有限要素法を用いた自由境界を持つ流れの二次元シミュレーション

Simulation of Flow Problems with Free Boundaries in 2D using the Finite Element Method.

○ Shi Han NG, 電通大, 東京都調布市調布ヶ丘 1-5-1, E-mail：shihanng@matuttis.mce.uec.ac.jp
Hans-Georg MATUTTIS, 電通大, 東京都調布市調布ヶ丘 1-5-1, E-mail：hg@mce.uec.ac.jp
Shi Han NG, The University of Electro-Communications,
Graduate School of Informatics and Engineering, Chofu, Chofugaoka 1-5-1, Tokyo 182-8585.
Hans-Georg MATUTTIS, The University of Electro-Communications,
Graduate School of Informatics and Engineering, Chofu, Chofugaoka 1-5-1, Tokyo 182-8585.

We simulate flow problems with the Finite Element method for free boundaries whose motion is obtained by time integration of the surface's flow velocities. The volume conservation is reasonable, and for the low-viscosity-range, we obtain good aggreement with the lubrication approximation. For the decay of a square fluid column, we find that for high viscosity, the corner of the square travels towards the wave front, while for low viscosity, it stays where it is, consistent with experimental results.

1. Introduction

Many technical problem in the dynamics of complex fluids, like multi-phase-flow and realistic flow in porous media and granular materials deal with the interaction between fluids and interfaces, rather than with the dynamics of fluid alone. Our long-term aim is to simulate multi-phase flow problems in granular materials "microscopically", i.e. by simulating the particles individually as polyhedra via the discrete-element simulation, and the fluid flow inside the fluid space via the finite element method. The "first principles" approach of simulating the microscopic interactions allows to test the validity of approximations common in the field.

The implementation of the following algorithms is in MATLAB[1] . We give no CPU-usage data and timings, as we are more interested to "get it right" than "get it fast". Once we have a satisfying algorithm, we can focus on the performance optimization. Though we are aware that this approach may be computationally costly, the CPU-performance is still increasing exponentially, and with the advent of GPGPU-cards and Multicore-PC-processors, it seems that substantial sustained computing power becomes available in the hands of researchers far away from supercomputing centers with their cumbersome application procedures. Our intention is, rather then focusing on massive-parallel implementations with time-consuming parallelizations, to develop a methodology where we can reduce the degrees of freedom without lowering the accuracy of the geometrical description, and to use time-integration methods for which the time-step is only limited by the time-scale of the physical phenomena, rather than by the lamentable stability requirements of easy-to implement algorithms.

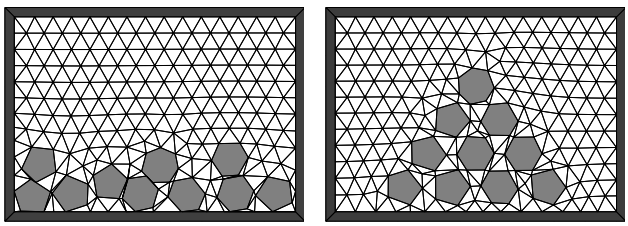

Fig. 1: Snapshot of a Discrete-Element method for hexagonal particles with triangular grid in which a pore-space which could hardly be discretized via a quadrilateral meshes.

2. Simulation method for the fluid part

Because we would like to run the simulation on arbitrary geometries, especially on pore-spaces which are a result of the dynamics of granular particles, as well as the surfaces, Finite Elements (FE) are the method of choice for the discretization, as they can be used also with triangular meshes. Meshes in the pore-space may become arbitrarily small or at least arbitrarily intractable for quadrilateral grids (see Fig. 1), without actually contributing much to the flow field. Therefore, we would like not to be limited with our step-size by the mesh-size due to the Neumann-stability conditions for explicit integrators, so an implicit integrator seems preferable. Most of the flow problems involving suspended material will be incompressible, which from the point of theoretical approaches means that the pressures turn out as Lagrange-multipliers[2] which inhibit the velocities from compressing the fluid. In numerical analysis, these types of problems are called Differential Algebraic Equations (DAE) and can be written in the following form:

$$\begin{pmatrix} M & G^T(q) \\ G(q) & 0 \end{pmatrix} \begin{pmatrix} u' \\ \lambda \end{pmatrix} = \begin{pmatrix} f(q,u) \\ -g_{qq}(u,u) \end{pmatrix} \quad (1)$$

This is an equation of motion for the masses M, the Jacobian of the constraints G, the time derivatives of the velocities u' and the external forces f, as well as the Hessian of the constraint (rewritten with some vector analytical identities) $g_{qq}(u,u)$. The DAE-form for the Navier-Stokes equation by the FEM-formulation with Backward-Difference Formula of second order (BDF2) by Gresho and Sani[4] is

$$\begin{bmatrix} \frac{1}{\Delta t_n} M + K + N(u_{n+1}) & C \\ C^T & 0 \end{bmatrix} \begin{pmatrix} u_{n+1} \\ P_{n+1} \end{pmatrix} = \begin{pmatrix} \frac{1}{\Delta t_n} M u_n + f_{n+1} \\ g_{n+1} \end{pmatrix}, \quad (2)$$

where M is the mass-matrix of the problem, K is the stiffness matrix, N is matrix of the non-linear terms, f_{n+1} are the external forces , the u_{n+1} are the flow velocities and the P_{n+1} are the pressures. We do not use any up-winding. Comparison with eq. (1) shows that the pressures in eq. (2) indeed turn out as the Lagrange

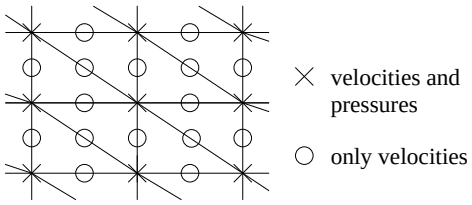

Fig. 2: Friedrichs-Keller grid ("FEM-grid") for P2P1-elements, where the velocities are given at the sites with the crosses and the circles, and the pressures are defined on the sites with the crosses only.

parameters. Therefore, instantaneous changes of the pressure are possible, there is no necessity, as in the case for the Marker-and-Cell method, for a smooth iteration, which behaves effectively as an "equation of motion" for the pressures and leads to restrictions with respect to the timestep and the accuracy. Though pressures which fluctuate strongly are an indication of a too large timestep, it is reassuring to have method which indicates too large timesteps without blowing up.

Eq. (2) is implicit, so that the values of the new timestep $n+1$ must be compute via Newton-Raphson iteration from the previous values, where the "Matrix" is the Jacobian of the velocities. Usually we need about three iteration for fixed boundary conditions. As the BDF2-method is not self-starting, i.e. at the first timestep, additionally to the initial conditions data from "before" the initial conditions are needed, and additionally, consistent initial conditions are needed for the pressure, so that the incompressibility condition is not violated. Therefore, we start our solution from a stationary state, computed via Newton-Raphson iteration for the stationary Navier-Stokes equation. The approach of integrating out the time dimension via a solver for ordinary differential equations (ODE), instead of discretizing the time direction also via Finite Elements is sometimes referred to as "semi-discretization." For the spatial discretization, we use P_1P_2-elements, i.e. the pressures are approximated with affine elements, while the velocities are approximated with quadratic elements (see Fig. 2).

3. Philosophy of the surface modeling

Conventional Eulerian, i.e. grid based approaches leave the grid "as it is" and introduce additional data structures to trace the surface. In the Marker-and-Cell (MAC) method[5, 6], these "markers" are used to interpolate the surface between the grid points for which the flow is computed. More recent methods solve partial differential equations to determine the position of the surface with various approaches (level set method[7], volume of fluid method[8], advection equation[9]). As we have a Finite Element simulation at hand which allows to restructure and re-mesh the grid in every timestep[10], we would rather stick with our FEM-grid without introducing additional data structures for a variety of reasons:

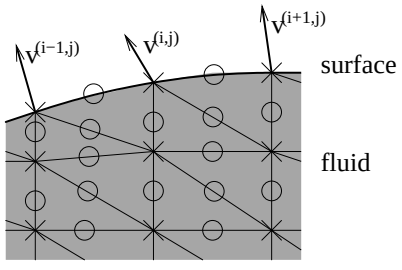

Fig. 3: Gravity wave and discretization near the surface (inset) with normal velocities set to zero (circles) and velocities computed (crosses).

"Economy of thought": "Ockham's razor" is the principle that unnecessary principles should not be introduced to solve a problem: When our grid defines the boundary already, why introduce another "grid" for the boundary. In particular, we are rather averse to introduce an entity which does not exist in nature: The fluid surface should also be its physical boundary, so the surface information should also be sufficient to model the boundary.

Mechanic Impedance: When we introduce, additionally to the boundary, a geometric entity with which the fluid interacts, there is a risk that this will alter the mechanic impedance (the way that mechanic signal propagate over the boundary, or how they are reflected there) without the possibility to control or evaluate this effect.

Practicability: In the future, we want to simulate porous media with particles and many thousand fluid interstices etc. Introducing such an immense number of additional surfaces, including the corresponding overlaps or intersections may lead to non-unique algorithmic choices.

Analogy: Finite Elements methods in structural mechanics don't need additional data structures to describe the surfaces. So shouldn't we be able to do without additional constructs als in fluid modeling?

Another aspect is that our semi-discrete implementation is already a formulation of the FEM-equations with ordinary differential equations. Because the FEM-part of the simulation yields the velocities of the fluid at every lattice point, we should therefore be able to calculate new positions of the lattice points at the surface, and use the new position of the lattice points without any additional data structure. So if the position of the surface at the discrete timestep $t = n$ would be given for the lattice site (i, j) as $\mathbf{x}_n^{(i,j)}$, and we had obtained the velocity by our FEM-procedure on the free surface as $\mathbf{v}_n^{(i,j)}$ (for a sketch, see Fig. 4) we could compute the new position of the lattice point via an Euler-integration for a time-step τ as

$$\mathbf{x}_{n+1}^{(i,j)} = \mathbf{x}_n^{(i,j)} + \mathbf{v}_n^{(i,j)}\tau. \tag{3}$$

Of course, due to the bad stability- and accuracy-properties, it will be advisable to look for ODE-solvers with better properties. Because we do not want to deform our P_2P_1-elements, we move only the corners of the triangles and interpolate the center points of the edges accordingly.

With respect to the boundary conditions one modification in comparison to conventional fluid simulations becomes necessary: Imagine a gravity wave (a wave

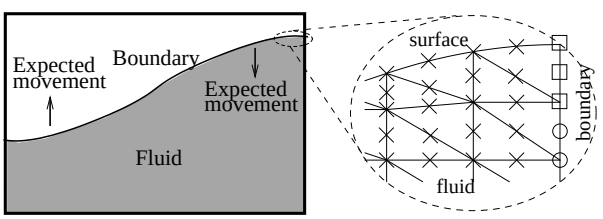

Fig. 4: Gravity wave (left) and schematic discretization near the surface in the inset, to the right. Grid points with normal velocities and tangential velocity set to zero (○), all velocity components velocities computed (×) and the grid points with normal velocity set to zero while the tangential velocity is computed (□).

which moves under the influence of gravity, not a oscillatory solution of the field equations of general relativity, of course) in a container of length l and wave length $2l$, with a phase so that the extrema are at the boundary, see Fig. 4. Obviously, the surface must move down at the maximum, and must move up at the minimum. While the tangential velocities for the elements on the boundary away from the surface must be set to 0 "inside" the computational domain, for the element on the surface the tangential component of the velocity must be computed, else no movement of the surface of the boundary is possible. At the same time, the normal velocity for this element must be zero. In an earlier stage of the program development, we forgot and obtained curious, elongated triangles. This formulation with moving elements has the advantage that, as eq. (2) allows the inclusion of arbitrary external forces f_{n+1}, additional models for the surface tension can be implemented in Finite-Element formulation, making use of well-established techniques from structural mechanics like the beam models and equations.

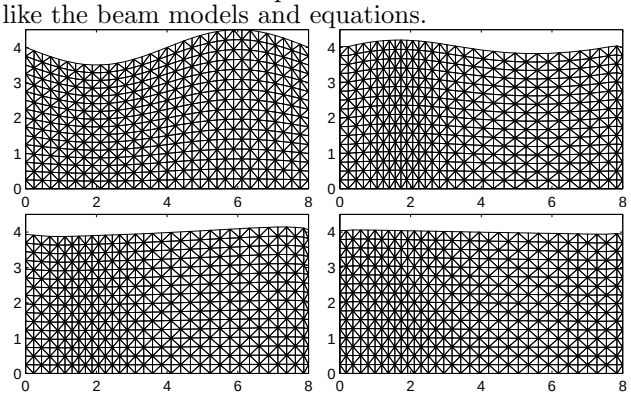

Fig. 5: Snapshots from the time evolution of the gravity wave at the initial condition ($t = 0.0$, upper left), ($t = 1.2$, upper right), $t = 3.6$ (lower left) and $t = 5.6$ (lower right).

4. Test implementation for a gravity wave

To test our implementation, we start with the simulation of an actual gravity wave. First of all, we want to know how well the volume is conserved for different integrators. We compute the initial stationary state by setting the pressure at the upper free surface to zero. The Newton-Iteration for the stationary state yields the hydrostatic pressure, higher pressures below the maximum of the wave, and lower pressure of the minimum. Because we have to make use of the velocities at the intervals which are determined by our BDF2-integration of the internal flow, we have the velocity available only at discrete times, so one-step methods (Runge-Kutta-type methods which use several function evaluations per timestep) cannot be implemented. For the problem in Fig. 5, we tentatively implemented an Euler-Step, as well as a second order Adams-Bashforth method (AB2), which computes the positions at the next timestep $n + 1$from the current positions x_n, the current velocities v_n and the previous velocities v_{n-1} as

$$x_{n+1} = x_n + \tau \left(\frac{3}{2} v_n - \frac{1}{2} v_{n-1} \right). \tag{4}$$

As a tentative parameter of the verisimilitude of the simulation, we take the conservation of the volume. For our gravity wave, the viscosity was chosen as unity (in SI-units), the system size width 8 × (average) height 4 (in units of [m]). We prefer to avoid dimensionless units, because with free surfaces we have not only inertia effects from the flow, but also from the gravity, and in the future, another scale may enter due to the surface tension. For easier comparison with the experiment, we would like to stay with SI-units. The initial pressure profile we obtained with zero pressure on the surface, which lead to a hydrostatic pressure distribution, where the pressure was higher below "hills", and lower under "valleys" of the wave. For the gravity wave in this section, the re-meshing is done is such a way that we we compute the new position of the new mesh points on the surface. The new position for the the new mesh points below the surface is chosen in the horizontal direction vertically below the surface points. The new position in the vertical direction is computed equidistantly along the vertical. As can be seen in Fig. 5, the elements at the left and right boundary are shortened in this process, while the elements which are what was initially the maximum of the wave, are elongated.

As an initial confirmation, we made a test-run with a flat surface and found that in the absence of the motion of a surface, the volume is constant. The result for the volume for $\tau = 0.008$ and a spatial discretization of $N_{elem} = 4760$ elements can be seen in Fig. 6: Obviously, the volume conservation is better for the higher integration order integrator. When the motion of the surfaces slows down, the increase of the error also slows down. Actually, we have up to now discussed only how in eqs. (3) and (4) the old velocities to obtain the new positions, but we have to discuss at which position we evaluate the velocities. Because we have a Eulerian formulation, we should not just use the old positions at different timesteps: As the philosophy of our method is, rather than trace the movement of the lattice points in a Lagrangian way, just to predict the new position of

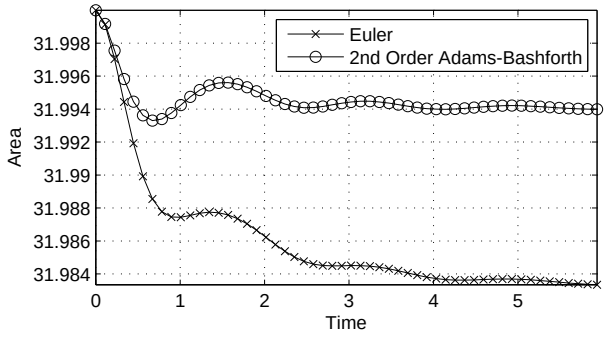

Fig. 6: Time evolution of the volume for the integration with Euler (×) and Adams Bashforth of second order (○) for $\tau = 0.008$ and 4760 grid-points. Symbols are not drawn for every timestep to keep the figure readable.

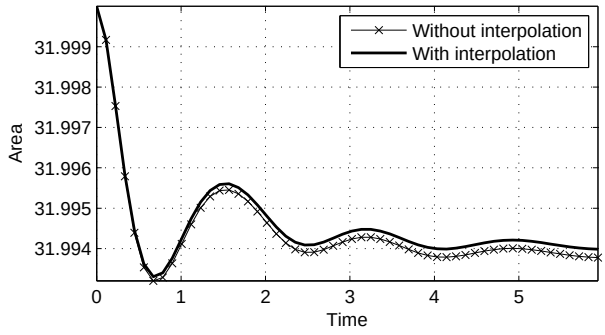

Fig. 7: Time evolution of the volume for the second order Adams Bashforth method with (-) and without (×) interpolation for stepsize of $\tau = 0.008$ and $N_{elem} = 4760$ elements.

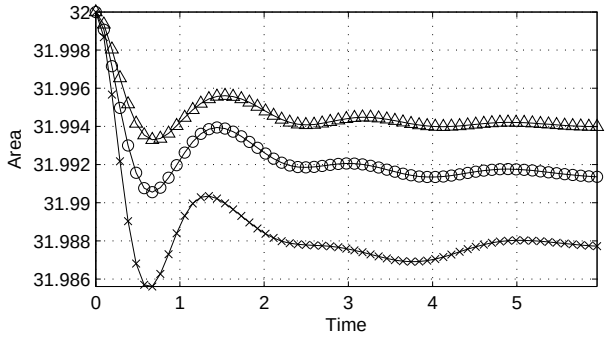

Fig. 8: Time evolution of the volume with AB2 for the stepsize of $\tau = 0.008$ and different numbers of mesh points $N_{elem} = 1088$ ($\times$), $N_{elem} = 2600$ ($\circ$) and $N_{elem} = 4760$ ($\triangle$).

the surface, we choose for AB2

$$x_{n+1} = x_n + \tau \left(\frac{3}{2} v_n(x_n) - \frac{1}{2} v_{n-1}(x_n) \right), \qquad (5)$$

i.e. the new position of a grid-point of the surface is computed from the current position of the grid-point and the previous flow velocities at the current position. If at step $n-1$ the grid-point was not at x_n yet, we have to obtain it by interpolation. This is easy in our FEM-formulation if x_n is lower than the previous surface height, i.e. if the surface is falling. Unfortunately, if the surface is rising, we would have to extrapolate to points where there was no fluid at all, so we have to abandon the use of eq. (5) and just use the velocities of the grid point at the previous timestep in AB2,

$$x_{n+1} = x_n + \tau \left(\frac{3}{2} v_n(x_n) - \frac{1}{2} v_{n-1}(x_{n-1}). \right). \qquad (6)$$

That our argument about the interpolation is justified can be seen from Fig. 7: The error in the volume conservation is larger without interpolation, though not significantly. So in the following, we will use interpolated velocities if the surface moves downward. If the surface moves upward, we would need extrapolation, but we have to decided to do without, as neither is extrapolation an exact science, nor, with respect to the volume conservation for interpolation, can be expect significant improvements.

Let us next study the effect of the mesh size for the conservation of the volume. For the same stepsize of $\tau = 0.008$, we have investigated meshes with 1088, 2600 and 4760 mesh-points. As can be seen in Fig. 8, the

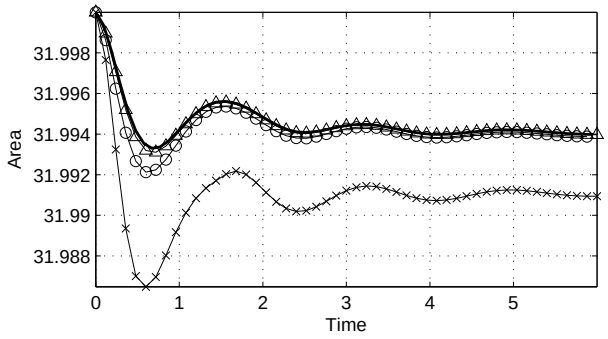

Fig. 9: Time evolution of the volume for 4760 elements and various timesteps for integration with AB2 for (from below) $\tau = 0.06$ ($\times$), $\tau = 0.03$ ($\circ$), $\tau = 0.015$ ($\triangle$) and $\tau = 0.008$ ($-$).

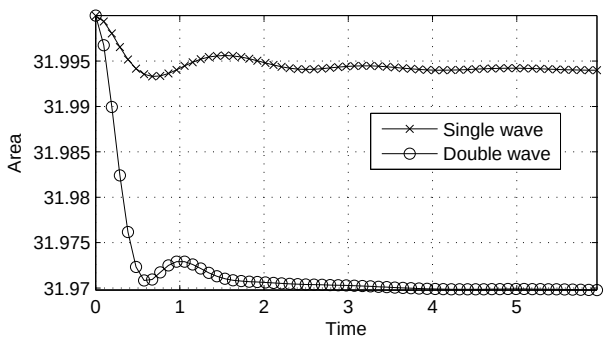

Fig. 10: Time evolution of the volume for $N_{elem} = 4760$ and stepsize $\tau = 0.008$ for integration with AB2 for a single ($\times$) and double ($\bigcirc$) wave.

accuracy of the volume conservation becomes (as is to be expected) better with decreasing mesh size, i.e. increasing number of grids. In the next step, we want to verify the effect of the timestep for a system with 1088 elements. As can be seen in Fig. 9, the improvement from $\tau = 0.06$ to $\tau = 0.008$ is monotonous. We have to conclude that the remaining error (the deviation from 32) is rather due to the lattice discretization than due to the time discretization, especially if we compare with the result for 4760 grid points in Fig. 8.

According to Panton[12] (p. 536), the flow of a gravity wave must be irrotational. By inspection of the flow field, we could verify that for various viscosities this was the case. It would it is now interesting how our algorithm behaves if we have more wave maxima. For a system with the same amplitude, but half the wave length with our highest resolution of $N_{elem} = 4760$ and a stepsize of $\tau = 0.008$, we can see (Fig. 10) that the error is significantly larger than for a single wave. In improvement in the accuracy will be only possible with a smaller mesh-size. Nevertheless, for the time being, let us stick with the error in the conservation without any additional tricks like rescaling of elements etc, because the error in the volume may have its uses, as a large error may indicate deficiencies in the spatial discretization. The number of Newton-Raphson iteration varies between four and two iterations, it depends on the flow velocities, but not on the resolution of the grid. Actually, the source of the error in the volume conservation seems to be largely due to the behavior at the solid boundary: Fig. 11 shows that the out-most elements on the surface have a rather weird angle, compared to the rest of the surface, which is rather smooth.

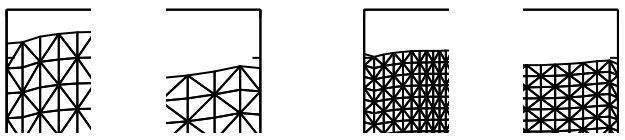

Fig. 11: Zoom into the corners of the the snapshots from the time evolution of the gravity wave with one (left) and two maxima (right) at $t = 1.6$.

5. Simulation of the collapse of a water column

Now we can turn to more interesting problems for the sake of verifying the algorithm. A popular testcase, especially for Lagrangian (particle-based) methods is the time evolution of a water-"step". While boundary conditions are not a great concern for particle methods, for our FEM-method we have to choose the "right" boundary conditions: At every node on the surface, the pressures are set to zero, at every node which does not touch a wall, the velocities are computed, and at every node which touches a wall, the normal velocities

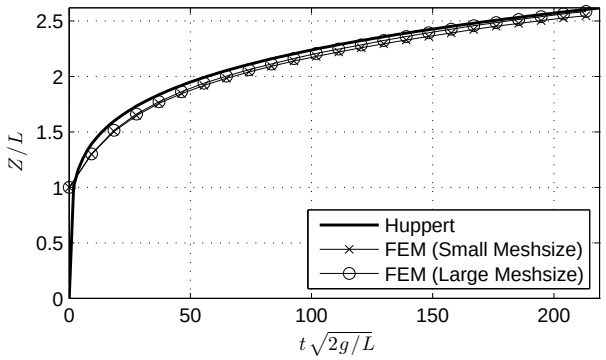

Fig. 12: Comparison of time evolution of the advancement of a wave front under the lubrication approximation with our simulation with large (circles, 400 triangles) and fine (crosses, 3000 triangles) grid for $1/\nu = 10$ and stepsize $\tau = 0.01$.

are set to zero, and the tangential velocities are computed. (Choosing the pressure hydrostatic on the vertical face of the step would inhibit any evolution, as the pressure field would be the same as in the presence of a wall). The next issue concerns the choice of the grid: Because the deformations are not small, and cannot only be taken into account by rescaling the grid in the vertical direction, we have to re-mesh the grid globally. For this, we use the algorithm which we originally developed for grains in fluid [13]. As it was initially written for convex domains, some modifications have become necessary.

Huppert[14] derived the time dependence the advancement of the front Z with time t as

$$Z(t) = 1.411 \left(\frac{gq^3}{3\nu} \right)^{1/5} t^{1/5} \qquad (7)$$

for an initial area q, gravity g and viscosity ν under the lubrication approximation, i.e. under the assumption that effects due to surface tension etc. could be neglected and the height h of the water column is much smaller than its width l[15]. Thought the latter assumption is rather problematic in our case, our data in Fig 12 compare rather well with eq. (7) at $1/\nu = 10$. Changes of ν over a reasonable range of parameters did not lead to any changes in the curve. We leave it to the reader to interpret the good correspondence as either a confirmation of our simulation method or the lubrication approximation. As the agreement with the lubrication approximation becomes better for larger mesh sizes (circles in Fig 12) , but the error in the volume in Fig 13 increases up to about 1% for the find grid, and 1.5% for the rough mesh, it is worth having a look at the actual wave fronts. We see in Fig. 14 that at $t = 8.5$[s] the

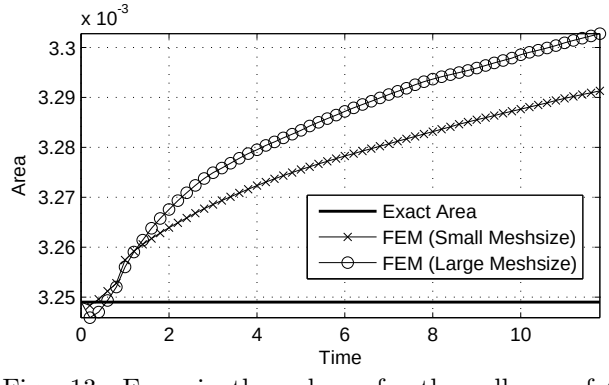

Fig. 13: Error in the volume for the collapse of the squre water column at $1/\nu = 10$ and stepsize $\tau = 0.01$.

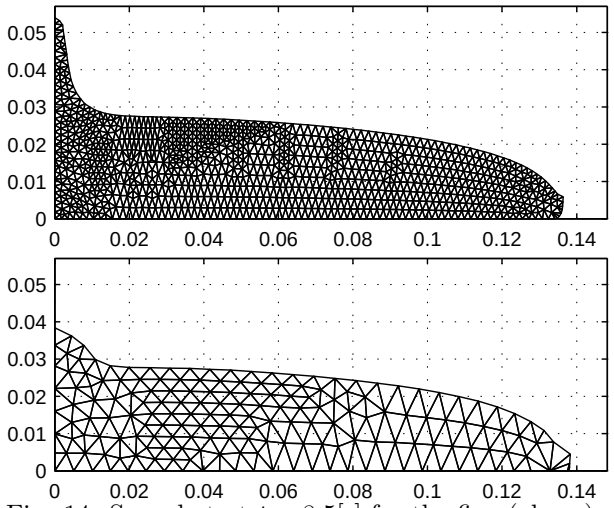

Fig. 14: Snapshot at $t = 8.5$[s] for the fine (above) and rough mesh (below) for the collapse of the squre water column at $1/\nu = 10$ and stepsize $\tau = 0.01$.

wavefront looks rather ragged both for fine and rough meshes. For this, our meshing algorithm responsible which adds new nodes if the distance between nodes becomes larger than 1.5 the choosen characteristic length for the nodes. It seems that in the next version of the remeshing algorithm, we also have to take care of the structure of the flow field. Near the left wall, one can see that when we supply a good resolution, actually a thin film develops (Fig. 14, above), while for too rough grids, (Fig. 14, below), the film is actually cut off, and only a meniscus remains near the wall. This shape of the fluid surface is purely an effect of the viscosity, as we have not yet modeled cohesive or adhesive forces in the simulation. On the one hand, it is conforting to see that we can run our algorithm both with physically small resolution (to capture even thin films), or with rough resolution if we are only interested in a more large scale dynamics. Certainly is not possible for many algorithms to be run beyond multiples of the actual resolution, so we consider our implementation as a success (due, probably, to the stiff BDF2-integrator for the flow field). On the other hand, we have no error indicator if we are actually above the resolution scale of physical phenomena, so no automatic error detection for the spatial resolution is possible (for the time step, the Newton-iteration does not converge if the timestep is choosen too large.)

For higher Reynolds numbers, we want to compare our simulation with experimental data rather than with other simulations or theories. Especially with the experiment by Martin and Moyce[16] is of interest, as these authors argue that for their dimensions of the vessels, air resistance and surface tension can be neglected, exactly the conditions which we have in our simulation.

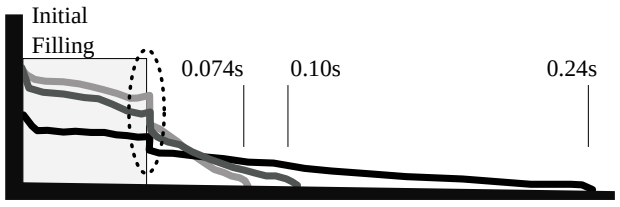

Fig. 15: Advancing of the fluid front in the experiment of Martin and Moyce[16] for the collapse of a fluid column of 57×57 mm. The line-width is approximately the width of the shadows in the original frames, graphics is mirrored compare to Martin and Moyce, because for the simulation it is more convenient to have the origin on the left.

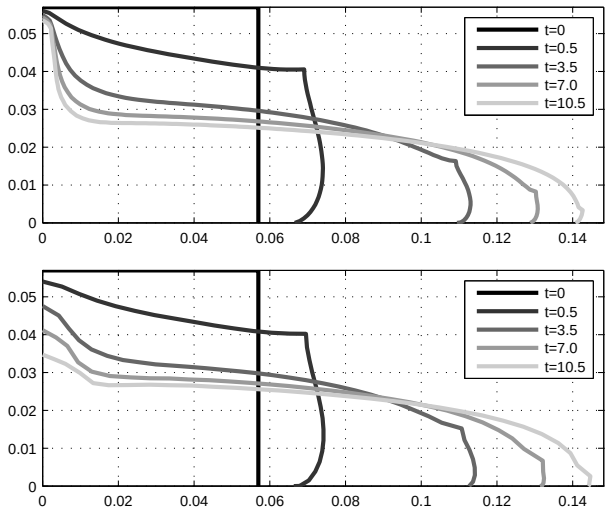

Fig. 16: Time evolution for the fronts in for for the fine (above) and rough mesh (below) from Fig. 14 for the collapse of the squre water column at $1/\nu = 10$ and stepsize $\tau = 0.01$. The corner travels towards the front.

We have extracted the outlines of the graphics and associated them with the corresponding with the time data data in Fig. 15. In the following, we focus on the shape for short time scales, which we consider a more meaningful for the verisimilitude of the simulation than the long-term time evolution, in which geometric details are smeared out and momentum-conservation and energy-decay dominate.

What is striking in the high-speed pictures (300 frames/s) of Martin and Moyce[16] is, that the initial sharp step-shape between initial water level and vertical boundary (dotted oval in Fig. 15) is, within the limits set by the shadows in the figures, rather well preserved during the collapse. Moreover, the upper surfaces looses convexity both on the left and on the right side. The resilience of the "corner" on the right is rather surprising, not to say counter-intuitive, but as the pressures in upper right corner, are negligible, there are no forces which could cause the decay of the angle. For low inverse viscosity $1/\nu = 10$ in Fig. 16, the cusp from the initial square profils traveled fast towards the front. Within the deadline for this preprint, we were not able to run the simulation at the necessary inverse viscosity (about $1/\nu = 10000$), but for the largest inverse viscosities we could reach ($1/\nu = 500$), the cusp indeed stayed where it was, see Fig. 17 As we have no surface tension implemented, the cusp becomes sharper and sharper, as in the case of triangular ocean waves above a shear flow, which also steepen and become sharper until they break.

In the particle-simulations by Koshizuka et al.[17], the upper surface of the column is convex, and their experimental data are not clear enough to determine the shape of the column. In our simulation, where we start with a singular surface (the gradient has a jump at the right corner), it turns out that the singularity is preserved. Though there is a certain uncertainty, as we have to investigate the effects of the grid discretization in more detail, if we assume that our simulation gives the correct solution, we have to conclude that such a singularity in the surface does not vanish on hydrodynamical grounds, but due to surface tension, which is not yet implemented in our code.

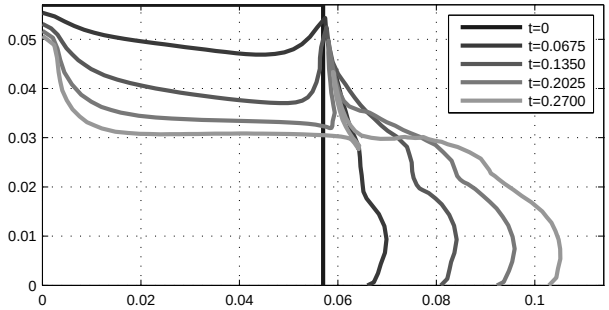

Fig. 17: Time evolution for the fronts in for for the fine (above) and rough mesh (below) from Fig. 14 for the collapse of the squre water column at $1/\nu = 10$ and stepsize $\tau = 0.01$. The corner travels towards the front.

6. Summary and Conclusions

We have implemented a Finite-Element method with free surfaces where we integrate out the motion of the surface elements according to the velocity data obtained from the FEM-method on the surface. Compared to conventional efforts, which try to solve partial differential equations for the motion of the surface, the additional effort in our method with respect to new data structures etc. is negligible. The method shows structures (development of a non-convex horizontal surface) which is lacking in particle-based simulations for the same problem. This gives us hope that the method can be extended (with better re-meshing algorithms and the inclusion of surface tension) so that it may yield a higher degree of realism (not to say precision) than other methods which rely on unphysical surface- or particle dynamics. Certainly, the results are affected by remeshing and mesh refinement, and certainly we could identify a problem in the remeshing of rapidly advancing fronts. Beyond that, we want to implement surface tension, modeled by beam equations for which the curvature is evaluated at the fluid surface, as well as better gridding algorithms with a finer mesh towards the surface than in the bulk of the fluid.

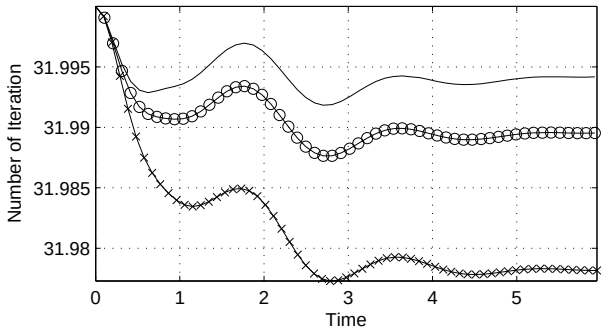

Fig. 18: Time evolution of the volume for the integration with Euler (×), Adams Bashforth of second (○) and third (full line) order for $\tau = 0.008$. Symbols are not drawn for every timestep to keep the figure readable.

Appendix: Other Integrators

Beyond AB2, we have implemented a variant of the third order Adams-Bashforth method (AB3)

$$x_{n+1} = x_n + \tau \left(\frac{23}{12} v_n - \frac{4}{3} v_{n-1} + \frac{5}{12} v_{n-2} \right). \qquad (8)$$

Using at the nth timestep the velocity v_{n-2} from two timesteps earlier in eq. (8) may not contribute to the improvement of the accuracy in the case of strong movement of the surface. We therefore replaced v_{n-2} by using the centered-difference formula of the acceleration

$$a_{n+1} = \frac{v_n - v_{n-2}}{2\tau}, \qquad (9)$$

which is available as the "right hand side" from eq. (2) so that

$$v_{n-2} = v_n - 2\tau a_{n+1}. \tag{10}$$

In that case, our variant of the AB3-method looks like

$$x_{n+1} = x_n + \tau\left(\frac{7}{3}v_n - \frac{4}{3}v_{n-1} - \frac{5}{6}a_{n-1}\right). \tag{11}$$

In analogy to eq. (6), we used $a_{n-1}(x_n)$ if interpolation was possible, and $a_{n-1}(x_{n-1})$ where extrapolation would have become necessary. Initially, the result looked promising, as can be seen in Fig. 18: The volume conservation was better than for the first and second order method.

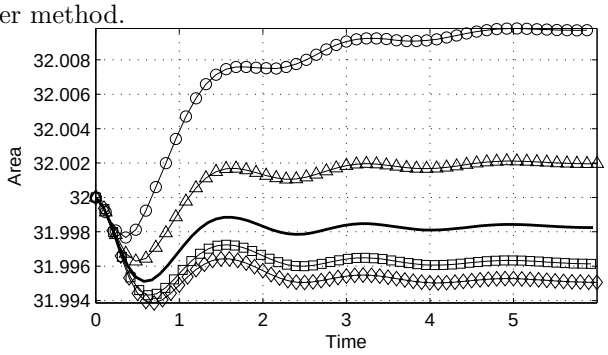

Fig. 19: Time evolution of the volume for 4760 elements and various timesteps for integration with AB3 for $\tau = 0.03$ ($\circ$) $\tau = 0.015$ ($\triangle$) $\tau = 0.008$ ($-$) $\tau = 0.004$ ($\square$) and $\tau = 0.002$ ($\diamond$).

Unfortunately, when we decreased the stepsize, it turned out that the algorithm did not converge in third order, but only linearly, as can be seen in Fig. 19. In principle, numerical algorithms may not converge when decreasing the timestep due to the fact that a decrease in timestep, while decreases the discretization error, but at the same time increases the rounding error (see. Fig. 20). Nevertheless, in our case, that cannot be the reason, as AB2 converges without problems with the same timestep.

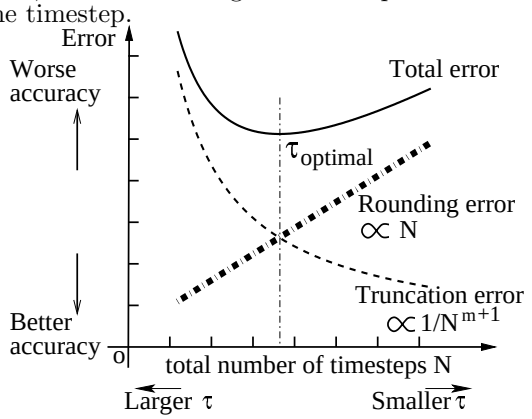

Fig. 20: Schematic representation of the numerical error as a sum of the truncation error and the discretization error: The smallest timestep does not give the most accurate result, but there is an optimal timestep.

The introduction of the acceleration a_{n-1} in eq. (10) instead of v_{n-2} is not the reason for the loss of accuracy either: For a time integration of $x = -\ddot{x}$ with "exact" input-data, eq. (11) lead to a smaller error than with eq. (11). In In fact, we had overlooked the fact that for n-th order integration with Adams-Bashforth, all variables which enter in the formula must be obtained to n-th order. If the v_i, a_i used in eqs. (8,11) are of lower than third order, so will be the results for the x_i. As our velocities and acceleration are only computed up to second order with our BDF2-integration in eq. (2), we have to conclude that the third order formula is meaningless and does not lead to improved accuracy. We could confirm using manipulated output from the BDF2-integration: When we intentionally introduced an error there, the AB3-integration was much more strongly affected than the AB2-integration.

When we were already about it, we decided to try out other methods: The midpoint-rule

$$x_{n+1} = x_n + v_n\tau + a_n\tau^2, \tag{12}$$

named after the fact that it can be derived from centered differences which can be derived from eq. (10) . Further, we wanted to try out the Adams-Moulton-formula in third order

$$x_{n+1} = x_n + \tau\frac{1}{12}\left(5v_{n+1} + 8v_n - v_{n-1}\right). \tag{13}$$

Because using implicit integrators with the "new" velocities v_{n+1} makes the re-computation of the whole BDF2-step necessary, we tried to avoid this by modifying the Adams-Moulton formula in eq. (13) to an "explicit formula" using the centered differences in eq. (10) for timestep n so that

$$x_{n+1} = x_n + \tau\frac{1}{12}\left(8v_n + 10\tau a_n + 4v_{n-1}\right). \tag{14}$$

In Fig. 21, we have contrasted the results for different methods: The only method which gives more accurate results than AB2 for the timestep used is AB3, and that only because it is not yet fully converged: With smaller timestep (necessary for larger velocities and/or smaller viscosities), its error will become larger than in Fig. 21, so for our input data (computed with BDF2), Adams-Bashforth in second order gives the most satisfying results.

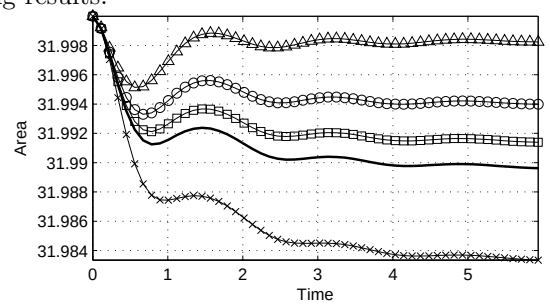

Fig. 21: Numerical error in the conservation of the volume for several methods 4760 mesh points and timestep $\tau = 0.008$: Euler ($\times$), Midpoint ($\square$), 2nd Order Adams-Bashforth ($\circ$), 3rd Order Adams-Bashforth ($\triangle$) and 3rd Order Adams-Moulton ($-$).

References

1. The MathWorks, Inc. (2009), MATLAB Version 7.9.0.529 (R2009b).

2. Lanczos, C., The Variational Principles of Mechanics, 4th Edition (Dover Publications, 1986).

3. Hairer, E. and Wanner, G., Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, 2nd Edition (Springer-Verlag, 1996).

4. Gresho, P. M. and Sani, R. L., Incompressible Flow and the Finite Element Method Volume 2: Isothermal Laminar Flow (John Wiley and Sons, Ltd., 2005), pp. 805–806.

5. Browne, L. W. B., The Marker and Cell Technique, in Numerical Simulation of Fluid Motion, ed. Noye, J. (North-Holland Publishing Company, 1978), pp. 223–247.

6. Harlow, F. H. and Welch, J. E., "Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface", *The Physics of Fluids*, 8(12), (1965), pp. 2182–2189.

7. Osher, S. and Fedkiw, R., Level Set Methods and Dynamic Implicit Surfaces (Springer, 2003).

8. Hirt, C. W. and Nichols, B. D., "Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries", *Journal of Computational Physics*, 39(1), (1981), pp. 201–225.

9. Caboussat, A., Maronnier, V., Picasso, M., and Rappaz, J., Numerical Simulation of Three Dimensional Free Surface Flows with Bubbles, in Lecture Notes in Computational Science and Engineering, Vol. 35 (Springer-Verlag, 2003), pp. 69–86.

10. Sakai, Y. (2007), "Dynamic Remeshing for Finite Element Method (in Japanese)", Undergraduate's thesis, The University of Electro-Communications, Department of Mechanical Engineering and Intelligent Systems.

11. Engeln-Müllges, G. and Uhlig, F., Numerical Algorithms with Fortran (Springer, 1996).

12. Panton, R. L., Incompressible Flow Second Edition (John Wiley & Sons, Inc., 1996).

13. Ng, S. H. (2009), "Generation of triangular disordered Grids for Fluid Dynamics Simulation using the Finite Element Method", Undergraduate's thesis, The University of Electro-Communications, Department of Mechanical Engineering and Intelligent Systems.

14. Huppert, H. E., "The propagation of two-dimensional and axisymmetric viscous gravity currents over a rigid horizontal surface", *Journal of Fluid Mechanics*, 121, (1982), pp. 43–58.

15. Gratton, J., Mahajan, S. M., and Minotti, F., Non Newtonian Gravity Creeping Flow (Trieste: International Centre for Theoretical Physics, 1988), pp. 1–18.

16. Martin, J. C. and Moyce, W. J., "Part IV. An Experimental Study of the Collapse of Liquid Columns on a Rigid Horizontal Plane", *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 244(882), (1952), pp. 312–324.

17. Koshizuka, S. and Oka, Y., "Moving-Particle Semi-Implicit Method for Fragmentation of Incompressible Fluid", *Nuclear Science and Engineering*, 123, (1996), pp. 421–434.