ON THE SIMULATION OF FLOWS WITH VIOLENT FREE SURFACE MOTION

Rainald Löhner\(^*\), Chi Yang\(^†\), and Eugenio Oñate\(^‡\)

*School of Computational Sciences, George Mason University, MS 4C7, Fairfax, VA 22030-4444, USA

‡International Center for Numerical Methods in Engineering, Universidad Politécnica de Catalunya, Barcelona, Spain

A Volume of Fluid (VOF) technique has been developed and coupled with an incompressible Euler/Navier Stokes solver operating on adaptive, unstructured grids to simulate the interactions of extreme waves and three-dimensional structures. The present implementation follows the classic VOF implementation for the liquid-gas system, considering only the liquid phase. Extrapolation algorithms are used to obtain velocities and pressure in the gas region near the free surface. The VOF technique is validated against the classic dam-break problem, as well as series of 2-D sloshing experiments and results from SPH calculations. These and a series of other examples demonstrate that the present CFD method is capable of simulating violent free surface flows with strong nonlinear behaviour.

I. INTRODUCTION

Flows with violent free surface motion occur in many engineering applications. Sloshing can be a major problem for fuel and water tanks. Any naval design has to take into consideration high sea states and wave breaking to predict slamming, green water on deck and sloshing (e.g. in liquid natural gas (LNG) tankers). Coastal and offshore engineering has to take into consideration high sea states and wave breaking for the design and outlay of offshore platforms, piers, harbours and near-shore residential developments.

The computation of highly nonlinear free surface flows is difficult because neither the shape nor the position of the interface between air and water is known a priori; on the contrary, it often involves unsteady fragmentation and merging processes. There are basically two approaches to compute flows with free surface: interface-tracking and interface-capturing methods. The former computes the liquid flow only, using a numerical grid that adapts itself to the shape and position of the free surface. The free surface is represented and tracked explicitly either by marking it with special marker points, or by attaching it to a mesh surface. Various surface fitting methods for attaching the interface to a mesh surface were developed during the past decades using the finite element method. In the interface tracking methods, the free surface is treated as a boundary of the computational domain, where the kinematic and dynamic boundary conditions are applied. These methods can not be used if the interface topology changes significantly, as is contemplated here for overturning or breaking waves. The second possible approach is given by the so-called interface-capturing
methods\textsuperscript{15, 26, 59, 57, 51, 60, 49, 8, 17, 5, 16, 28, 11}. These consider both fluids as a single effective fluid with variable properties; the interface is captured as a region of sudden change in fluid properties. The main problem of complex free surface flows is that the density $\rho$ jumps by three orders of magnitude between the gaseous and liquid phase. Moreover, this surface can move, bend and reconnect in arbitrary ways. The difficulties that can arise if one treats either the complete system or just the liquid phase can be illustrated on two small examples. The first one considers hydrostatic flow, where the exact solution is $\mathbf{v} = 0, p = -\mathbf{g} \cdot (\mathbf{x} - \mathbf{x}_0)$, where $\mathbf{x}_0$ denotes the position of the free surface (see Figure 1). Unless the free surface coincides with the faces of elements, there is no way for typical finite element shape functions to capture the discontinuity in the gradient of the pressure. This implies that one has to either increase the number of Gauss-points\textsuperscript{10} or modify (e.g., enrich) the shape function space\textsuperscript{11}. Using the standard linear element procedure leads to spurious velocity jumps at the interface, as any small pressure gradient that ‘pollutes over’ from the water to the air region will accelerate the air considerably. This in turn will lead to loss of divergence, causing more spurious pressures. The whole cycle may, in fact, lead to a complete divergence of the solution.

Figures 1,2: Hydrostatic Pressure Distribution and Communicating Tubes

Faced with this dilemma, most flows with free surfaces have been solved neglecting the air. The shortcomings of this approach can be illustrated in the second example, sketched in Figure 2. A tube is filled with a column of fluid which is interrupted by a pocket of air. Neglecting the air would never allow the higher column to push the lower column up. In the present case, we have followed this approach, fully aware of the limitations. The remainder of the paper is organized as follows: Section 2 summarizes the basic elements of the present incompressible flow solver; Sections 3,4 describe the temporal and spatial discretization; Section 5 describes the volume of fluid extensions; some examples are shown in Section 6; finally, some conclusions are given in Section 7.

II. BASIC ELEMENTS OF THE SOLVER

In order to fix the notation, the equations describing incompressible, Newtonian flows in an arbitrary Lagrangian Eulerian (ALE) frame are written as

$$\rho \mathbf{v}_{i+1} + \rho \mathbf{v}_a \nabla \mathbf{v} + \nabla p = \nabla \mu \nabla \mathbf{v} + \rho \mathbf{g} ,$$  \hspace{1cm} (1)

$$\nabla \cdot \mathbf{v} = 0 .$$ \hspace{1cm} (2)

Here $\rho$ denotes the density, $\mathbf{v}$ the velocity vector, $p$ the pressure, $\mu$ the viscosity and $\mathbf{g}$ the gravity vector. The advective velocity if given by $\mathbf{v}_a = \mathbf{v} - \mathbf{w}$, where $\mathbf{w}$ is the mesh velocity. We remark that both the gaseous and liquid phases are considered incompressible, thus Eqn.(2). The liquid-gas interface is described by a scalar equation of the form:
\[ \Phi_{t} + \mathbf{v} \cdot \nabla \Phi = 0 . \]  

(3)

For the classic VOF technique, \( \Phi \) represents the total density of the material in a cell/element or control volume\(^{45, 26, 57, 51, 49, 8, 17, 5, 28} \). For pseudo-concentration techniques, \( \Phi \) represents the percentage of liquid in a cell/element or control volume. For the level set (LS) approach \( \Phi \) represents the signed distance to the interface\(^{51, 16} \). For a combinations of these approaches, see Sussman and Puckett\(^{52} \).

Since over a decade\(^{37, 43, 48, 40} \) the numerical schemes chosen to solve the incompressible Navier-Stokes equations given by Eqns.(1,2) have been based on the following criteria:

- Spatial discretization using **unstructured grids** (in order to allow for arbitrary geometries and adaptive refinement);
- Spatial approximation of unknowns with **simple finite elements** (in order to have a simple input/output and code structure);
- Temporal approximation using **implicit integration of viscous terms and pressure** (the interesting scales are the ones associated with advection);
- Temporal approximation using **explicit integration of advective terms**;
- **Low-storage, iterative solvers** for the resulting systems of equations (in order to solve large 3-D problems); and
- Steady results that are **independent from the timestep** chosen (in order to have confidence in convergence studies).

### III. TEMPORAL DISCRETIZATION

For most of the applications listed above, the important physical phenomena propagate with the **advective** timescales. We will therefore assume that the advective terms require an explicit time integration. Diffusive phenomena typically occur at a much faster rate, and can/should therefore be integrated implicitly. Given that the pressure establishes itself immediately through the pressure-Poisson equation, an implicit integration of pressure is also required. The hyperbolic character of the advection operator and the elliptic character of the pressure-Poisson equation have led to a number of so-called projection schemes. The key idea is to predict first a velocity field from the current flow variables without taking the divergence constraint into account. In a second step, the divergence constraint is enforced by solving a pressure-Poisson equation. The velocity increment can therefore be separated into an advective-diffusive and pressure increment:

\[ \mathbf{v}^{n+1} = \mathbf{v}^{n} + \Delta \mathbf{v}^{n} + \Delta \mathbf{v}^{p} = \mathbf{v}^{*} + \Delta \mathbf{v}^{p} . \]  

(4)

For an explicit (forward Euler) integration of the advective terms, with implicit integration of the viscous terms, one complete timestep is given by:

- **Advective-Diffusive Prediction**: \( \mathbf{v}^{n} \rightarrow \mathbf{v}^{*} \)

\[ \left[ \frac{\rho}{\Delta t} - \partial \nabla \mu \nabla \right] \left( \mathbf{v}^{*} - \mathbf{v}^{n} \right) + \mathbf{v}_{a}^{n} \cdot \nabla \mathbf{v}^{n} + \nabla p^{n} = \nabla \mu \nabla \mathbf{v}^{n} + \rho \mathbf{g} ; \]

(5)

- **Pressure Correction**: \( p^{n} \rightarrow p^{n+1} \)

\[ \nabla \cdot \mathbf{v}^{n+1} = 0 ; \]  

(6)

\[ \rho \frac{\mathbf{v}^{n+1} - \mathbf{v}^{*}}{\Delta t} + \nabla \left( p^{n+1} - p^{n} \right) = 0 ; \]  

(7)
which results in

\[ \nabla \cdot \frac{1}{\rho} \nabla (p^{n+1} - p^n) = \frac{\nabla \cdot \mathbf{v}^n}{\Delta t} ; \]

- **Velocity Correction:** \( \mathbf{v}^* \to \mathbf{v}^{n+1} \)

\[ \mathbf{v}^{n+1} = \mathbf{v}^* - \frac{\Delta t}{\rho} \nabla (p^{n+1} - p^n) . \] (9)

At steady state, \( \mathbf{v}^* = \mathbf{v}^n = \mathbf{v}^{n+1} \) and the residuals of the pressure correction vanish, implying that the result does not depend on the timestep \( \Delta t \). \( \theta \) denotes the implicitness-factor for the viscous terms (\( \theta = 1 \): 1st order, fully implicit, \( \theta = 0.5 \): 2nd order, Crank-Nicholson). One can replace the one-step explicit advective-diffusive predictor by a multistage Runge-Kutta scheme\(^2\), allowing for higher accuracy in the advection-dominated regions and larger timesteps without a noticeable increment in CPU cost.

A \( k \)-step, time-accurate Runge-Kutta scheme or order \( k \) for the advective parts may be written as:

\[ \rho \mathbf{v}^i = \rho \mathbf{v}^n + \alpha^i \gamma \Delta t \left( -\rho \mathbf{v}^{i-1}_a \cdot \nabla \mathbf{v}^{i-1} - \nabla p^n + \nabla \mu \nabla \mathbf{v}^{i-1} \right) ; \quad i = 1, k - 1 ; \] (10)

\[ \frac{\rho}{\Delta t} - \theta \nabla \mu \nabla \left( \mathbf{v}^k - \mathbf{v}^n \right) + \rho \mathbf{v}^{k-1}_a \cdot \nabla \mathbf{v}^{k-1} + \nabla p^n = \nabla \mu \nabla \mathbf{v}^{k-1} . \] (11)

Here, the \( \alpha^i \) are the standard Runge-Kutta coefficients \( \alpha^i = 1/(k + 1 - i) \). As compared to the original scheme given by Eqn.(5), the \( k - 1 \) stages of Eqn.(10) may be seen as a predictor (or replacement) of \( \mathbf{v}^n \) by \( \mathbf{v}^{k-1} \). The original right-hand side has not been modified, so that at steady-state \( \mathbf{v}^n = \mathbf{v}^{k-1} \), preserving the requirement that the steady-state be independent of the timestep \( \Delta t \). The factor \( \gamma \) denotes the local ratio of the stability limit for explicit time-stepping for the viscous terms versus the timestep chosen. Given that the advective and viscous timestep limits are proportional to:

\[ \Delta t_a \approx \frac{h}{|V|} ; \quad \Delta t_v \approx \frac{\rho h^2}{\mu} , \] (12)

we immediately obtain

\[ \gamma = \frac{\Delta t_v}{\Delta t_a} \approx \frac{\rho |V|h}{\mu} \approx Re_h , \] (13)

or, in its final form:

\[ \gamma = \min(1, Re_h) . \] (14)

In regions away from boundary layers, this factor is \( O(1) \), implying that a high-order Runge-Kutta scheme is recovered. Conversely, for regions where \( Re_h = O(0) \), the scheme reverts back to the original one (Eqn.(5)). Projection schemes of this kind (explicit advection with a variety of schemes, implicit diffusion, pressure-Poisson equation for either the pressure or pressure increments) have been widely used in conjunction with spatial discretizations based on finite differences\(^{33,2,3,1} \), finite volumes\(^{29} \), and finite elements\(^{21,12,23,37,43,48,40,53,13,130,9,36,31,42,7} \).

One **complete timestep** is then comprised of the following substeps:

- Predict velocity (advective-diffusive predictor, Eqns.(5,10,11));
- Extrapolate the pressure (imposition of boundary conditions);
- Update the pressure (Eqn.(8));
- Correct the velocity field (Eqn.(9));
- Extrapolate the velocity field; and
- Update the scalar interface indicator.

IV. SPATIAL DISCRETIZATION

As stated before, we desire a spatial discretization with unstructured grids in order to:
- Approximate arbitrary domains, and
- Perform adaptive refinement in a straightforward manner, i.e.

without changes to the solver.
From a numerical point of view, the difficulties in solving Eqns. (1-3) are the usual ones. First-order derivatives are problematic (overshoots, oscillations, instabilities), while second-order derivatives can be discretized by a straightforward Galerkin approximation. We will first treat the advection operator and then proceed to the divergence operator. Given that for tetrahedral grids solvers based on edge data structures incur a much lower indirect addressing and CPU overhead than those based on element data structures, only these will be considered.

4.1 The Advection Operator

It is well known that a straightforward Galerkin approximation of the advection terms will lead to an unstable scheme (recall that on a 1-D mesh of elements with constant size, the Galerkin approximation is simply a central difference scheme). Three ways have emerged to modify (or stabilize) the Galerkin discretization of the advection terms:
- Integration along characteristics,
- Taylor-Galerkin (or streamline diffusion); and
- Edge-based upwinding.

Of these, we only consider the third option here. The Galerkin approximation for the advection terms yields a right-hand side (RHS) of the form:

\[ r^i = D^{ij} F_{ij} = D^{ij} (f_i + f_j), \]

where the \( f_i \) are the ‘fluxes along edges’

\[ f_i = S_k^{ij} F_k^i, \quad S_k^{ij} = \frac{d_k^{ij}}{D^{ij}}, \quad D^{ij} = \sqrt{d_k^{ij} d_k^{ij}}, \]

\[ F_{ij} = f_i + f_j, \quad f_i = (S_k^{ij} v_k^i) v_i, \quad f_j = (S_k^{ij} v_k^j) v_j, \]

and the edge-coefficients are based on the shape-functions \( N^i \) as follows:

\[ d_k^{ij} = \frac{1}{2} \int_\Omega (N_k^i N_j - N_k^j N_i) \, d\Omega. \]

A consistent numerical flux is given by
\[ F_{ij} = f_i + f_j - |v^j| (v_i - v_j) , \quad v^{ij} = \frac{1}{2} S^{ij}_k (v^i_k + v^j_k) . \] (19)

As with all other edge-based upwind fluxes, this first-order scheme can be improved by reducing the difference \( v_i - v_j \) through (limited) extrapolation to the edge center\(^{41}\). The same scheme is used for the transport equation that describes the propagation of the VOF fraction, pseudo-concentration or distance to the free surface given by Eqn.(3).

4.2 The Divergence Operator
A persistent difficulty with incompressible flow solvers has been the derivation of a stable scheme for the divergence constraint (2). The stability criterion for the divergence constraint is also known as the Ladyzenskaya-Babuska-Brezzi or LBB condition\(^{24}\). The classic way to satisfy the LBB condition has been to use different functional spaces for the velocity and pressure discretization\(^{18}\). Typically, the velocity space has to be richer, containing more degrees of freedom than the pressure space. Elements belonging to this class are the p1/p1+bubble mini-element\(^{50}\), the p1/iso-p1 element\(^{56}\), and the p1/p2 element\(^{64}\). An alternative way to satisfy the LBB condition is through the use of artificial viscosities\(^{37}\), ‘stabilization’\(^{19,55,20}\) or a ‘consistent numerical flux’ (more elegant terms for the same thing). The equivalency of these approaches has been repeatedly demonstrated\(^{50,37,41}\). The approach taken here is based on consistent numerical fluxes, as it fits naturally into the edge-based framework. For the divergence constraint, the Galerkin approximation along edge \( i,j \) is given by

\[ F_{ij} = f_i + f_j , \quad f_i = S^{ij}_k v^i_k , \quad f_j = S^{ij}_k v^j_k . \] (20)

A consistent numerical flux may be constructed by adding pressure terms of the form:

\[ F_{ij} = f_i + f_j - |\lambda^{ij}| (p_i - p_j) , \] (21)

where the eigenvalue \( \lambda^{ij} \) is given by the ratio of the characteristic advective timestep of the edge \( \Delta t \) and the characteristic advective length of the edge \( l \):

\[ \lambda^{ij} = \frac{\Delta t^{ij}}{l^{ij}} . \] (22)

Higher order schemes can be derived by reconstruction and limiting, or by substituting the first-order differences of the pressure with third-order differences:

\[ F_{ij} = f_i + f_j - |\lambda^{ij}| (p_i - p_j + \frac{l^{ij}}{2} (\nabla p_i + \nabla p_j)) . \] (23)

This results in a stable, low-diffusion, fourth-order damping for the divergence constraint.

V. VOLUME OF FLUID EXTENSIONS

The extension of a solver for the incompressible Navier-Stokes equations to handle free surface flows via the VOF or LS techniques requires a series of extensions which are the subject of the present section. Before going on, we remark that both the VOF and LS approaches were implemented as part of this effort. Experience indicates that both work well. For VOF, it is important to have a monotonicity preserving scheme for \( \Psi \). For LS, it is important to balance the cost and accuracy loss of reinitializations via a vis propagation. Given that the advection solvers used are all monotonicity preserving, and that the VOF option is less CPU-demanding than LS, only the VOF technique is considered in the following.

5.1 Extrapolation of the Pressure
The pressure in the gas region needs to be extrapolated properly in order to obtain the proper velocities in
the region of the free surface. This extrapolation is performed using a three step procedure. In the first step, the pressures for all point in the gas region are set to (constant) values, either the atmospheric pressure or, in the case of bubbles, the pressure of the particular bubble. In a second step, the gradient of the pressure for the points in the liquid that are close to the liquid-gas interface are extrapolated from the points inside the liquid region (see Figure 3). This step is required as the pressure gradient for these points can not be computed properly from the data given. Using this information (i.e. pressure and gradient of pressure), the pressure for the points in the gas that are close to the liquid-gas interface are computed.

5.2 Extrapolation of the Velocity
The velocity in the gas region needs to be extrapolated properly in order to propagate accurately the free surface. This extrapolation is started by initializing all velocities in the gas region to \( v = 0 \). Then, for each subsequent layer of points in the gas region where velocities have not been extrapolated (unknown values), an average of the velocities of the surrounding points with known values is taken (see Figure 4).

5.3 Imposition of Constant Mass
Experience indicates that the amount of liquid mass (as measured by the region where the VOF indicator is larger than a cut-off value) does not remain constant for typical runs. The reasons for this loss or gain of mass are manifold: loss of steepness in the interface region, inexact divergence of the velocity field, boundary velocities, etc. This lack of exact conservation of liquid mass has been reported repeatedly in the literature\(^{51,52,16} \). The recourse taken here is the classic one: add/remove mass in the interface region in order to obtain an exact conservation of mass. At the end of every timestep, the total amount of fluid mass is compared to the expected value. The expected value is determined from the mass at the previous timestep, plus the mass-flux across all boundaries during the timestep. The differences in expected and actual mass are typically very small, so that quick convergence is achieved by simply adding and removing mass appropriately. The amount of mass taken/added is made proportional to the absolute value of the normal velocity of the interface:

\[
\nu_n = \left| \mathbf{v} \cdot \frac{\nabla \Phi}{|\nabla \Phi|} \right| .
\]

In this way the regions with no movement of the interface remain unaffected by the changes made to the interface in order to impose strict conservation of mass.

5.4 Deactivation of Air Region
Given that the air region is not treated/updated, any CPU spent on it may be considered wasted. Most of
the work is spent in loops over the edges (upwind solvers, limiters, gradients, etc.). Given that edges have to be grouped in order to avoid memory contention/allow vectorization when forming right-hand sides\cite{38,39}, this opens a natural way of avoiding unnecessary work: form relatively small edge-groups that still allow for efficient vectorization, and deactivate groups instead of individual edges\cite{41}. In this way, the basic loops over edges do not require any changes. The if-test whether an edge group is active or deactivate occurs outside the inner loops over edges, leaving them unaffected. On scalar processors, edge-groups as small as n\textsubscript{egrp}=8 are used. Furthermore, if points and edges are grouped together in such a way that proximity in memory mirrors spatial proximity, most of the edges in air will not incur any CPU penalty.

5.5 Treatment of Bubbles
The treatment of bubbles follows the classic assumption that the timescales associated with speed of sound in the bubble are much faster than the timescales of the surrounding fluid. This implies that at each instance the pressure in the bubble is (spatially) constant. As long as the bubble is not in contact with the atmospheric air (see Figure 5), the pressure can be obtained from the isentropic relation:

\[
\frac{p_b}{p_0} = \left(\frac{\rho_b}{\rho_0}\right)^\gamma
\]

(25)

where \(p_b, \rho_b\) denote the pressure and density in the bubble and \(p_0, \rho_0\) the reference values (e.g. those at the beginning of the simulation). The gas in the bubble is marked by solving a scalar advection equation of the form given by Eqn.(3):

\[
\frac{\partial b}{\partial t} + \textbf{v}_g \cdot \nabla b = 0
\]

(26)

At the beginning of every timestep the total volume occupied by gas is added. From this volume the density is inferred, and the pressure is computed from Eqn.(25).

![Figure 5: Bubble in Water](image)

At the end of every timestep, a check is performed to see if the bubble has reached contact with the air. Should this be the case, the pressure in the bubble is set to atmospheric pressure. One then typically observes a rather quick collapse of the bubble.

---

8 of 32

American Institute of Aeronautics and Astronautics Paper 2005-0291
VI. EXAMPLES

6.1 Breaking Dam Problem: This is a classic test case for free surface flows.

![Breaking Dam: Problem Definition](image)

Figure 6a: Breaking Dam: Problem Definition

![Breaking Dam: Surface Discretization for the Coarse Mesh](image)

Figure 6b: Breaking Dam: Surface Discretization for the Coarse Mesh
Figures 6c-f: Breaking Dam: Flowfield at Different Times

The problem definition is shown in Figure 6a. This case was run on a coarse mesh with nelem=16,562 elements, a fine mesh with nelem=135,869 and an adaptively refined mesh (where the coarse mesh was the base mesh) with approximately nelem=30,000 elements. The refinement indicator for the latter was the free surface, and the mesh was adapted every 5 time steps. Figure 6b shows the discretization for the coarse mesh, and Figures 6c-f the development of the flowfield and the free surface until the column of water hits the right wall. Note the mesh adaptation in time. The results obtained for the horizontal location of the free surface along the bottom wall are compared to the experimental values of Martin and Moyse\textsuperscript{44}, as well as the numerical results obtained by Hansbo\textsuperscript{25}, Kölke\textsuperscript{34} and Walhorn\textsuperscript{58} in Figure 6g. The dimensionless time and displacement are given by $\tau = t\sqrt{2g/a}$ and $\delta = x/a$, where $a$ is the initial width of the water column. As one can see, the agreement is very good, even for the coarse mesh. The difference between the adaptively refined mesh and the fine mesh was almost indistinguishable, and therefore only the results for the fine mesh are shown in the graph.
Figure 6g: Breaking Dam: Horizontal Displacement

Figure 7a: 3D Dam-Break Wave With Circular Cylinder: Problem Definition
Figure 7b: Free Surface Wave Elevation
6.2 3D Dam-Break Wave Interacting with a Circular Cylinder: The previous example validated the accuracy of the numerical model for studing dam breaking. This example considers a three-dimensional dam-break wave interacting with a circular cylinder. The tank is 20 m long, 5 m wide, and 10 m high. The volume of water initially contained behind a thin gate is 4 m \times 5 m \times 7 m. The circular cylinder, which has a radius \( r = 1 \) m and height \( h = 5 \) m, is placed in the middle of the tank. The problem definition is shown in Figure 7a. The entire tank is selected as the computational domain with \texttt{nelem=1,315,224} elements. Figure 7b shows a sequence of snapshots of the free surface wave elevation, and Figure 7c the time history of the horizontal force acting on the cylinder.

![Figure 7c: Wave Impact Force on Cylinder](image)

6.3 Sloshing of a 2D Tank Due to Sway Excitation: This example considers the sloshing of a partially filled 2D tank. The main tank dimensions are \( L = H = 1 \) m, with tank width \( B = 0.1 \) m. The problem definition is shown in Figure 8a. Experimental data for this tank with a filling level \( h/L = 0.35 \) have been provided by Olsen\(^ {46}\), and reported in Faltisen\(^ {14}\) and Olsen and Johnsen\(^ {47}\) where the tank was undergoing a sway motion, i.e., the tank oscillates horizontally with law \( x = Asin(2\pi t/T) \). A wave gage was placed 0.05 m from the right wall and the maximum wave elevation relative to a tank-fixed coordinate system was recorded. In the numerical simulations reported by Landrini et al. (2003) using the SPH method, the forced oscillation amplitude increases smoothly in time and reaches its steady regime value in 10 T. The simulation continues for another 30 T, and the maximum wave elevation is recorded in last 10 periods of oscillation.
Figure 8a: 2D Tank: Problem Definition

![Figure 8a: 2D Tank: Problem Definition](image)

Figure 8b: 2D Tank: Time History of Lateral Force $F_x$

We followed the same procedure as Landrini\textsuperscript{35} in our numerical simulation for 32 cases, which correspond to 2 amplitudes ($A = 0.025, 0.05$) and 16 periods, ranging from $T = 1.0 - 1.8$ seconds or $T/T_1 = 0.787 - 1.42$, where $T_1 = 1.27$ seconds. When $h/L = 0.35$ the primary resonances of the first and the third modes occur at $T/T_1 = 1.0$ and $T/T_1 = 0.55$, respectively. The secondary resonance of the second mode is at $T/T_1 = 1.28$ (see Landrini et al. 2003). The present VOF results for the time history of the lateral force $F_x$ when $T = 1.2, 1.3$ and $A = 0.025, 0.05$ are shown in Figure 8b. The corresponding time history of the wave elevation at the wave probe A1 (see Figure 8a) are shown in Figure 8c. Some free surface snapshots are shown in Figure 8d. The present VOF results for maximum wave elevation $\zeta$ at the wave probe A1 (see Figure 8a) are compared with the experimental data and SPH results\textsuperscript{35} in Figure 8e for $A/L = 0.025, 0.05$. 

\textsuperscript{35}American Institute of Aeronautics and Astronautics Paper 2005-0291
The predicted lateral absolute values of maximum forces are compared with the experimental data and SPH results\textsuperscript{35} in Figure 8f for $A/L = 0.05$ (there is no force data available for $A/L = 0.025$). Figure 8g shows the comparison of predicted lateral absolute values of maximum forces for $A/L = 0.025, 0.05$. It can be seen from Figures 8e-f that both maximum wave height and lateral absolute values of maximum forces predicted by present VOF method agrees fairly well with the experimental data and SPH results, with a small phase shift among the three results. Figures 8b and 8c are typical time history plots. It should be noted from these figures that even after a long simulation time (40 periods), steady state results are not generally obtained. This is due to very small damping in the system. Landrini\textsuperscript{35} noted the same behavior in their numerical simulations. As a result, the predicted maximum wave elevation and the lateral absolute values of maximum forces plotted in Figure 8e are average maximum values for the last few periods for the cases when the steady state is not reached.
Figure 8d: Snapshots of Free Surface Wave Elevation for $T = 1.3$ and $A/L = 0.05$

Figure 8e: 2D Tank: Maximum Wave Height (Probe A1)
6.4 Sloshing of a 3D Tank Due to Sway Excitation: In order to study the three dimensional effects, the sloshing of a partially filled 3D tank is considered. The main tank dimensions are \( L = H = 1m \), with tank width \( b = 1m \). The problem definition is shown in Figure 9a. The 3D tank has the same filling level \( h/L = 0.35 \) as the 2D tank. The 3D tank case is run on a mesh with nelem=561,808 elements, and the 2D tank is run on a mesh with nelem=54,124 elements. The numerical simulations are carried out for both 3D and 2D tanks, where both tanks are undergoing the same prescribed sway motion given by \( x = A \sin(2\pi t/T) \). The simulations were carried out for \( A = 0.025 \) and \( T = 1.27 \) (i.e. \( T/T_i = 1 \)). The forced oscillation amplitude increases smoothly in time and reaches its steady regime value in 10 \( T \). The simulation continues for another 70 \( T \). In order to show the 3D effects, the forces are nondimensionalized with \( \rho g L^2 b \) for both 2D and 3D tanks. Figures 9b,c show the time history of the force \( F_x \) (horizontal force in the same direction as the tank moving direction) for both 2D and 3D tanks. Figure 9d shows the time history of the force \( F_z \) (horizontal force perpendicular to the tank moving direction) for 3D tank. It is very interesting to observe from Figures 9e,d that there are almost no 3D effects for the first 25 oscillating periods. The 3D modes start to appear after 25 \( T \), and fully build up at about 40 \( T \). The 3D flow pattern then remains steady and periodic for the rest of the simulation, which is about 40 more oscillating periods.

Figure 8f: 2D Tank: Maximum Absolute Values of Lateral Force \( F_x \) for \( A/L = 0.025,0.05 \)

Figure 9a: 3D Tank: Problem Definition
Figures 9e-g show a sequence of snapshots of the free surface wave elevation for the 3D tank. For the first set of snapshots (see Figure 9e), the flow is still two-dimensional. The 3D flow starts to build up in the second set of snapshots (see Figure 9f). The flow remains periodic three-dimensional for the last 40 periods. Figure 9g show the typical snapshots of the free surface for the last 40 periods. The 3D effects are clearly shown in these plots. Figure 9h shows a sequence of snapshots of the free surface wave elevation for the 2D tank at the same time instance as those shown in Figure 9e. The flow remains periodic and two-dimensional for the rest of the simulation.

Figure 9b: 3D Tank: Time History of Force $F_x$ for a 2D Tank at $A/L = 0.025, T/T_1 = 1$

Figure 9c: 3D Tank: Time History of Force $F_x$ for a 3D Tank at $A/L = 0.025, T/T_1 = 1$

Figure 9d: 3D Tank: Time History of Force $F_z$ for a 3D Tank at $A/L = 0.025, T/T_1 = 1$
Figure 9e: Snap Shots of the Free Surface Wave Elevation for 3D Tank
Figure 9f: Snap Shots of the Free Surface Wave Elevation for 3D Tank
Figure 9g: Snap Shots of the Free Surface Wave Elevation for 3D Tank
Figure 9h: Snap Shots of the Free Surface Wave Elevation for 2D Tank
6.5 Sinking Tank: This test case is included to show the combination of the present techniques with mesh movement and remeshing. The problem definition is given in Figure 10a. The tank is assumed filled with air, and a hole opens up at the bottom of one of the compartments. The mass of the tank was estimated by assuming that when floating empty, half of the tank is outside the water. This yielded approximately $m = 15700 \text{ kg}$. The moment of inertia was estimated at $\Theta_z = 80000 \text{ kg m}^2$. The filling of the tank, and the ensuing movement, can be seen from Figures 10b-f, which show the free surface, velocity and mesh in the plane of symmetry as the calculation proceeds. Note the deformation of the mesh during the run, as well as the effect of the (only 4) automatic global remeshings required to guarantee a proper mesh. The mesh had approximately $\text{nelem=530,000 elements}$. The position and velocity of the center of mass as a function of time, as well as the trajectory are summarized in Figures 10g,h.

![Figure 10a: Sinking Tank: Problem Definition](image)

Figure 10g-h: Sinking Tank: Trajectory of Center of Mass
Figures 10b-f: Sinking Tank: Flowfield at Different Times
6.6 Drifting Ship: This example shows the use of the present methodology to predict the effects of drift in waves for large ships. The problem definition is given in Figure 11a. The ship is a generic LNG tanker, and is considered rigid. The waves are generated by moving the left wall of the domain. A large element size was specified at the far end of the domain in order to dampen the waves. The mesh at the ‘wave-maker plane’ is moved using a sinusoidal excitation. The ship is treated as a free, floating object subject to the hydrodynamic forces of the water. The surface nodes of the ship move according to a 6 DOF integration of the rigid body motion equations. Approximately 30 layers of elements close to the ‘wave-maker plane’ and the ship are moved, and the Navier-Stokes/VOF equations are integrated using the arbitrary Lagrangean-Eulerian frame of reference. The LNG tanks are assumed 80% full. This leads to an interesting interaction of the sloshing inside the tanks and the drifting ship. The mesh had approximately \( m \text{elem}=2,670,000 \) elements, and the integration to 3 minutes of real time took 20 hours on a PC (3.2 GHz Intel P4, 2GB bytes RAM, Linux OS, Intel compiler). Figure 11b shows the evolution of the flowfield, and Figure 11c-d the body motion. Note the change in position for the ship, as well as the roll.

Figure 11a: Ship Adrift: Problem Definition
Figure 11b: Evolution of the Free Surface

Figures 11c-d: Position of Center of Mass
6.7 Bubble Collapse Beneath Generic Ship: This example shows the use of the present methodology to predict the effects of bubble collapse close to structures. The problem definition is given in Figure 12a. The ship is a generic ferry. The reference values for the bubble, which was located at mid-ship and approximately 4 m from the hull, were set as follows: volume \( V_0 = 128 \text{ m}^3 \), density \( \rho_0 = 1.25 \text{ kg/m}^3 \), pressure \( p_0 = 1.0 \times 10^8 \text{ N/m}^2 \), polytropic coefficient: \( \gamma = 1.4 \). The initial radius for the bubble was set to \( r = 2 \text{ m} \). The mesh had approximately \( \text{nelem}=1,530,000 \) elements. Figures 12c-f show the evolution of the flowfield. Note the change of shape for the bubble, first into a torus and subsequently into a rather complex shape. The pressure recorded at midship on the hull is shown in Figure 12g.

![Figure 11e: Roll Angle vs. Time](image)

![Figure 12a: Bubble Collapse: Problem Definition](image)
Figures 12b-g: Surface Mesh and Evolution of Bubble
VII. CONCLUSIONS AND OUTLOOK

A Volume of Fluid (VOF) technique has been developed and coupled with an incompressible Euler/Navier
Stokes solver operating on adaptive, unstructured grids to simulate the interactions of extreme waves and
three-dimensional structures. The present implementation follows the classic VOF implementation for the
liquid-gas system, considering only the liquid phase. Extrapolation algorithms to obtain velocities and
pressure in the gas region near the free surface have been implemented. The VOF technique was validated
against the classic dam-break problem, as well as series of 2-D sloshing experiments and results from SPH
calculations. Other examples presented include violent wave interaction with a column, a leaky tank filling
with water and sinking, an LNG tanker drifting in high sea state, and a bubble collapsing under a generic
ship.
When taken together, these recent advances, which include:

- Accurate, fast incompressible Navier-Stokes solvers operating on adaptive, unstructured grids;
- Robust Volume of Fluid (VOF) techniques for free surface flows;
- Deactivation techniques to speed up calculations; and
- Extensive parallelization of solvers

have made it possible to simulate routinely flows with violent free surface motion with a high degree of
accuracy in a relatively short time, allowing decision-making based on them.
Like every human endeavour, numerical algorithms are subject to continuous improvements. Present research
is directed at the proper treatment of:

- Surface tension;
- Incoming and outgoing waves for 3-D VOF-based free surface flows;
- Free surface wall boundary conditions for RANS, NS cases (i.e., those cases where the velocity at the
  wall $v = 0$); and
- Multiple bubble interaction (splitting, merging, etc.).
VIII. ACKNOWLEDGEMENTS

A considerable part of this work was carried out while the first author was visiting the International Center for Numerical Methods in Engineering (CIMNE) of the Universidad Politécnica de Catalunya, Barcelona, Spain. The support for this visit is gratefully acknowledged.

The authors also wish to thank Mr. Andrea Colagrossi of the INSEAN for providing both experimental data and numerical results for the sloshing of a 2D tank.

References


