods” (2014) is written in full Japanese, except the formulas! Some tricks are proposed to reduce time-consuming, mainly by decreasing step of calculation, modifying the workflow to avoid unnecessary loop etc. They are suggesting reducing the accuracy as long as it stays in suitable error area in order to reach good results for less time. This includes streamlining mathematical functions as linearized functions.

Equations and discretization

With all these statements said, we are entering now in some mathematical features, by introducing movement equations, interaction formulas and discretized operators. This part will just remind the purpose of E-MPS, all the improvements made during my internship at Okayama University are explained in detail after.

Transport equations

We can write conservative momentum on the fluid area, this is Navier-Stokes equation

$$\frac{Du}{Dt} = -\frac{1}{\rho}\nabla P + \mathrm{\nu}.\Delta \mathrm{u} + \mathrm{f} \tag{1}$$

With $\frac{Du}{Dt}$ the differential of velocity, ∇ the gradient operator, ν is cinematic viscosity, Δ the Laplacian operator and f all the outside strength, here we only consider gravity force.

Then we can say that fluid keep is mass constant so that we write

$$\frac{\partial \rho}{\partial t} = -\nabla u = 0 \tag{2}$$

With ∇ the gradient operator on velocity.

According to meshless features, Navier-Stokes equations are written in particles viewpoint as

$$\frac{Du_i}{Dt} = -\frac{1}{\rho_i}\nabla P_i + \mathrm{\nu}.\Delta \mathrm{u_i} + \mathrm{f_i} \tag{3}$$

It will be computed in that way, solving each Navier-stokes equation for each particle and then switch to another particle. By the way, to consider the action of j particle on the I one, it is mandatory to introduce an interacting particle model as Kernel functions.

Kernel function

We need to define an interaction model because fluid particle velocities and pressure are linked with each other. But we understand that if particles are too far, it will be meaningless interactions between them. Kernel functions are introduced to weight these actions and formula used here is:

$$w(r) = \begin{cases} \frac{r_e}{r} - 1 & (0 \leq r < r_e) \\ 0 & (r_e < r) \end{cases} \tag{4}$$

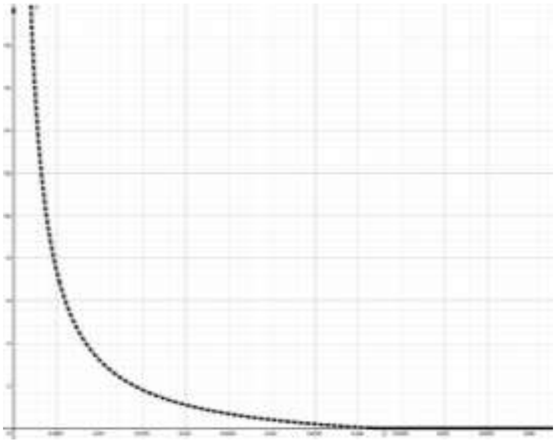


Figure 1: x-axis is distance r[m] and y-axis is w(r) [UA]

From which r_e is the lowest distance to share physical properties, its value is taken as $r_e = 3.5l_0$ with l the space between particles. r is the current distance between particles. It would be easier to show the purpose of these later, but one thing that we must kept in mind is the function shape of w.

Lots of kernel functions are still existing, but in E-MPS we use one which is equal to zero for faraway particles and equal to infinity when they are closer. It is also a simple equation because it appears in every computing things, so the simplest shape is also the fastest processing.

Density

The lagragian approach means that density has to be defined for every particle, so equations are defined as a summation of nearby neighbours.

$$n_i = \sum_{j \neq i} w\left(\left|\overrightarrow{x_{ij}}\right|\right) \tag{5}$$

With $\overrightarrow{x_{ij}} = \overrightarrow{x_j} - \overrightarrow{x_i}$, in all the report we simplify this script by x_{ij}.

Also, to show the incompressibility, we should write

$$\rho_i = m.N_i = \frac{m.n_i}{\int_V w(r)dV} \tag{6}$$

Assuming that each particle has the same mass, n_i steady values ensure incompressibility aims. In the following papers, we are using n_0 as the first computation of density, and we will adjust current density with n_0 to validate incompressibility.

Discretization

Gradient

For the particle i, the gradient vector is defined regarding all the particles in its neighbourhood. A weight is given by the kernel function, so for the particle i surrounded by particles j, the gradient vector of φ is discretized as:

$$\langle \nabla \varphi \rangle_i = \frac{d}{n^0} \sum_{j \neq i} \left[\frac{\varphi_i + \varphi_j}{|x_{ij}|^2} \, x_{ij} \, w(|x_{ij}|) \right] \tag{7}$$

with d the space dimension, n^0 the fluid density.

As shown before, the gradient is only used to represent pressure. So φ is the pressure function in E-MPS method.

Laplacian

For a time-dependent diffusion problem, it can be shown that the variance of the distribution increases by $2dv\Delta t$ per time step Δt, so it leads to the following discretization:

$$\langle \nabla \varphi \rangle_i = \frac{2d}{n^0 \lambda^0} \sum_{j \neq i} [\varphi_{\mathrm{ij}} . w(|x_{ij}|)] \tag{8}$$

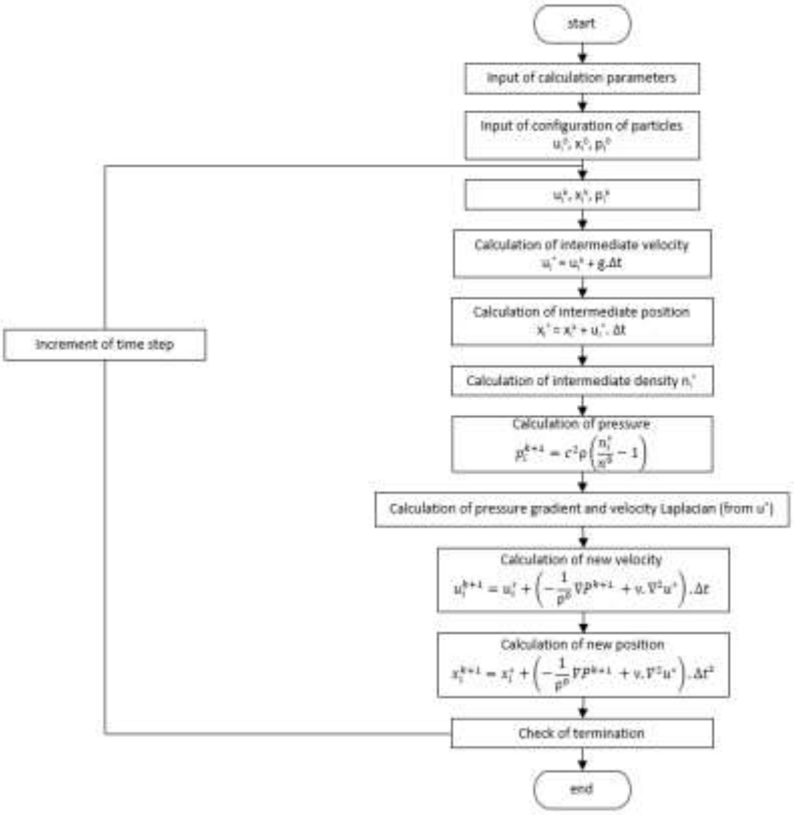
With

$$\lambda^0 = \frac{\int_V w\big(|x_{ij}|\big) |x_{ij}|^2 . dV}{\int_V w\big(|x_{ij}|\big) . dV} \tag{9}$$

Note that V is the area close to the studied particle.

Laplacian is only used with velocity calculation, so φ is the particle velocity in the E-MPS method.

Programming workflow


start
Input of calculation parameters
Input of configuration of particles
u_i^0, x_i^0, p_i^0
u_i^k, x_i^k, p_i^k
Calculation of intermediate velocity
$u_i^* = u_i^k + g.\Delta t$
Calculation of intermediate position
$x_i^* = x_i^k + u_i^* . \Delta t$
Calculation of intermediate density n_i^*
Calculation of pressure
$p_i^{k+1} = c^2 \rho \left(\frac{n_i^*}{n^0} - 1 \right)$
Calculation of pressure gradient and velocity Laplacian (from u*)
Calculation of new velocity
$u_i^{k+1} = u_i^* + \left(-\frac{1}{\rho^0} \nabla P^{k+1} + v. \nabla^2 u^* \right) . \Delta t$
Calculation of new position
$x_i^{k+1} = x_i^* + \left(-\frac{1}{\rho^0} \nabla P^{k+1} + v. \nabla^2 u^* \right) . \Delta t^2$
Check of termination
end
Increment of time step

From a computing point of view, the figure below shows how the functions are running, and we could appreciate all the structures previously shown. We now can introduce some other physical features that are mandatory to compile the simulation.

Intermediary time step

To ensure stability, E-MPS is working by mid-step calculation in which first only gravity is considered, then pressure and viscosity are added. In that way, step k goes to step k* thanks to

$$u^{*} = u^{k} + g\Delta t \tag{10}$$

For velocity, and for locations:

$$x^{*} = x^{k} + u^{*}\Delta t \tag{11}$$

Note that this mid-step cut the incompressibility off, particles are moving in any direction without paying attention to the neighbourhood. Calculating density as n^{*} shows a slight difference to n_0, that is the weakly compressible fluid hypothesis. To keep up the pressure field, we compare the new density as the previous one which was supposed constant.

Pressure calculation

Another point is that pressure calculation involves mid-step calculation density n^{*}. As shown in the figure, the formula is

$$P_i(n^{*}) = \rho c^{2}\left(\frac{n^{*}}{n^{0}} - 1\right) \tag{12}$$

With c^2 a constant parameter which allowed consistency. Nevertheless, in this paper w,e will work on another experimental pressure calculation because the previous one has an unphysical behaviour in some cases.

II. Ways of improvements

Thanks to the bibliography, I was able to develop some tricks to increase accuracy and decrease the computational time needed to solve hydrostatic and dam break systems. This part also shows few advances that I've made, it concerns accuracy by research of better pressure field calculation but also the computation time by switching from wall particle type as only one-layer particle wall.

Accuracy of E-MPS

Pressure Gradient

Due to instabilities, the pressure gradient is always modified. Even if our model works, sometimes pressures are fluctuating, and so gradient discretization is not adequate. I'm proposing here to develop CMPS model to obtain a larger gradient and improve stability. As an example, the figure shows how we can deal with this problem. Firstly, we need to determine for each particle, which particle in the neighbouring area has the smaller pressure. According to the broad movements, smaller values are close to each other. Added in discretized gradient, they erase volatility by reducing the gap when neighbour values are too big.

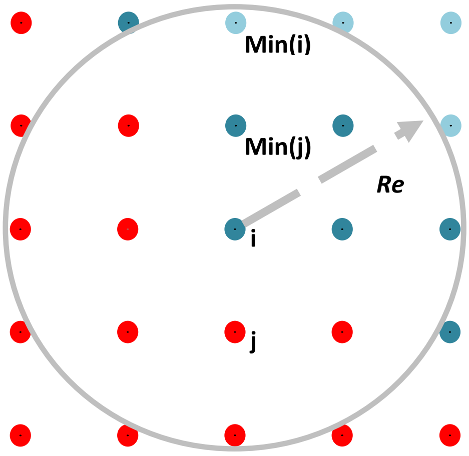


Figure 2: Difference between the two minus particles will erase the pressure gap between I and j when gradient will be calculated

Consequently, the discretized gradient has a new shape:

$$\langle \nabla \mathrm{P}\rangle_i = \frac{d}{n^0}\sum_{j\neq i}\left[\left(\frac{(\mathrm{P_i}+\mathrm{P_j})-(P_i^{min}+P_j^{min})}{\left|x_{ij}\right|^2}\, x_{ij}\, w\left(\left|x_{ij}\right|\right)\right)\right] \tag{13}$$

This is working well in the case of dam breaks but not in the case of hydrostatic. In this second one, we are preferring to keep model previously built.

Velocity Laplacian

Khayyer and Gotoh(2010) proposed to consider the mathematical definition of Laplacian, as a divergence of gradient in MPS.

$$\langle \nabla^2 v\rangle_i = \frac{1}{n_0}\sum_{j\neq i}\left\{\frac{\partial v_{ij}}{\partial r_{ij}}\frac{\partial w_{ij}}{\partial r_{ij}} + v_{ij}\left(\frac{\partial^2 w_{ij}}{\partial^2 r_{ij}} + \frac{D_s-1}{r_{ij}}\frac{\partial w_{ij}}{\partial r_{ij}}\right)\right\} \tag{14}$$

Due to kernel function

$$w(r) = \begin{cases} \frac{r_e}{r} - 1 & (0 \leq r < r_e) \\ 0 & (r_e < r) \end{cases} \tag{15}$$

Velocity Laplacian is derived as

$$\langle \nabla^2 v \rangle_i = \frac{1}{n_0} \sum_{j \neq i} \frac{(5 - D_s) r_e}{|r_{ij}|^3} v_{ij} \tag{16}$$

Thus, the calculation begins

$$\langle \nabla^2 v \rangle_i = \frac{(5 - D_s) r_e}{n_0} \sum_{j \neq i} \frac{v_j - v_i}{|r_{ij}|^3} \tag{17}$$

Study on pressure function

To reduce instabilities due to pressure calculation, previous study worked on a new formula. I choose to go on that way and find out the best parameters for each problem.

$$P(n_i) = \begin{cases} \frac{P_0}{\left(1 - \frac{n_{fs}}{n^\infty}\right)\left(\frac{1+\Delta}{\Delta}\right) - 1} * \left(\frac{n^\infty - n^{fs}}{n^\infty - n_i} - 1\right) & (0 \leq n_i < n^0) \\ \frac{P_0}{\left(1 - \frac{n_{fs}}{n^\infty}\right)\left(\frac{1+\Delta}{\Delta}\right) - 1} * \frac{n^0\left(n^\infty - n^{fs}\right)}{(n^\infty - n^0)^2} * \left(\frac{n_i}{n^0} - 1\right) + P_0 & (n_i \geq n^0) \end{cases} \tag{18}$$

With $n^\infty = n^0(1 + \Delta)$ and Δ is chosen by user.

This function is based on 3 experimental shapes (or points) we consider. Near the surface, pressure is reset to 0 and due to kernel function, density near the surface has a hyperbolic progressing. Then for density equal to n^0, pressure must be set at P_0, it is primordial point. Finally, pressure is proportional to density when density is bigger than n^0.

Delta parameter must be chosen wisely because it could modify wave shapes and create instabilities by removing impermeability. For example, when density is bigger than basic density, choose a little value of delta increasing pressure and pressure gradient will be tremendous. If it was chosen to little, pressure will be continuous, and movement will be quite different from the experiment. I decided to figure out the best parameter by comparison with experimental data.

Computation time

Even if we can decrease the computation time by using parallelization codes, we consider in this paper that the algorithm could be improved by reducing useless calculation as wall particles calculation. In fact, wall particles have a key role because they are managing density and impermeability, they have also an impact on the pressure of fluid particles. As we saw before, each particle is concerned by its neighbouring, so we can't just remove wall particle, build a "one-particle-wall" and then running the program like that. We must replace each role of wall particles and then create new particles that fit with the previous model.

Density linearization

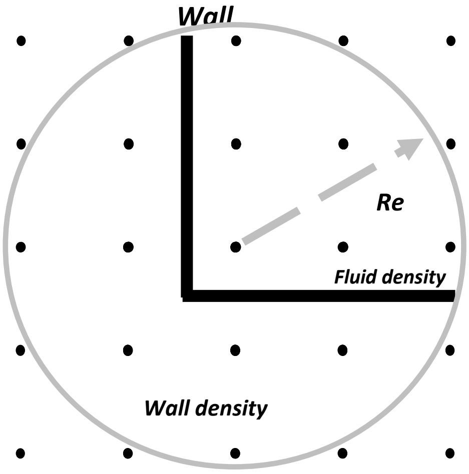


Figure 3: behavior of particle near walls

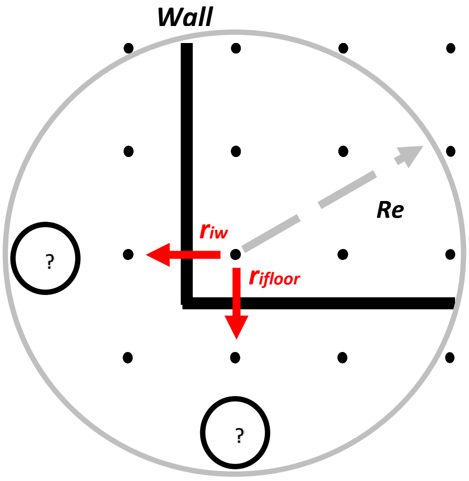


Figure 4 : determination of near wall particle and particle lack is pointed out.

In the first place, wall particles allow calculating density for particle near walls. As shown on the figure, each particle along the wall side are weighted by kernel functions and by summation we obtain the "wall" density. In the case in which we remove all the particles, except the first line, the summation will not be representative of the wall weight. Overall density will be less important for particles near walls instead of being the same.

If we consider particles by their locations, we show that we can create a link between wall density and distance to walls. Walls and not wall, due to edges in 2D (and 3D), it's necessary to introduce different function in the case of particles are constrained by 1,2 or 3 walls. In this paper we will just talk about function for 2 Dimensions, to extend in 3 Dimensions, procedures are at least equals.

Using previous simulations, I created a list in which we find: wall densities, distance to the nearest particle for each orientation. Means that for edges, 2 distances are saved, one for wall and the other for the floor. Then, particles were sorted by their distance and if they were considered as being near edges. Finally, I created a lot of matrix system to figure out the most convenient function to join all particles.

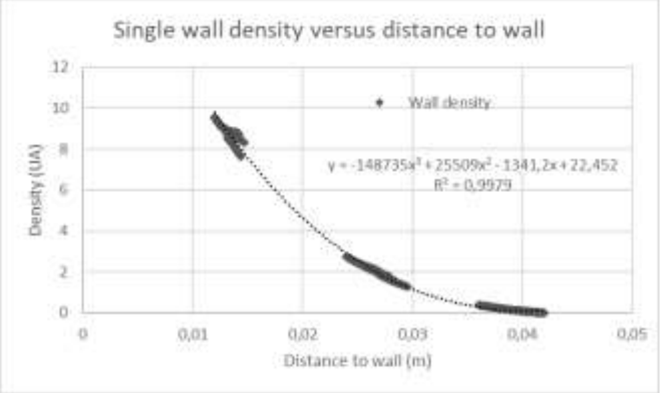


Figure 5 : New formula for particle near single wall

For particles near only one wall, I decided to determine a function as

$$Z(r_{iw}) = a \times r_{iw}^3 + b \times r_{iw}^2 + c \times r_{iw} + d \qquad (19)$$

Which is fitting well with experimental points. The most important error on the value is near the wall because our model is based on a discrete wall, not a continuous one. Also, to show this accuracy we obtain the following equation

$$Error(\%) = \max\left(\frac{Z^{exp}(r_{iw}) - Z^{poly}(r_{iw})}{Z^{exp}(r_{iw})}\right) = 5\% \qquad (20)$$

Concerning edges, I wrote this matrix equation :

$$A \times x = b$$

In which b is the wall density, x is a vector including constant parameter and A is a matrix picturing the system. Given that we have two parameters called $r_{i,wall}$ & $r_{i,floor}$, we want this kind of equations

$$Z\left(r_{i,wall} = r_W, r_{i,floor} = r_F\right) = x_1 \times r_W{}^2 + x_2 \times r_W{}^1 \times r_F{}^1 + x_3 \times r_F{}^2 + x_4 \times r_W + x_5 \times r_F + x_6$$

Our system is composed of m=15 000 equations, x has only 6 lines so we changed the A shape as :

$$A = inv(A' \times A) \times A'$$

And thus

$$x = inv(A' \times A) \times A' \times b$$

The matrix system is solved by OCTAVE software.

I did some experiments about the most convenient shape for A system, but in the end, I figure out that the best function has to be a third polynomial degree.

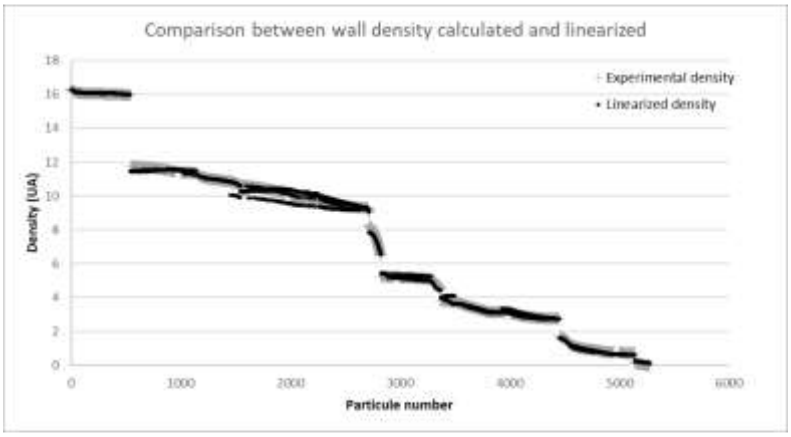


Figure 6 : Prior particle densities compares to new ones, calculated by linearized functions

By comparison, the polynomial form appears to be a good compromise and we should keep it to decrease calculation time broadly. Here is the function used :

$$Z(r_W, r_F) = x_1 r_W^3 + x_2 r_W^2 r_F + x_3 r_W r_F^2 + x_4 r_F^3 + x_5 r_W{}^2 + x_6 r_W{}^1 r_F{}^1 + x_7 r_F{}^2 + x_8 r_W + x_9 r_F + x_{10} \qquad (21)$$

And x vector is

$$\vec{x} = (\mathbf{-275281; 155818; 129318; -115105; 30071; -19788; 17700; -739; -1047; 33,6})$$

Velocity Laplacian modified

In addition to the High-Laplacian improvement which would concern all the fluid particle, we developed a new wall formula, based on previous Japanese papers. Considering that wall particles are still removed, it was mandatory to keep the effect of displaced particles.

$$\langle \nabla^2 v \rangle_i = \langle \nabla^2 v \rangle_{i,fluid} + \langle \nabla^2 v \rangle_{i,wall} \tag{22}$$

Wall terms were firstly

$$\langle \nabla^2 v \rangle_{i,wall} = \frac{2d}{n^0 \lambda^0} \sum_{j \in wall} \left[v_{ij} . w(|x_{ij}|) \right] \tag{23}$$

Considering velocity between particle i and j as the difference between velocity of i and velocity of j, we showed that

$$\langle \nabla^2 v \rangle_{i,wall} = \frac{2d}{n^0 \lambda^0} \sum_{j \in wall} \left[(v_j - v_i) w(|x_{ij}|) \right] \tag{24}$$

Owing to the general movement of particle j, and without angular moment, velocity Laplacian due to the wall is

$$\langle \nabla^2 v \rangle_{i,wall} = \frac{2d}{n^0 \lambda^0} (v_{wall} - v_i) \sum_{j \in wall} \left[w(|x_{ij}|) \right] \tag{25}$$

We reduced this equation as

$$\langle \nabla^2 v \rangle_{i,wall} = \frac{2d}{n^0 \lambda^0} (v_{wall} - v_i) \times Z_{wall}(r_{iw}) \tag{26}$$

Instead of calculating a big summation which requires to loop around the neighbouring area, this function only operates one time. Calculation time is inevitably decreased, and this equation fits well with our new model of density.

Pressure Gradient modified

Pressure Gradient was quite difficult to deal with. As shown in the bibliography, many methods have been tried and according to our previous models some of them are not available. Mirror particles, for example, gave us interesting results but matrix processing asked too much time and we must found a better solution.

Based on the accurate polygon wall representation, I chose to use the same approach based on fictive wall particles, but without building them. Given that our model allocates pressure to all wall particles, which constrains fluid particles and allow them to stay behind the wall, I decided to make “pressure points” at every location in grey on the figure. Accordingly, I allocate pressure based on existing wall particles. Means that pressure points have not the same values as removed particles.

Due to this new pressure field, it is mandatory to adjust pressure points by another weighted function. This function has to take wall geometry into account. Flat walls are easy to imagine in terms of internal forces, so the pressure will be more important on the ground and less important near the fluid boundary. In the case of fast velocities, sometimes impermeability is not safe, and I added external forces rely on particle bound.

$$\langle \nabla P \rangle_i = \langle \nabla P \rangle_i^{fluid} + \langle \nabla P \rangle_i^{wall} + f_{imp} \tag{27}$$

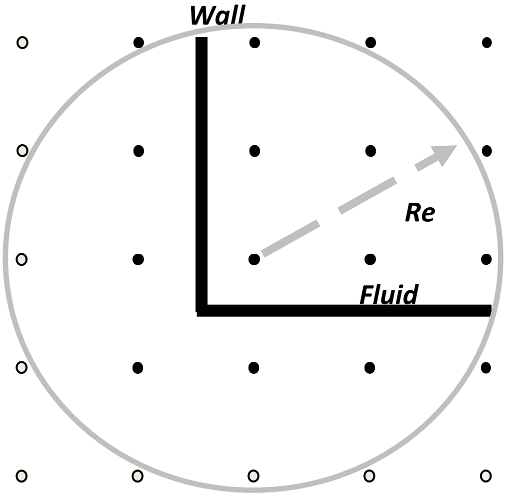


Figure 7 : virtual calculation of pressure points

Previously, pressure gradient was

$$\langle \nabla \mathrm{P} \rangle_i = \frac{d}{n^0} \sum_{j \neq i} \left[\frac{\mathrm{P_i} + \mathrm{P_j}}{|x_{ij}|^2} \, x_{ij} \, w(|x_{ij}|) \right] \tag{28}$$

And wall pressure gradient is now modified as:

$$\langle \nabla \mathrm{P} \rangle_i^{wall} = \frac{d}{n^0} \sum_{j \in wall} \left[\left(\frac{\mathrm{P_i} + \mathrm{P_j}}{\left|x_{ij}\right|^2} \, x_{ij} \, w\left(\left|x_{ij}\right|\right) \right) \times (1 + w_1 + w_2 + w_3) \right] \tag{29}$$

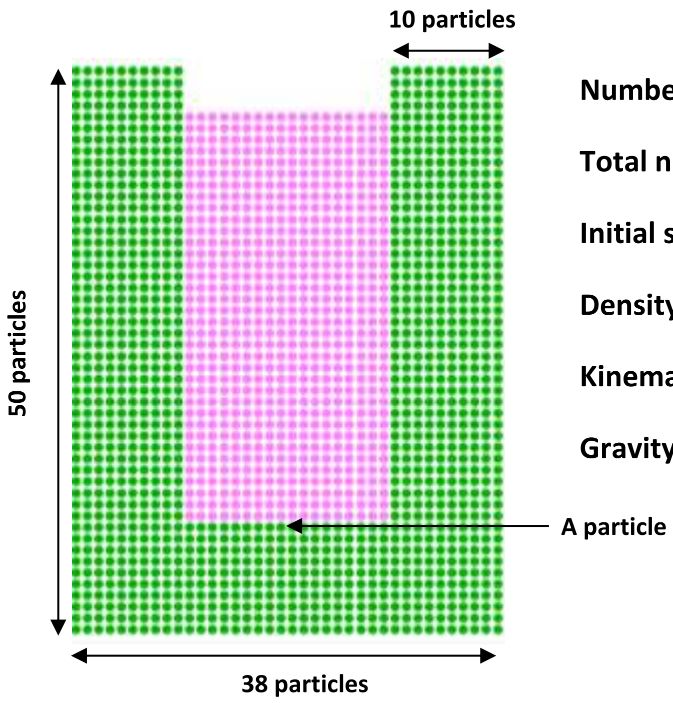
Where weight functions on the right part are less important than 1.

III. Experiments and Results

In this part, we will compare the previous model and our new one. In addition, an experimental study named “An Experimental Study of the Collapse of Liquid Columns on a Rigid Horizontal Plane” (MARTIN and MOYCE, 1952) is used to show the accuracy in the case of a dam-break problem.

Hydrostatic

Hydrostatic problem is one way to show the accuracy of our simulation. Indeed, it allows validating the shape of pressure representation because the viscosity term in Navier-Stokes formula would be quite little in front of the pressure term.



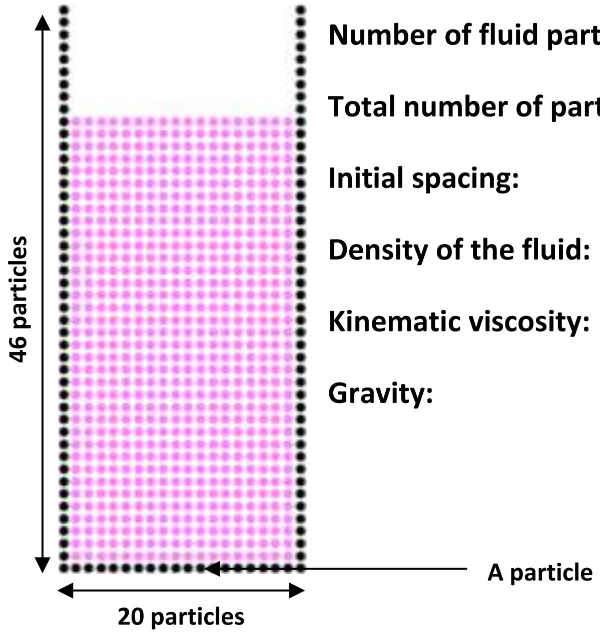
Number of fluid particles:	*648*
Total number of particles:	*1828*
Initial spacing:	*0.012 m*
Density of the fluid:	*1000 kg.m^{-3}*
Kinematic viscosity:	*0.001 m^{2}.s^{-1}*
Gravity:	*9.8 m.s^{-2}*

Figure 8 : Initial configuration of hydrostatic experiment in E-MPS previous model

Figure 8 shows the first shape of our experiment concerning hydrostatic. Green particles have properties of “particle wall type”. Which means they have the same process of calculation then pink fluid particles, but they are not allowed to move and their pressures are only calculated by summation of fluid particles in their neighbourhoods. They have been built to ensure impermeability and to constrain pink fluid particles in the tank.

Note two major things: firstly, kinematic viscosity is 1 000 times better than real viscosity. This is to solve consistent enquiries in our models. Secondly, green particles are two times more important than fluid particles. This is just to show the action of wall particle type on the computation time. Also, the appropriate number of green particles depends on kernel function distances r_e , and spacing between particles. Here 4 particles wall would be better in term of computation.

Now our new model is quite little compared to the previous one, but we have the same number of fluid particles and we deal with the black particles with properties of “wall particle”. Difference between green particles and black ones concerning density calculation. Black wall particle does not take time to calculate its density, it is only focused on pressure calculation.



Number of fluid particles:	***648***
Total number of particles:	***759***
Initial spacing:	***0.012 m***
Density of the fluid:	***1000 kg.m^{-3}***
Kinematic viscosity:	***0.001 m^{2}.s^{-1}***
Gravity:	***9.8 m.s^{-2}***

Figure 9 : Initial configuration of hydrostatic problem in E-MPS, new model

To analyze the accuracy, we proposed to visualize pressure which occurs on the ground. As a current hydrostatic problem, we know the theoretical shape of pressure which is :

$$P_A = \rho g \times (h_{freeSurface} - h_A) \tag{30}$$

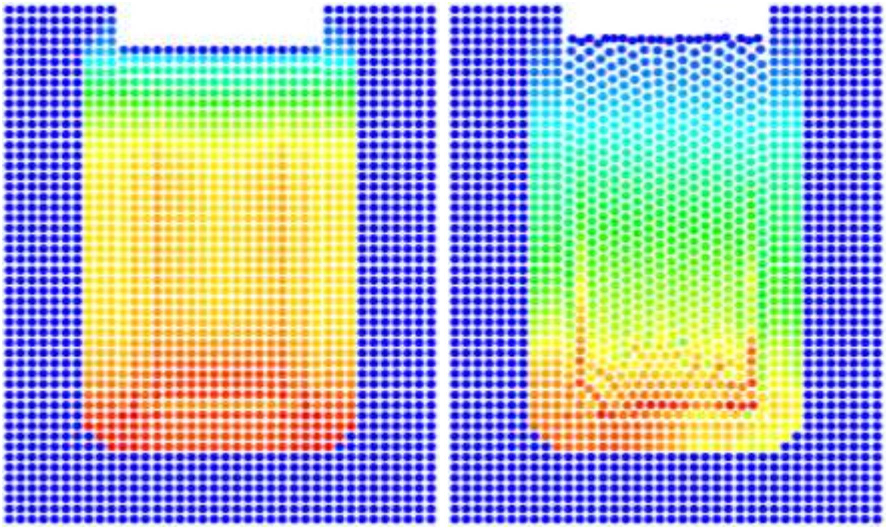


Figure 10 : Left : time is 0.1s ; Right : time is 1.5s

Figure 10 presents the prior model at 2 different time steps. While the beginning of the simulation shows a good pressure scattering, the end of the simulation appears to be more ambivalent. Pressure field on the ground is dispersed and also edge pressures are asymmetric. At the top of the column, we observe fluid particles repelled by the wall. Note that in this model, CMPS is not working.

Figure 11 is the new model with the same steps than figure 10. Beginnings are at least slightly different from previous experiments, we notice that the pressure field in the centre appears to be more important than nearby walls.

At 1.5s, the scattering of pressure shows relevant points. Firstly, it is well-cut step by step, with high pressures on the ground and low pressures at the fluid/gas interface. Then edge pressure seems to be more important than pressure for the same height, but it is less significant compares to figure 10. Only one or two particles are concerned. Finally, it appears that wall particle repels more fluid particle than a prior experiment, this is still a problem due to interface treatment and pressure wall representation.

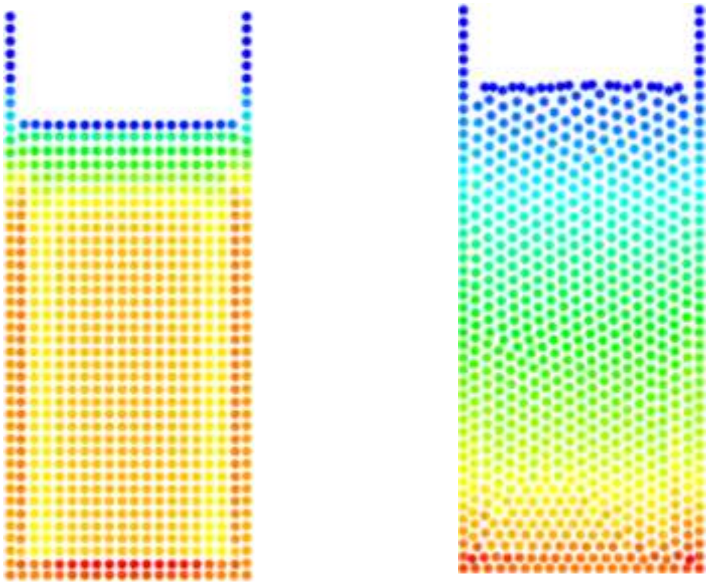
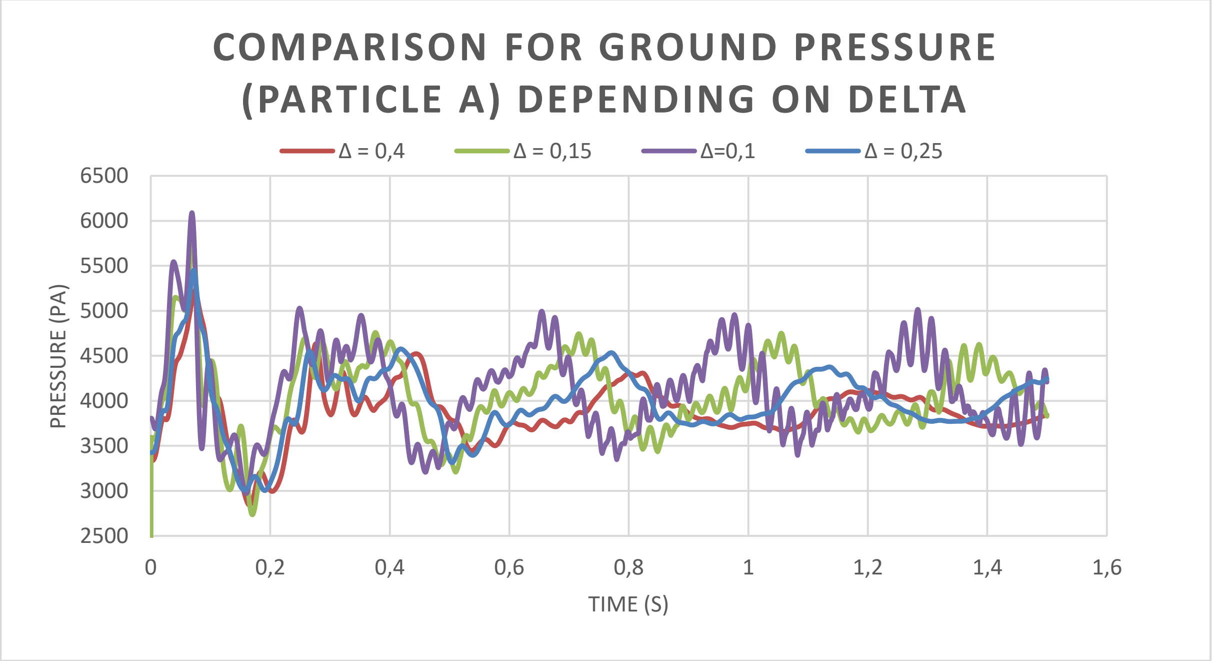


Figure 11 : Left: time is 0.1s; Right: time is 1.5s

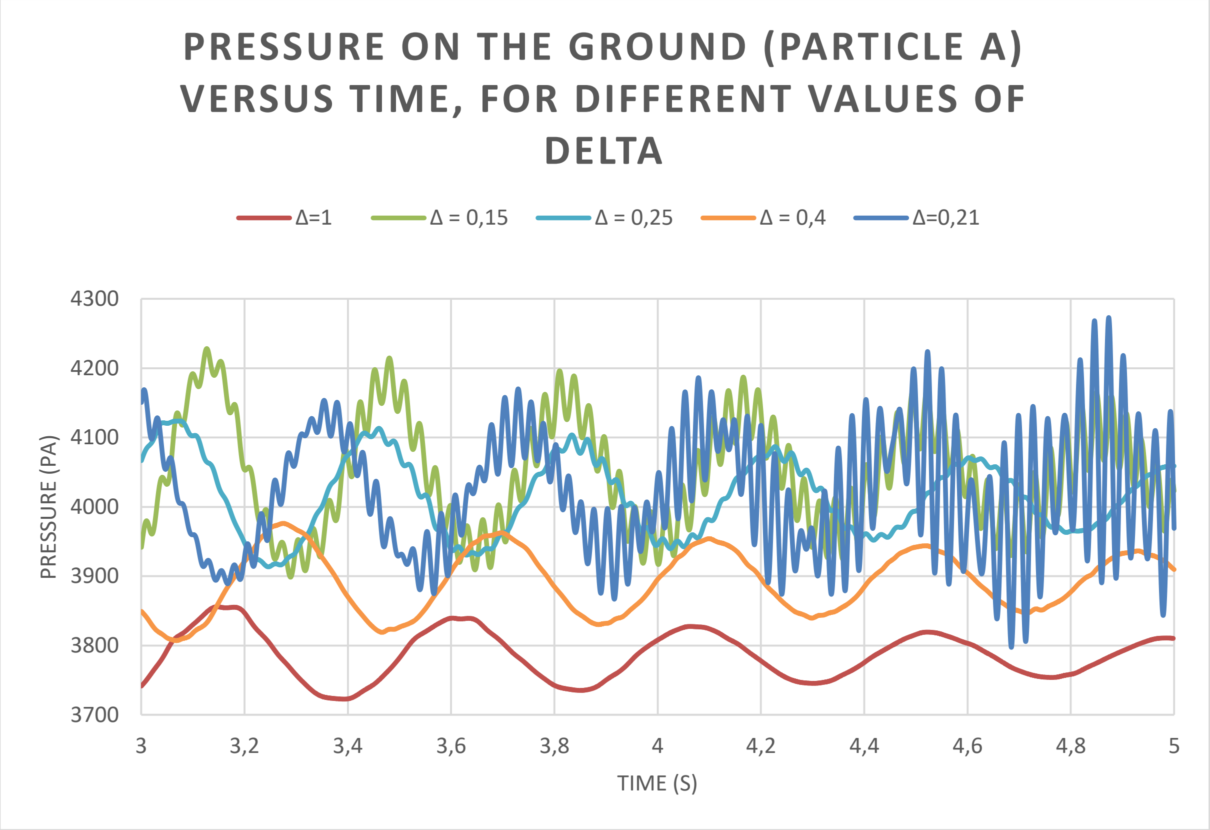
Accuracy

The shape of the hydrostatic problem seems interesting, but it surely depends on set up parameters. Pressure calculation needs a delta value to be launched, and it relies on empirical results. As we know the theoretical pressure for the A-point, it is possible to look for the best empirical delta which fit with the theory.



Graphic 1: Pressure of A-point depends on delta values.

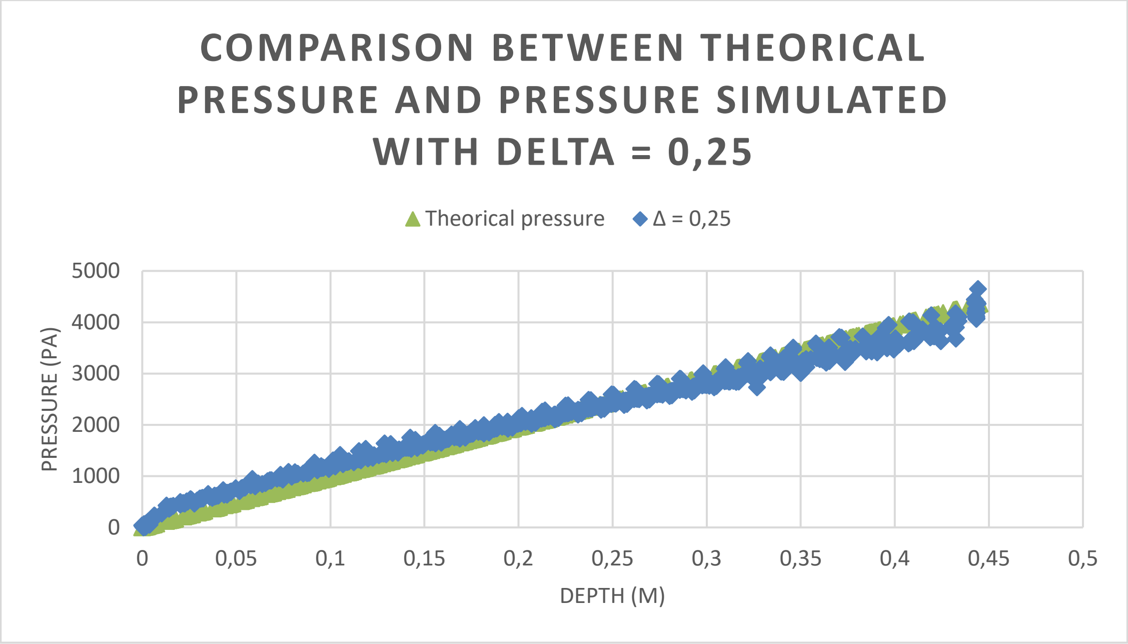
Graphic 1 shows pressure volatility, furthermore for some delta values the stability is not enough. Delta values less than 0.15 are fluctuating with high frequency in addition of lower frequency. Over 0.15, high frequency is less important, and we must choose the best parameter by referring to the theoretical value.



Graphic 2: another scale for A pressure analysis

After 3 seconds some new features are relevant, for example with delta value around 0.2, pressure begins to fluctuate. Something interesting is the mean value according to Delta. When delta increases, mean pressure is decreasing. However even oscillations are less important, mean values are drifting away from the theoretical value which is supposed to be 4,2 kPa.

I chose to keep delta equals to 0.25 because it seems to be a good middle ground between oscillations and mean pressure value. To validate this analysis, we propose to study the whole field of pressure.



Graphic 3: Analysis of the whole pressure field in blue, compared to the theoretical value in green

Pressure field with delta equals to 0.25 is similar than theoretical pressure. We point the shape out which is nearby the real one. Nevertheless, the beginning of the function does not follow this same proportionality. That means we can still improve the pressure formula.

With that being said, the next step of our work is to show computation time improvements.

Computation time

We remind that calculation time depends on previously developed clustering of neighbours, so the figure submits this number. We must bear in mind that previous model was not well updated, so that time consuming will be irrelevant for the hydrostatic approach. The purpose of this study is to show that a lot of wall particles are requesting a big amount of time.

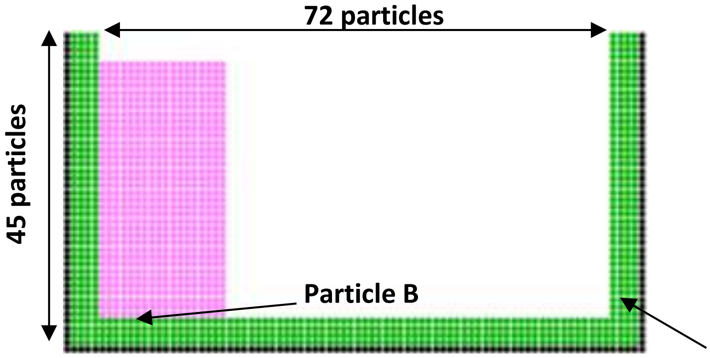
Number of candidates	*Time for the Previous model (s)*	*Time for the New model (s)*	*Iteration*
17	95,06	50,62	7500
12	90,23	45,09	7500

As we expected, the new model reduces from 45% to 50% processing time. Also, neighbouring parameters do not modify this result tremendously.

We just have shown that our new one wall model is faster and give accurate results. The pressure gradient is also well represented, we can now work on the dam break problem.

Dam break

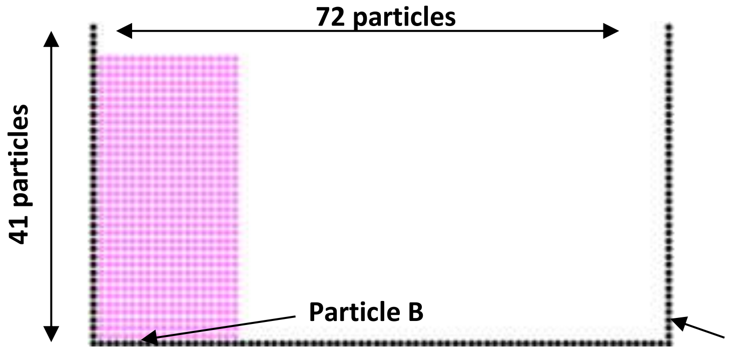
This time, previous work is well built in term of optimization, walls are 4 particles thick.



Number of fluid particles:	*648*
Total number of particles:	*1458*
Initial spacing:	*0.012 m*
Density of the fluid:	*1000 kg.m^{-3}*
Kinematic viscosity:	*0.001 m^{2}.s^{-1}*
Gravity:	*9.8 m.s^{-2}*

Figure 12 : Initial experiment for dam-break problem

In comparison with our new model, the number of particles is 45% less important.



Number of fluid particles:	*648*
Total number of particles:	*802*
Initial spacing:	*0.012 m*
Density of the fluid:	*1000 kg.m^{-3}*
Kinematic viscosity:	*0.001 m^{2}.s^{-1}*
Gravity:	*9.8 m.s^{-2}*

Figure 13: New model with one wall particle layout

As it was too big to stay in the main document, you could find the results and the comparison of pressure field in annexe 1. To sum up, due to the initial pressure field, some particles are exploding at the beginning, which creates some unnatural movements in the left tank area. Then wave development seems to be fitting with the reality, even when the wave crash happened. After this time, new wave shape is merging as expected. But, it is quite complicated to know if the shape is real because particle behaviour nearby left edge is strange. We propose in the next part to verify the accuracy of all these steps.

Accuracy

MARTIN and MOYCE provide experimental results concerning wave development for a dam-break problem. They carry out different shape experiments in order to define dimensionless outcomes.

So that, to contrast simulated results and know the better delta parameter, I made the same dimensionless operations which are

$$t' = t * \sqrt{\frac{2g}{L_0}} \tag{31}$$

$$L' = \frac{L}{L_0} \tag{32}$$

With initial water column height equals to L_0

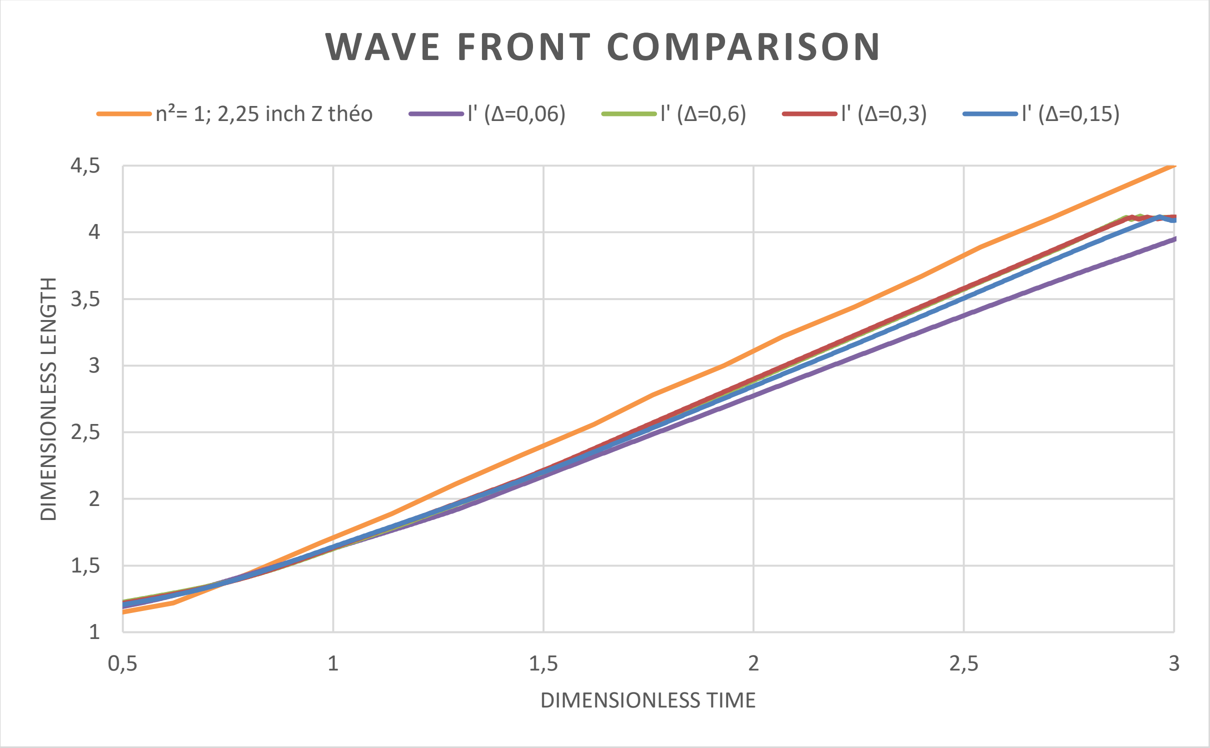


Figure 14: Comparison between experimental values of way front advance and simulated ones which depend on delta parameter

It appears that the best delta value is around 0.3. Underneath, waves are not running well; and above 0.3 major inquiries are to deal with exploding particles in the near beginning. In these respects, I decided to use 0.3 value for delta parameter in the following experiments.

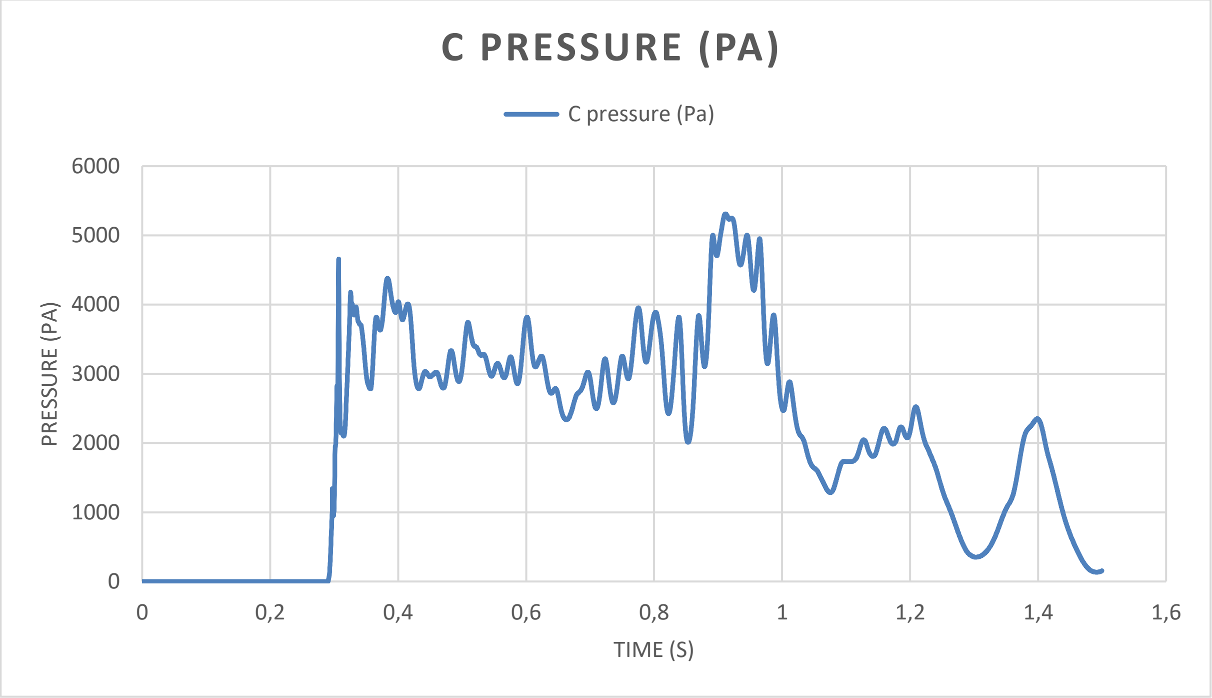


Figure 15: Pressure of C-point versus time

Similarly, other results are available concerning pressure on the right side of the tank area. It is very important for geotechnics studies and shape obtains should be the same as experimental ones. We compare our outcomes with another experiment made by HU and KASHIWAGI (annexe 2) and it appears that our simulation has some difficulties to deal with the first wave impact. The pressure value is 3 times less important than expected. I think that in a one-particle wall, calculation of pressure should be different, and it needs to be reviewed. By the way, the broad shape is quite the same, with another second wave impact slightly different from the experiment.

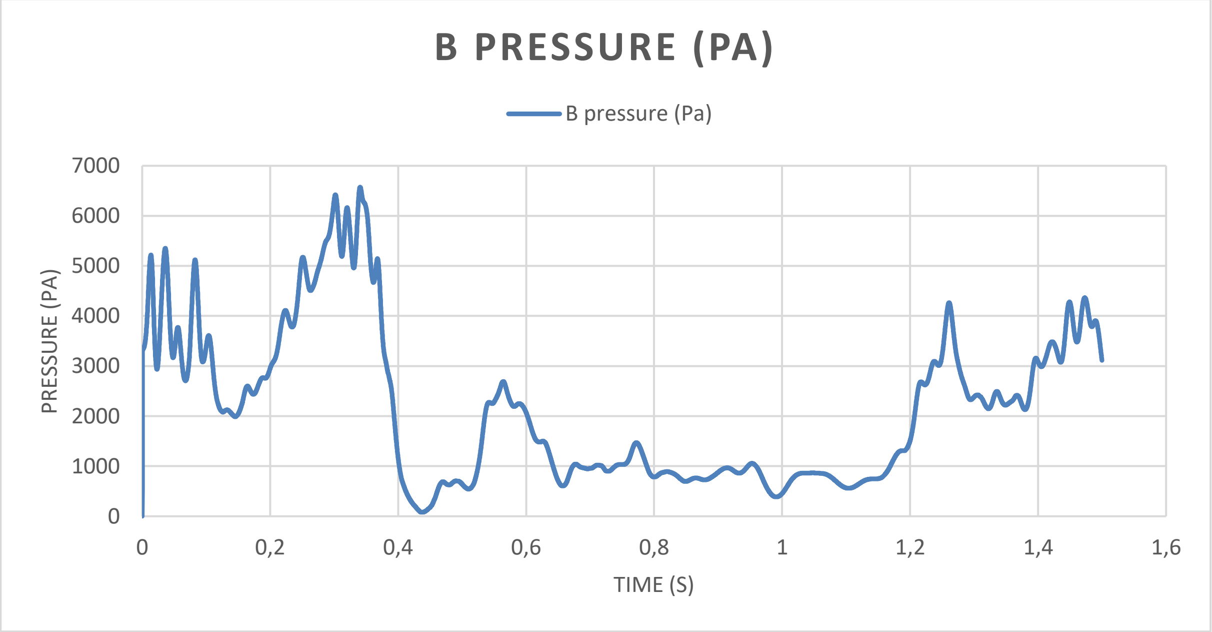


Figure 16: Pressure at B-point versus time

This other graphic is used to validate pressure exploding, in the near beginning. The maximum value is reached after explosions when particles are repelled by gravity. Minimal one is arriving just after, and it is lower than expected. Pressure near 0 means that it is a free surface particle, which is not entirely true after a one wave development. It is like water particles disappear from this area, pressure should be a little more important in that case. Then flux development is quite similar to the one we can imagine. It should be interesting to find some experimental values to compare with our sights.

Computation time

Now concerning simulation time, comparison with the previous model is interesting because it was efficient in terms of time cost. The number of neighbours was optimized to keep accuracy and it was providing a good time-consuming contrast to previous models.

ITERATION OF CANDIDATE LIST	NEW METHOD (S)	PREVIOUS METHOD (S)	NUMBER OF ITERATION
8	49,13	76,22	7500
12	49,45	76,50	7500
16	49,47	80,05	7500

Outcomes are unexpected to be at this level. Gain is near 35% and consistency is saved. This one particle wall provides better results without using hardware or soft parallelization which should be used soon.

Conclusion

To sum up this paper, it appears that Lagrangian procedures are useful in the case of fluid movements with free surface boundaries. I was able to add some enhancements by using physical laws in discretized ways, but major problem was to keep consistency by using computing tricks as unnatural forces etc... Modelling is still composed by the compromise between accuracy and calculation speed, and this paper points out that we can improve speed without losing a lot in terms of accuracy by using linearization and moreover by defining the thin wall. This last enhancement was difficult to build because it should use computing tricks and put distance with “real” laws. After a lot of research, it was mandatory to imagine another way of calculation to provide better outcomes. I tried some experimental model but the last one is for the moment the best one. Actually, it appears to be limited and a new way of pressure test needs to be done to be used by comparison with experimental results.

3 dimensions problems were not discussed here because it demands too much time to run. Thanks to my new research, I hope that time consuming could be less important, but it would be needed to create linearized functions. Also, 3D models have not any experimental results which could be used to validate the accuracy.

This internship was also a great opportunity to figure out what is a research work in the field of numerical analysis for fluid mechanics. Wide guidelines are complicated to deal with, but I tried to find some ways to answer them. The main problem was to understand how physical expectations are reached by computing, which was sometimes hard because MPS papers are most of the time writing in the Japanese language. Despite this complex environment, I learnt a lot from all the people I met and I’m finally grateful to have lived this research internship in Okayama University Lab.

Bibliography

Gotoh, H., & Okayasu, A. (2017). Computational wave dynamics for innovative design of coastal structures. *Proceedings of the Japan Academy, Série B Physical and Biological Sciences*, 525-546.

Koshizuka, Shibata, & Murotani. (2014). *Introduction to particle methods.*

Ma, Q., Zhou, Y., & Yan, S. (2016). A review on approaches to solving Poisson's equation in projection-based meshless methods for modelling strongly nonlinear water waves. *Journal of Ocean Engineering and Marine Energy*, 279-299.

Martin, J., & Moyce, W. (1952). *An Experimental Study of the Collapse of Liquid Columns on a Rigid Horizontal Plane.*

Mitsume, N., Yoshimura, S., Murotani, K., & Yamada , T. (2015). *Explicitly represented polygon wall boundary model for the explicit MPS method.*

Annexe

Annexe 1: Dam-break comparison

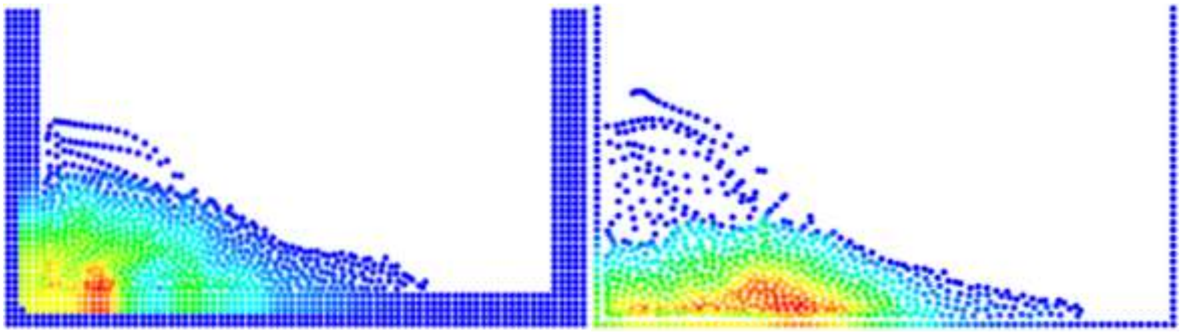


Figure 17 : Left: Previous model; Right: New Model; time equals to 0.25 s

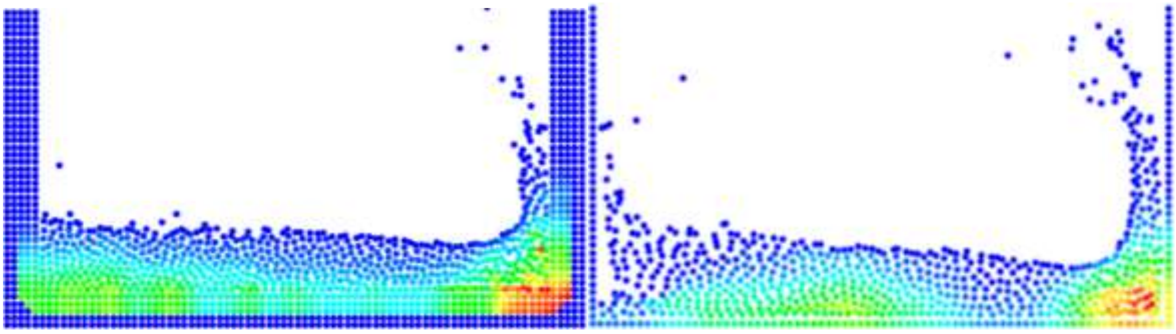


Figure 18 : Left: Previous model; Right: New model; time is 0.5s

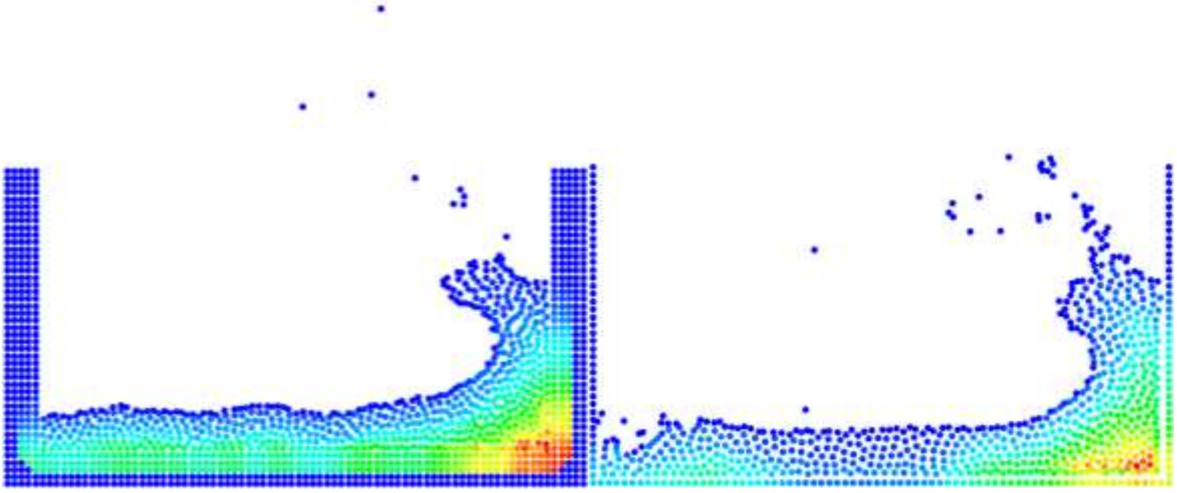


Figure 19 : Left: Previous model; Right: New model; time is 0.75s

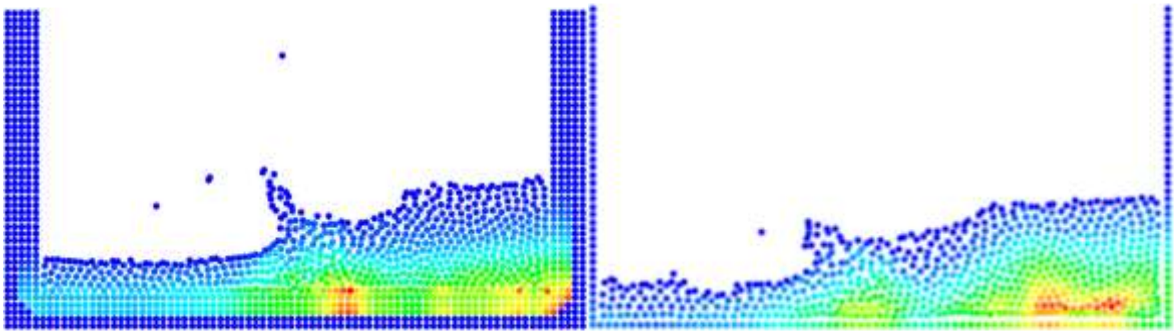


Figure 20 : Left: Previous mode; Right: New model; time is 1s

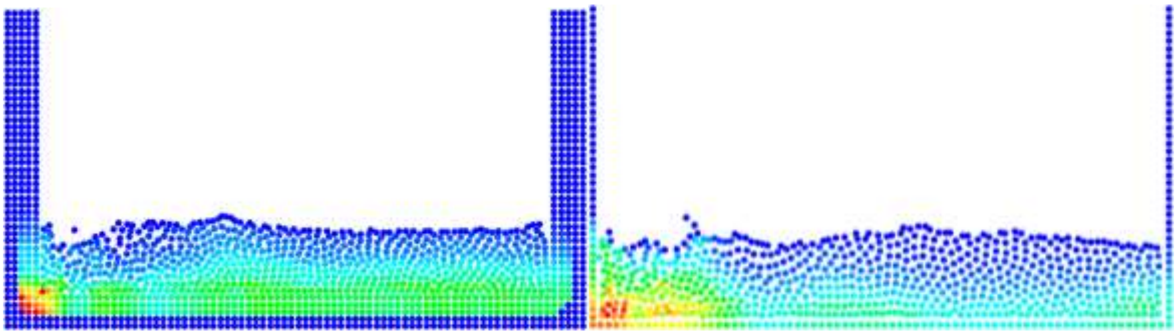


Figure 21 : Left: Previous model; Right: New model; time is 1,25s

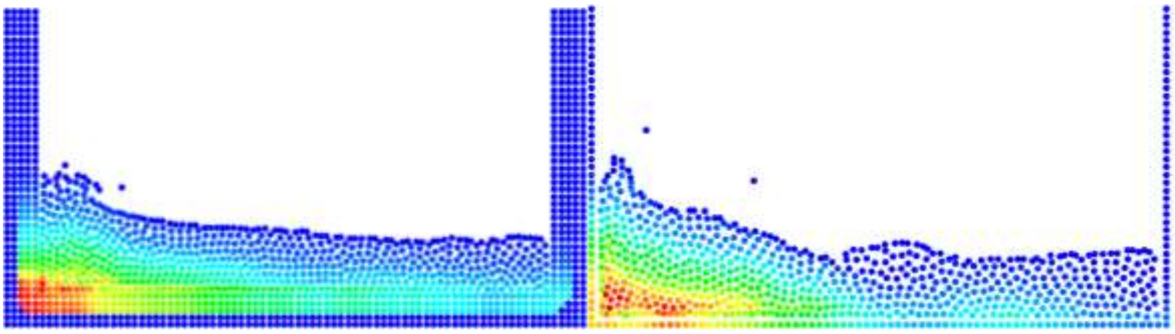
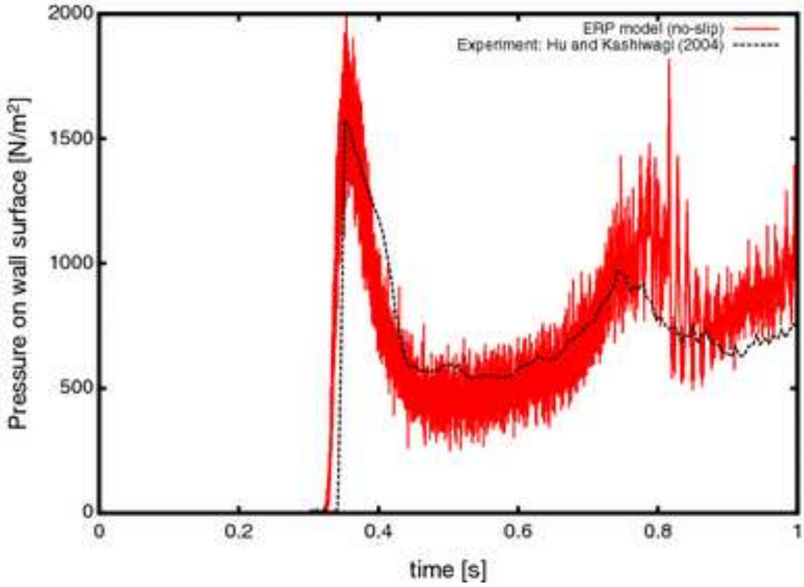


Figure 22: Left: Previous model; Right: New model; time is 1,5s

ANNEXE 2: HU AND KASHIWAGI EXPERIMENT


ERP model (no-slip)
Experiment: Hu and Kashiwagi (2004)
Pressure on wall surface [N/m^2]
time [s]