# CHAPTER 9

# Simulation of structural response to violent-free surface flows

Rainald Löhner, Chi Yang & Eugenio Oñate

## 9.1 INTRODUCTION

High sea states, waves breaking near shores and moving ships, the interaction of extreme waves with floating structures, green water on deck and sloshing (e.g. in liquid natural gas (LNG) tankers) are but a few examples of flows with violent free surface motion. The bodies exposed to these flows experience large, time-varying forces that can lead to fatigue and/or failure. It is therefore imperative to have a predictive capability in order to guarantee operability and optimize the lifespan under these conditions.

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 [1–6]. 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 cannot 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 [7–19]. 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. In order to illustrate the difficulties that can arise if one treats the complete system, consider a hydrostatic flow, where the exact solution is $v = 0, p = -\rho g \cdot (x - x_0)$, where $x_0$ denotes the position of the free surface. 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 [20] or modify (e.g. enrich) the shape function space [19]. 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. Faced with this dilemma, most flows with free surfaces have been solved neglecting the air. This approach neglects the pressure buildup due to volumes of gas enclosed by liquid, and therefore is not universal. However, in the present case, we have followed this approach, fully aware of the limitations.

The remainder of the chapter is organized as follows: Section 9.2 summarizes the basic elements of the present incompressible flow solver; Sections 9.3 and 9.4 describe the temporal and spatial discretization; Section 9.5 describes the volume of fluid extensions; the attention then turns to the structural solver (sections 9.6 and 9.7), as well as the coupling of flow and structural solvers (section 9.8); some examples are shown in section 9.9; finally, some conclusions are given in section 9.10.

## 9.2 BASIC ELEMENTS OF THE FLOW SOLVER

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

$$\rho v_{,t} + \rho v_a \cdot \nabla_v + \nabla_p = \nabla \mu \nabla v + \rho g \tag{9.1}$$

$$\nabla \cdot v = 0 \tag{9.2}$$

Here $\rho$ denotes the density, $v$ the velocity vector, $p$ the pressure, $\mu$ the viscosity and $g$ the gravity vector. The advective velocity if given by $v_a = v - w$, where $w$ is the mesh velocity. We remark that both the gaseous and liquid phases are considered incompressible, thus equation (9.2). The liquid-gas interface is described by a scalar equation of the form:

$$\Phi_{,t} + v_a \cdot \nabla \Phi = 0 \tag{9.3}$$

For the classic volume of fluid (VOF) technique, $\Phi$ represents the percentage of liquid in a cell/element or control volume (see [7, 8, 13–16, 18]). For pseudo-concentration (PC) techniques, $\Phi$ represents the total density of the material in a cell/element or control volume. For the level set (LS) approach $\Phi$ represents the signed distance to the interface [17].

Since over a decade [21–24] the numerical schemes chosen to solve the incompressible Navier-Stokes equations given by equations (9.1, 9.2) 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).

## 9.3 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:

$$v^{n+1} = v^n + \Delta v^a + \Delta v^p = v^* + \Delta v^p \tag{9.4}$$

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: $v^n \rightarrow v^*$:

$$\left[\frac{\rho}{\Delta t} - \vartheta \nabla \mu \nabla\right](v^* - v^n) + v_a^n \cdot \nabla v^n + \nabla p^n = \nabla \mu \nabla v^n + \rho g \tag{9.5}$$

– Pressure correction: $p^n \rightarrow p^{n+1}$:

$$\nabla \cdot v^{n+1} = 0 \tag{9.6}$$

$$\rho \frac{v^{n+1} - v^*}{\Delta t} + \nabla(p^{n+1} - p^n) = 0 \tag{9.7}$$

– which results in:

$$\nabla \cdot \frac{1}{\rho} \nabla(p^{n+1} - p^n) = \frac{\nabla \cdot v^*}{\Delta t} \tag{9.8}$$

– Velocity correction: $v^* \rightarrow v^{n+1}$:

$$v^{n+1} = v^* - \frac{\Delta t}{\rho} \nabla(p^{n+1} - p^n) \tag{9.9}$$

At steady state, $v^* = v^n = v^{n+1}$ and the residuals of the pressure correction vanish, implying that the result does not depend on the timestep $\Delta t$. $\vartheta$ denotes the implicitness-factor for the viscous terms ($j\vartheta = 1$: 1st order, fully implicit, $\vartheta = 0.5j$: 2nd order, Crank-Nicholson). One can replace the one-step explicit advective-diffusive predictor by a multistage Runge-Kutta scheme [25], 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 of order $k$ for the advective parts may be written as:

$$\rho v^i = \rho v^n + \alpha^i \gamma \Delta t(-\rho v_a^{i-1} \cdot \nabla v^{i-1} - \nabla p^n + \nabla \mu \nabla v^{i-1}), \quad i = 1, k-1 \tag{9.10}$$

$$\left[\frac{\rho}{\Delta t} - \vartheta \nabla \mu \nabla\right](v^k - v^n) + \rho v_a^{k-1} \cdot \nabla v^{k-1} + \nabla p^n = \nabla \mu \nabla v^{k-1} \tag{9.11}$$

Here, the $\alpha^i$ are the standard Runge-Kutta coefficients $\alpha^i = 1/(k + 1 - i)$. As compared to the original scheme given by equation (9.5), the $k - 1$ stages of equation (9.10) may be seen as a predictor (or replacement) of $v^n$ by $v^{k-1}$. The original right-hand side has not been modified, so that at steady state $v^n = v^{k-1}$, 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 time step chosen. Given that the advective and viscous time step limits are proportional to:

$$\Delta t_a \approx \frac{h}{|v|}; \quad \Delta t_v \approx \frac{\rho h^2}{\mu} \tag{9.12}$$

we immediately obtain:

$$\gamma = \frac{\Delta t_v}{\Delta t_a} \approx \frac{\rho |v| h}{\mu} \approx Re_h \tag{9.13}$$

or, in its final form:

$$\gamma = \min(1, Re_h) \tag{9.14}$$

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 (eq. (9.5)). 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 [4, 26–28], finite volumes [29], and finite elements [21–25, 30–39]. One complete time step is then comprised of the following sub-steps:

- Predict velocity (advective-diffusive predictor (eqs. (9.2, 9.7, 9.8));
- Extrapolate the pressure (imposition of boundary conditions);
- Update the pressure (eq. (9.8));
- Correct the velocity field (eq. (9.9));
- Extrapolate the velocity field; and
- Update the scalar interface indicator.

## 9.4 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 equations (9.1–9.3) 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 tetrahedral grids solvers based on edge data structures incur a much lower indirect addressing and CPU overhead than those based on element data structures [40], only these will be considered.

### 9.4.1 *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 [41, 42];
- Taylor-Galerkin (or streamline diffusion) [31, 43, 44]; and
- Edge-based upwinding [24].

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:

$$r^i = D^{ij}\mathbb{F}_{ij} = D^{ij}(f_i + f_j) \tag{9.15}$$

where the $f_i$ are the 'fluxes along edges':

$$f_i = S_k^{ij}F_i^k, \quad S_k^{ij} = \frac{d_k^{ij}}{D^{ij}}, \quad D^{ij} = \sqrt{d_k^{ij}d_k^{ij}} \tag{9.16}$$

$$\mathbb{F}_{ij} = f_i + f_j, \quad f_i = (S_k^{ij}v_i^k)v_i, \quad f_j = (S_k^{ij}v_j^k)v_j \tag{9.17}$$

And the edge-coefficients are based on the shape-functions $N^i$ as follows:

$$d_k^{ij} = \frac{1}{2}\int_\Omega (N_{,k}^i N^j - N_{,k}^j N^i)d\Omega \tag{9.18}$$

A consistent numerical flux is given by:

$$\mathbb{F}_{ij} = f_i + f_j - |\nu^{ij}|(v_i - v_j), \quad \nu^{ij} = \frac{1}{2}S_k^{ij}(v_i^k + v_j^k) \tag{9.19}$$

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 [40]. 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 equation (9.3).

### 9.4.2 *The divergence operator*

A persistent difficulty with incompressible flow solvers has been the derivation of a stable scheme for the divergence constraint (9.2). The stability criterion for the divergence constraint is also known as the Ladyzenskaya-Babuska-Brezzi or LBB condition [28]. The classic way to satisfy the LBB condition has been to use different functional spaces for the velocity and pressure discretization [46]. 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 [47], the p1/iso-p1 element [48], and the p1/p2 element [49]. An alternative way to satisfy the LBB condition is through the use of artificial viscosities [21], 'stabilization' [50–52] or a 'consistent numerical flux' (more elegant terms for the same thing). The equivalency of these approaches has been repeatedly demonstrated (e.g. [21, 40, 47]). 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:

$$\mathbb{F}_{ij} = f_i + f_j, \quad f_i = S_k^{ij}v_i^k, \quad f_j = S_k^{ij}v_j^k \tag{9.20}$$

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

$$\mathbb{F}_{ij} = f_i + f_j - |\lambda^{ij}|(p_i - p_j) \tag{9.21}$$

where the eigenvalue $\lambda^{ij}$ is given by the ratio of the characteristic advective time step of the edge $\Delta t$ and the characteristic advective length of the edge $l$:

$$\lambda^{ij} = \frac{\Delta t^{ij}}{l^{ij}} \tag{9.22}$$

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

$$\mathbb{F}_{ij} = f_i + f_j - |\lambda^{ij}| \left( p_i - p_j + \frac{1}{2}(\nabla p_i + \nabla p_j) \right) \tag{9.23}$$

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

## 9.5 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 *vis 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. In what follows, we will assume that $\Phi$ is bounded by values for liquid and gas (e.g. $0 \leq \Phi \leq 1$ for VOF, $\rho_g \leq \Phi \leq \rho_l$ for PC) and that the liquid-gas interface is defined by the average of these extreme values (i.e. $\Phi = 0.5$ for VOF, $\Phi = 0.5 \cdot (\rho_g + \rho_l)$ for PC, $\Phi = 0$ for LS).

### 9.5.1 *Extrapolation of the pressure*

The pressure in the gas region needs to be extrapolated 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 points 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 (Fig. 9.1a). 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.

### 9.5.2 *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 (Fig. 9.1b).

### 9.5.3 *Keeping interfaces sharp*

The VOF and PC options propagate Heaviside functions through an Eulerian mesh. The 'sharpness' of such profiles requires the use of monotonicity preserving schemes for advection, such as total variation diminishing (TVD) or flux-corrected transport (FCT) techniques [40]. Level set methods propagate a linear function, numerically a much simpler problem. Regardless of the technique used, one finds that shear and vortical flow fields will tend to smooth and distort $\Phi$. Fortunately, both TVD and FCT algorithms allow for limitors that keep the solution monotonic while enhancing the sharpness of the solution. For the TVD schemes Roe's Super-B limitor [53] produces the desired effect. For FCT one increases the anti-diffusion by a small fraction (e.g. $c = 1.01$). The limiting

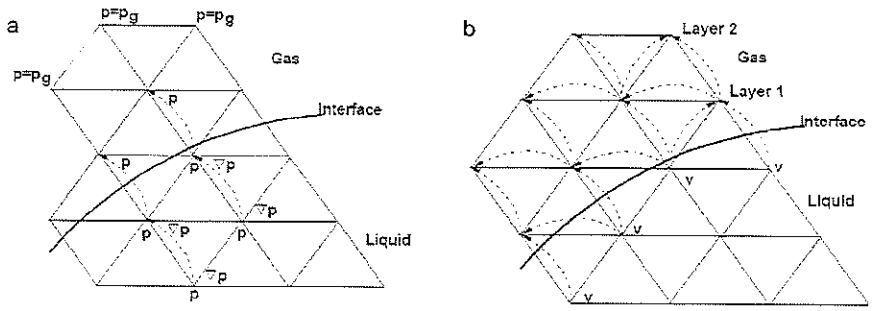


Figure 9.1. Extrapolation of the pressure (a) and velocity (b).

procedure keeps the solution monotonic, while the increased anti-diffusion steepens $\Phi$ as much as is possible on a mesh. With these schemes, the discontinuity in $\Phi$ is captured within 1–2 grid points for all times. For LS the distance-function $\Phi$ must be reinitialized periodically so that it truly represents the distance to the liquid-gas interface.

### 9.5.4 *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 [11, 17, 54]. 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 time step, the total amount of fluid mass is compared to the expected value. The expected value is determined from the mass at the previous time step, plus the mass-flux across all boundaries during the time step. The differences in expected and actual mass are typically very small (less than $10^{-4}$), 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:

$$V_n = \left| v \cdot \frac{\nabla \Phi}{|\nabla \Phi|} \right| \tag{9.24}$$

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. The addition and removal of mass typically occurs at points close the liquid-gas interface, where $\Phi$ does not assume extreme values. In some instances, the addition or removal of mass can lead to values of $\Phi$ outside the allowed range. If this occurs, the value is capped at the extreme value, and further corrections are carried out at the next iteration.

### 9.5.5 *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 [55, 56], 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 [40]. In this way, the basic loops over edges do not require any changes. The if-test whether an edge group is active or deactive occurs outside the inner loops over edges, leaving them unaffected. On scalar processors, edges-groups as small as negrp = 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.

### 9.5.6 *Validation*

The methodology for the calculation of flows with violent free surface motion described above has been repeatedly validated [57–59]. Among the cases treated, we mention: the classic dam-break problem, a series of 2-D and 3-D sloshing experiments and green water on deck for ships.

## 9.6 RIGID BODY MOTION

The movement of rigid bodies can be found in standard textbooks on classical mechanics (e.g. [60]). Due to its nonlinear character, rigid body motion in 3-D is not as straightforward as it may seem. Therefore, a more detailed description of the numerical implementation used is given here. The situation under consideration is shown in Figure 9.2.

Given the position vector of any point of the body:

$$r = r_c + r_0 \tag{9.25}$$

The velocity and acceleration of this point will be:

$$\dot{r} = \dot{r}_c + \dot{r}_0 = v_c + \omega \times r_0 \tag{9.26}$$

$$\ddot{r} = \dot{v}_c + \dot{\omega} \times r_0 + \omega \times (\omega \times r_0) \tag{9.27}$$

Using the vector-relationships:

$$r \times (\omega \times (\omega \times r)) = (r \cdot (\omega \times r))\,\omega - (r \cdot \omega)\,(\omega \times r) = -\omega \times (r \otimes r) \cdot \omega \tag{9.28}$$

and the following abbreviations:

$$m = \int_\Omega dm = \int_\Omega \rho d\Omega, \quad I_{ij} = \int_\Omega r_0^i r_0^j \rho d\Omega \tag{9.29a, b}$$

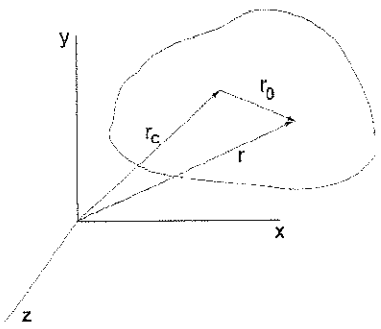


Figure 9.2. Rigid body motion.

$$\Theta = tr(I) \cdot 1 - I = \begin{bmatrix} I_{yy}+I_{zz} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{xx}+I_{zz} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{xx}+I_{yy} \end{bmatrix} \tag{9.29c}$$

We then have the following equations describing balance of forces and moments:

$$m\dot{v}_c = \sum F, \quad \Theta\dot{\omega} - \omega \times (I \cdot \omega) = \sum r_0 \times F \tag{9.30}$$

Observe that in 2-D, the second term on the left-hand side disappears, considerably simplifying the equations. However, in 3-D it usually does not. Another complication that arises only in 3-D is the temporal variation of the inertial matrix $\Theta$. As one can see from equation (9.30), the values of $\Theta$ will vary as the body rotates. This implies that during the simulation one has to follow the local frame of reference of the body. An explicit 2-step scheme is used to integrate the rigid body motion in time. This is reasonable, as in practical calculations the time-scales of the body movement are much larger than those associated with the fluid flow. Thus, $v_c, \omega$ are updated as follows:

$$v_c^{n+1} = v_c^n + \Delta t \dot{v}_c^n, \quad \omega^{n+1} = \omega^n + \Delta t \dot{\omega}^n \tag{9.31a, b}$$

For the time-interval $[t^n, t^{n+1}]$, the average velocities are given by:

$$v_c^{av} = 0,5^*(v_c^{n+1} + v_c^n), \quad \omega^{av} = 0,5^*(\omega^{n+1} + \omega^n) \tag{9.32a, b}$$

Some simulations require several thousand time steps. If one simply uses the velocities obtained at the boundary from equations (9.32a, b), the body shape becomes more and more distorted. This is a purely numerical artifact. It can be explained by looking at the situation depicted in Figure 9.3a. The portions of the body with higher velocity tend to 'elongate' the body. This implies that one ought to impose the exact rigid body motion when updating points on the surface.

With reference to Figure 9.3b, a point lying on the body at time $t = t^n$ is decomposed into three components:

$$r^n = r_c + r_\varphi + r_r \tag{9.33}$$

One can then define unit vectors in the directions of $r_\varphi$ and $r_r$:

$$e_\varphi = \frac{r_\varphi}{|r_\varphi|}, \quad e_\varphi = \frac{r_\varphi}{|r_\varphi|} \tag{9.34}$$

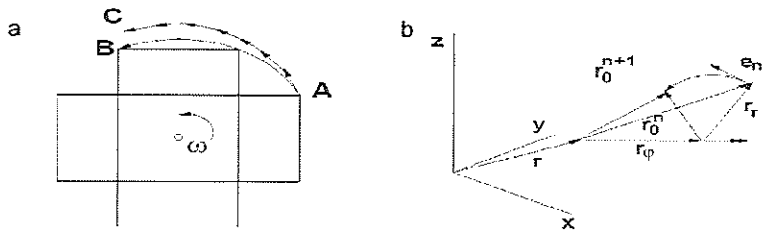


Figure 9.3. Rigid body motion.

And define the vector $e_n$ as:

$$e_n = e_\varphi \times e_r \tag{9.35}$$

Then, given the incremental rotation angle $\Delta\varphi = |\omega^{av}|\,\Delta t$, the new position for $r$ is obtained from:

$$r^{n+1} = r_c + \Delta t v_c^{av} + r_\varphi + |r_r|\,(\cos(\Lambda\varphi)e_r + \sin(\Lambda\varphi)e_n) \tag{9.36}$$

## 9.7 EIGENMODE INTEGRATION

Given a finite element discretization of an elastic structure, the resulting system of equations will be of the form:

$$M\ddot{w} + D\dot{w} + Kw = f \tag{9.37}$$

Here $M, D, K$ denote the mass, damping and stiffness matrices respectively, and $w$ is the vector of nodal displacement variables. The matrices $M, K$ are symmetric positive definite, and can be used to obtain a system of eigenmodes by solving the eigenvalue problem:

$$(-\omega_i^2 M + K)\cdot e^i = 0, \quad i = 1, n \tag{9.38}$$

The eigenvectors satisfy the following important orthogonality properties:

$$e^j M e^i = \delta^{ij}; \quad e^j K e^i = \omega_i^2 \delta^{ij} \tag{9.39}$$

where $\delta^{ij}$ is the Kronecker-$\delta$. The vector of unknowns $w$ can now be written in terms of these eigenvectors as:

$$w = e^i a_i \tag{9.40}$$

resulting in:

$$M e^i \ddot{a}_j + D e^i \dot{a}_i + K e^i a_i = f \tag{9.41}$$

If we also assume $e^j D e^i = d^{ii}\delta^{ij}$, we can decompose the former equation by multiplication with $e^j$. This results in:

$$\ddot{a}_j + d^{ij}\dot{a}_j + \omega_j^2 a_j = f \cdot e^j \tag{9.42}$$

i.e. a decoupled system of ODE's. Each one of these ODE's is integrated in time using Esther explicit Runge-Kutta schemes of higher order (RK4, RK5), or an implicit Newmark scheme [61] of second order. The eigenmode decomposition implicitly assumes an elastic structure and small deformations. As the objects can undergo large movement due to hydrodynamic forces, the rigid body motion is first advanced. Thereafter, the eigenmodes are integrated. In this way the problems associated with spurious 'elongations' of bodies due to rotation are avoided. The eigenmode mesh is independent of the mesh used for the flow simulation. The values at the boundaries required for load and displacement transfer are obtained from a general interpolation/projection library.

## 9.8 FLUID-STRUCTURE COUPLING

The question of how to couple CSD and CFD codes has been treated extensively in the literature (see [62] for a recent survey of the state of the art). Two main approaches have been pursued to date: strong coupling and loose coupling. The strong (or tight) coupling technique solves the discrete system of coupled, nonlinear equations resulting from the CFD, CSD and interface conditions in a single step. At each time step, the resulting matrix system is of the form:

$$\begin{bmatrix} K_{ss} & K_{sf} \\ K_{fs} & K_{ff} \end{bmatrix} \cdot \begin{pmatrix} \Delta U_s \\ \Delta U_f \end{pmatrix} = \begin{pmatrix} r_s \\ r_f \end{pmatrix} \tag{9.43}$$

where the sub-indexes $s, f$ stand for structure and fluid fields, $u$ are the unknowns, $r$ the right-hand sides (sum of internal and external forces/fluxes), the diagonal sub-matrices are the ones usually obtained for each sub-discipline, and the off-diagonal sub-matrices represent the coupling between disciplines. A Jacobi iteration for this complete system may we written as:

$$K_{ss}\Delta u_s^i = r_s - K_{sf}\Delta u_f^{i-1} \tag{9.44a}$$

$$K_{ff}\Delta u_f^i = r_f - K_{fs}\Delta u_s^{i-1} \tag{9.44b}$$

The steps taken in each iteration may also be interpreted as follows:

- Obtain loads from fluid and apply to structure ($K_{sf}\Delta u_f^{i-1}$);
- Obtain new displacements ($\Delta u_s^i$);
- Obtain mesh velocities for the fluid boundary from the structure ($K_{fs}\Delta u_f^{i-1}$); and
- Obtain new flow variables($\Delta u_f^i$).

This interpretation is not exact, as the mesh motion of the flow solver, and the displacement field of the structure are linked beyond nearest neighbors in $K_{sf}$, $K_{fs}$. The interpretation would only be exact for explicit time-stepping schemes. However, it is useful in deriving the so-called loose coupling technique, which solves the complete FSI system given by equation (9.43) by using an iterative strategy of repeated 'CFD solution followed by CSD solution' until convergence is achieved (Fig. 9.4). In this case, the coupling matrices in equations (9.44a, b) contain only the direct load and displacement transfer terms.

Special cases of the loose coupling approach include the direct coupling in time of explicit CFD and CSD codes and the incremental load approach of steady aero- and hydro-elasticity. The variables on the boundaries are transferred back and forth between the different codes by a master code that directs the multi-disciplinary run. Each code (CFD, CSD, ..) is seen as a subroutine, or object, that is called by the master code, or as a series of processes that communicate via message passing. This implies that the transfer of geometrical and physical information is performed between the different codes without affecting their efficiency, layout, basic functionality, and coding styles. At the same time, different CSD or CFD codes may be replaced, making this a very modular approach. This allows for a straightforward re-use of existing codes and the choice of the 'best

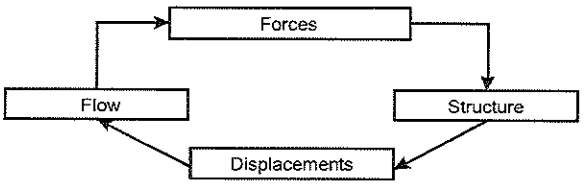


Figure 9.4. Loose coupling for fluid/structure/thermal simultations.

model' for a given application. The information transfer software may be developed, to a large extent, independently from the CSD and CFD codes involved, again leading to modularity and software reuse. For this reason, this approach is favored for industrialization. Indeed, considerable effort has been devoted to develop general, scalable information transfer libraries for displacements, velocities, forces and fluxes [63–67].

The Jacobi iteration shown above can be improved by using an under-relaxed predictor-corrector scheme. Denoting by $i$ the iteration step, $\alpha$ the under-relaxation factor, $xs$ the position of the surface of the structure wetted by the fluid, $\sigma_f$ the stresses exerted by the fluid on the structure, $f(sf)$ the surface deformation due to fluid loads and $g(xs)$ the change of fluid stresses due to surface deformation, the predictor-corrector scheme for each time step takes the following form:

while: not converged:

update structure with fluid loads:

$$X_s^i = (1-\alpha)X_s^{i-1} + \alpha f(\sigma_f^i)$$

update fluid with structure position/velocity:

$$\sigma_f^i = (1-\alpha)\sigma_f^{i-1} + \alpha g(X_s^i)$$

endwhile

Typical under-relaxation factors are in the range $0.5 \leq a \leq 0.9$. Note that the flow of information is the same as in the case of explicit/explicit code coupling. Typically, for the cases shown, no more than 2–3 iterations per time step are required to converge forces, moments, displacements and velocities to a relative error below 1%.

## 9.9 EXAMPLES

### 9.9.1 *Drifting ship*

This example shows the use of the present methodology to predict the effects of drift in waves, as well as sloshing, for large ships. The problem definition is given in Figure 9.5a. The ship is a generic liquid natural gas (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 arbitrarian 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 2.67 millions of elements (Mels) and the integration to 3 minutes of real time took 20 hours on a PC (3.2 GHz Intel P4, 2 GBytes RAM, Lunix OS, Intel compiler). Figure 9.5b shows the evolution of the flow field, and Figures 9.5c, d and 9.6 the body motion. Note the change in position for the ship, as well as the roll.

### 9.9.2 *Drifting fleet of ships*

This example shows the use of the present methodology to predict the effects of drift and shielding in waves for a group of ships. The ships are the same LNG tankers as used in the previous example, but the tanks are considered full. The boundary conditions and mesh size distribution is similar to the one used in the previous example. The ships are treated as a free, floating objects subject to the hydrodynamic forces of the water. The surface nodes of the ships move according to a 6 DOF integration of the rigid body motion equations. Approximately 30 layers of elements close to the

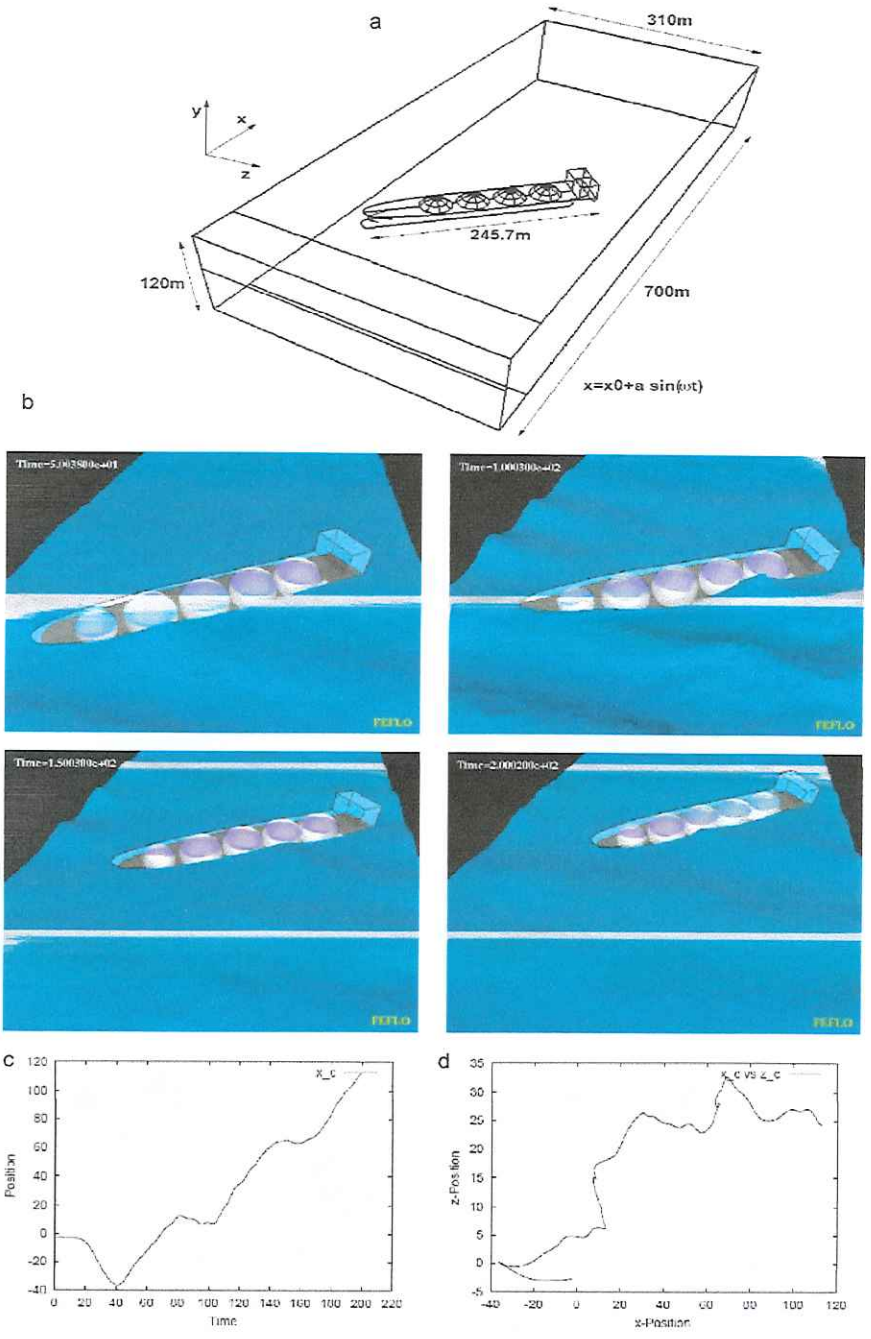


Figure 9.5. Ship adrift: (a) Problem definition; (b) Evolution of the free surface; (c) and (d) Position of center of mass.

'wave-maker plane' and the ships are moved, and the Navier-Stokes/VOF equations are integrated using the arbitrarian Lagrangean-Eulerian frame of reference. The mesh had approximately 10 Mels and the integration to 6 minutes of real time took 10 hours on an SGI Altix using 6 processors (1.5 GHz Intel Itanium II, 8 GBytes RAM, Lunix OS, Intel compiler). Figures 9.7a–d show the evolution of the flow field and the position of the ships. Note how the ships in the back are largely unaffected by the waves as they are 'blocked' by the ships in front, and how these ships cluster together due to wave forces.

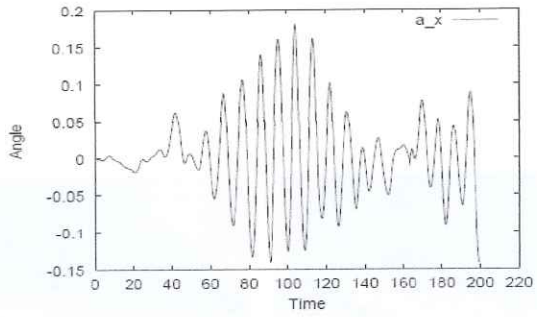


Figure 9.6. Ship adrift: Roll angle *vs* time.

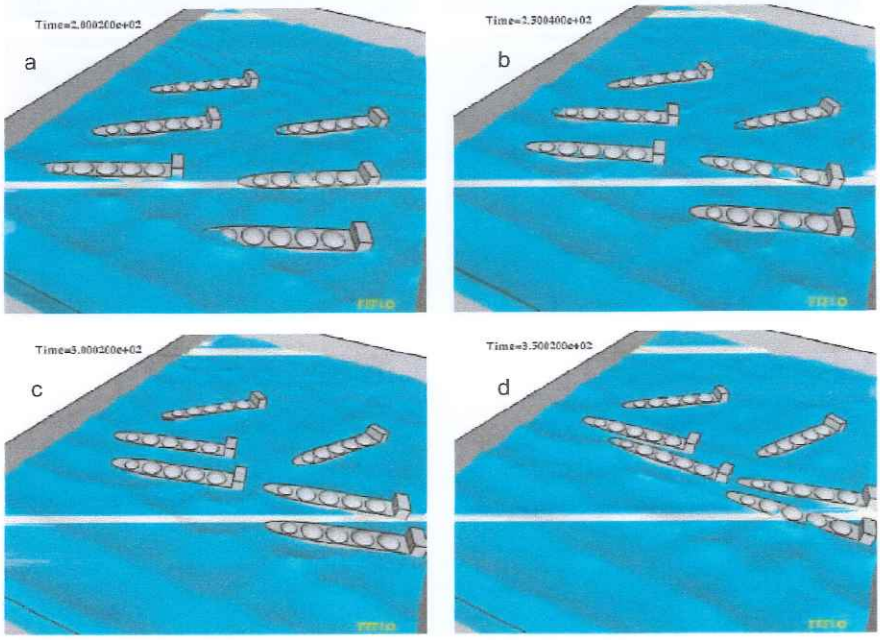


Figure 9.7. LNG tanker fleet: Evolution of the free surface.

### 9.9.3 *Hydroelastic response of ship in heavy sea state*

This example shows the use of the present methodology to predict the hydroelastic effects of large waves on ships. The problem definition is given in Figure 9.8a. The ship is the same generic LNG tanker used before, and is allowed to move rigidly in the $y$-direction, to rotate rigidly in the

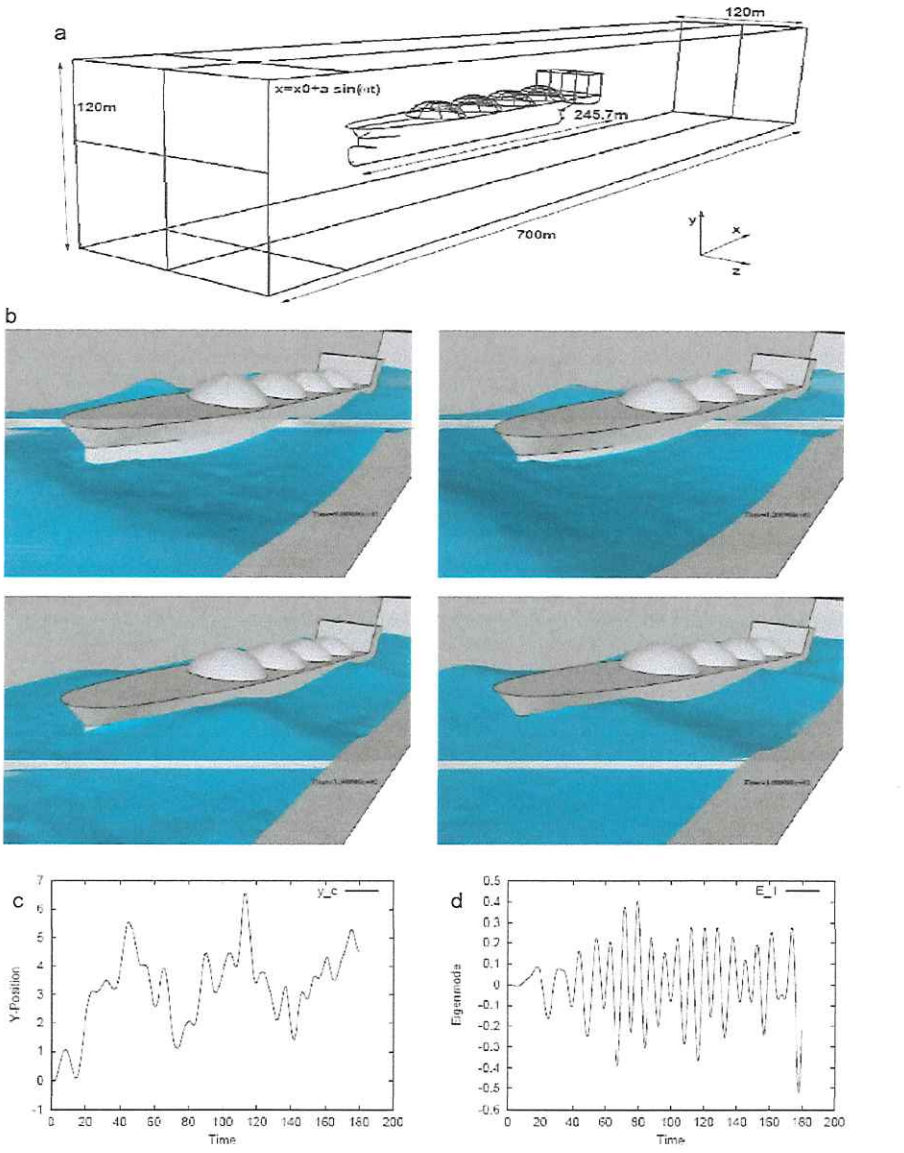


Figure 9.8. Ship in heavy sea state: (a) Problem definition; (b) Evolution of the free surface/ship motion; (c) and (d) Evolution of center of gravity and 1st eigenmode.

$z$-direction, and to deform elastically in the vertical ship direction (original $y$-direction). The three main bending eigenmodes were computed using the ABAQUS CSD code. These modes, together with their stiffnesses and masses were then used as input to FEEIGEN. Boundary conditions and mesh distribution are similar to the previous examples. The surface nodes of the ship move according to the 6 DOF integration of the rigid body motion equations, as well as the eigenmodes. 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 arbitrarian Lagrangean-Eulerian frame of reference. The mesh had approximately 1.25 Mels and the integration to 3 minutes of real time took 2.5 hours on a Dell laptop PC (Intel P4 CPU, 2Gbyte RAM, Suse Linux OS, Intel compiler). Figure 9.8b shows the evolution of the flow field. Note the change in position for the ship, as well as the roll. The position of the center of gravity may be discerned from Figure 9.8c, and the evolution of the first eigenmode from Figure 9.8d. Note the initial 'surfacing' of the ship due to the imbalance of weight and floating forces, as well as the eigenmode values, which lead to approximately 50 cm of whipping at the stern of the ship.

## 9.10 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. The velocities and pressure in the gas region near the free surface are obtained via extrapolation algorithms. This methodology for the prediction of flows has been coupled to 6 degree of freedom (rigid) and eigenmode integrators for structures. This enables an accurate prediction of structural response to loads stemming from flows with violent free surface motion. 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;
- Extensive parallelization of solvers;
- Fast, accurate and conservative displacement and load transfer;
- Loose coupling for implicit flow and structural solvers;

have made it possible to simulate the structural response to flows with violent free surface motion with a high degree of accuracy, allowing decision-making based on them. Like every human endeavor, 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 $v = 0$);
- Multiple bubble interaction (splitting, merging, etc.); and
- Cracking in structures.

## ACKNOWLEDGEMENTS

A considerable part of this work was carried out at 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.

REFERENCES


1. Hino, T.: Computation of free surface flow around an advancing ship by the Navier-Stokes equations. *Proceedings 5th International Conference Numerical Ship Hydrodynamics*, Hiroshima, Japan, 1989.
2. Hino, T., Martinelli, L. and Jameson, A.: A finite-volume method with unstructured grid for free surface flow. *Proceedings 6th International Conference Numerical Ship Hydrodynamics*, Iowa City, IA, 1993.
3. Martinelli, L. and Farmer, J.R.: Sailing through the nineties: Computational fluid dynamics for ship performance analysis and design. In: D.A. Caughey and M.M. Hafez (eds): *Frontiers of computational fluid dynamics*. John Wiley, New York, NY, 1994.
4. Alessandrini, B. and Delhommeau, G.: A multigrid velocity-pressure free surface elevation fully coupled solver for calculation of turbulent incompressible flow around a hull. *Proceedings 21st Symp. on Naval Hydrodynamics*, June 1996, 24–28 June 1996, Trondheim, Norway, 1996.
5. Cowles, G. and Martinelli, L.: Fully nonlinear hydrodynamic calculations for ship design on parallel computing platforms. *Proceedings 21st Symp. on Naval Hydrodynamics*, June 1996, 24–28 June 1996, Trondheim, Norway, 1996.
6. Hino, T.: An unstructured grid method for incompressible viscous flows with a free surface. American Institute of Aeronautics and Astronautics, AIAA-97-0862, 1997.
7. Martin, J.C. and Moyce, W.J.: An experimental study of the collapse of a liquid column on a rigid horizontal plane. *Phil. Trans. Royal Soc. London* A244 (1952), pp. 312–324.
8. Hirt, C.W. and Nichols, B.D.: Volume of fluid (VOF) method for the dynamics of free boundaries. *J. Comput. Phys.* 39 (1981), pp. 201–205.
9. Yabe, T. and Aoki, T.: A Universal solver for hyperbolic equations by cubic-polynomial interpolation. *Comput. Phys. Comm.* 66 (1991), pp. 219–242.
10. Unverdi, S.O. and Tryggvason, G.: A front tracking method for viscous incompressible flows. *J. Comput. Phys.* 100 (1992), pp. 25–37.
11. Sussman, M., Smereka, P. and Osher, S.: A levelset approach for computing solutions to incompressible two-phase flow. *J. Comput. Phys.* 114 (1994), pp.146–159.
12. Yabe, T.: Universal solver CIP for solid, liquid and gas. In: M.M. Hafez and K. Oshima (eds): *CFD Review*. John John Wiley, New York, NY, 1997.
13. Scardovelli, R. and Zaleski, S.: Direct numerical simulation of free-surface and interfacial flow. *Annu. Rev. Fluid Mech.* 31 (1999): pp. 567–603.
14. Chen, G. and Kharif, C.: Two-dimensional Navier-Stokes simulation of breaking waves. *Phys. Fluids* 11:1 (1999), pp. 121–133.
15. Fekken, G., Veldman, A.E.P. and Buchner, B.: Simulation of green water loading using the Navier-Stokes equations. *Proceedings 7th International Conference on Numerical Ship Hydrodynamics*, Nantes, France, 1999.
16. Biausser, B., Fraunie, P., Grilli, S. and Marcer, R.: Numerical analysis of the internal kinematics and dynamics of three-dimensional breaking waves on slopes. *Int. J. Offshore Polar Eng.* 14:4 (2004), pp. 247–256.
17. Enright, D., Nguyen, D., Gibou, F. and Fedkiw, R.: Using the particle level set method and a second order accurate pressure boundary condition for free surface flows. In: M. Kawahashi, A. Ogut and Y. Tsuji (eds): *Proceedings 4th ASME-JSME Joint Fluids Eng. Conf.* FEDSM2003-45144 , 6–10 July 2003, Honolulu, HI, 2003, pp. 1–6.
18. Huijsmans, R.H.M. and van Grosen, E.: Coupling freak wave effects with green water simulations. *Proceedings of the 14th ISOPE*, 23–28 May 2004, Toulon, France, 2004.
19. Coppola-Owen, A.H. and Codina, R.: Improving Eulerian two-phase flow finite element approximation with discontinuous gradient pressure shape functions. *Int. J. Numer. Methods Fluids* (2005).
20. Codina, R. and Soto, O.: A Numerical model to track two-fluid interfaces based on a stabilized finite element method and the level set technique. *Int. J. Numer. Methods Fluids* 4 (2002), pp. 293–301.
21. Löhner, R.: A fast finite element solver for incompressible flows. American Institute of Aeronautics and Astronautics, AIAA-90-0398, 1990.
22. Martin, D. and Löhner, R.: An implicit linelet-based solver for incompressible flows. American Institute of Aeronautics and Astronautics, AIAA-92-0668, 1992.
23. Ramamurti, R. and Löhner, R.: A parallel implicit incompressible flow solver using unstructured meshes. *Comput and Fluids* 5 (1996), pp. 119–132.
24. Löhner, R., Yang, C., Oñate, E. and Idelssohn, S.: An unstructured grid-based, parallel free surface solver. *Appl. Numer. Math.* 31 (1999), pp. 271–293.

25. Löhner, R.: Multistage explicit advective prediction for projection-type incompressible flow solvers. *J. Comput. Phys.* 195 (2004), pp. 143–152.
26. Kim, J. and Moin, P.: Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comput. Phys.* 59 (1985), pp. 308–323.
27. Bell, J.B., Colella, P. and Glaz, H.: A second-order projection method for the Navier-Stokes equations. *J. Comput. Phys.* 85 (1989), pp. 257–283.
28. Bell, J.B. and Marcus, D.L.: A Second-Order Projection Method for Variable-Density Flows. *J. Comput. Phys.* 101:2 (1992), pp. 334–348.
29. Kallinderis, Y. and Chen, A.: An incompressible 3-D Navier-Stokes method with adaptive hybrid grids. American Institute of Aeronautics and Astronautics, AIAA-96-0293, 1996.
30. Gresho, P.M., Upson, C.D., Chan, S.T. and Lee, R.L.: Recent progress in the solution of the time-dependent, three-dimensional, incompressible Navier-Stokes equations. In: T. Kawai (ed): *Proceedings 4th International Symposium Finite Element Methods in Flow Problems*, University of Tokyo Press, 1982, pp. 153–162.
31. Donea, J., Giuliani, S., Laval, H. and Quartapelle, L.: Solution of the unsteady Navier-Stokes equations by a fractional step method. *Comput. Methods Appl. Mech. Eng.* 30 (1982), pp. 53–73.
32. Gresho, P.M. and Chan, S.T.: On the theory of semi-implicit projection methods for viscous incompressible flows and its implementation via a finite element method that introduces a nearly-consistent mass matrix. *Int. J. Num. Methods Fluids* 11 (1990), pp. 621–659.
33. Takamura, A., Zhu, M. and Vinteler, D.: Numerical simulation of pass-by maneuver using ALE technique. *Proceedings JSAE Annual Conf.* (Spring), May 2001, Tokyo, Japan, 2001.
34. Eaton, E.: Aero-acoustics in an automotive HVAC module. *Proceedings American PAM User Conf.*, 24–25 October 2001, Birmingham, MI, 2001.
35. Karbon, K.J. and Kumarasamy, S.: Computational aeroacoustics in automotive design, computational fluid and solid mechanics. *Proceedings 1st MIT Conference on Computational Fluid and Solid Mechanics*, June 2001, Boston, MA, 2001, pp. 871–875.
36. Codina, R.: Pressure stability in fractional step finite element methods for incompressible flows. *J. Comput. Phys.* 170 (2001), pp. 112–140.
37. Li, Y., Kamioka, T., Nouzawa, T., Nakamura, T., Okada, Y. and Ichikawa, N.: Verification of aerodynamic noise simulation by modifying automobile front-pillar shape. JSAE 20025351, *Proceedings JSAE Annual Conf.*, July 2002, Tokyo, 2002.
38. Karbon, K.J. and Singh, R.: Simulation and design of automobile sunroof buffeting noise control. *Proceedings 8th AIAA-CEAS Aero-Acoustics Conf.*, June 2002, Brenckridge, MN, 2002.
39. Camelli, F., Löhner, R., Sandberg, W.C. and Ramamurti, R.: VLES study of ship stack gas dynamics. American Institute of Aeronautics and Astronautics, AIAA-04-0072, 2004.
40. Löhner, R.: *Applied CFD Techniques*. John Wiley and Sons, New York, NY, 2001.
41. Huffenus, J.D. and Khaletzky, D.: A finite element method to solve the Navier-Stokes equations using the method of characteristics. *Int. J. Num. Methods Fluids* 4 (1984), pp. 247–269.
42. Gregoire, J.P., Benque, J.P., Lasbleiz, P. and Goussebaile, J.: 3-D industrial flow calculations by finite element method. *Springer lecture notes in physics* 218. Springer, Berlin, Germany, 1985, pp. 245–249.
43. Kelly, D.W., Nakazawa, S., Zienkiewicz, O.C. and Heinrich, J.C.: A note on anisotropic balancing dissipation in finite element approximation to convection diffusion problems. *Int. J. Numer. Methods Eng.* 15 (1980), pp. 1705–1711.
44. Brooks, A.N. and Hughes, T.J.R.: Streamline upwind/Petrov Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Eng.* 32 (1982), pp. 199–259.
45. Gunzburger, M.D.: Mathematical aspects of finite element methods for incompressible viscous flows. In: D.L. Dwoyer, M.Y. Hussaini and R.G. Voigt (eds): *Finite elements: Theory and application*. Springer, Berlin, Germany, 1987, pp. 124–150.
46. Fortin, M. and Thomasset, F.: Mixed finite element methods for incompressible flow problems. *J. Comput. Phys.* 31 (1979), pp. 113–145.
47. Soulaimani, A., Fortin, M., Ouellet, Y., Dhatt, G. and Bertrand, F.: Simple continuous pressure elements for two- and three-dimensional incompressible flows. *Comput. Methods Appl. Mech. Eng.* 62 (1987), pp. 47–69.
48. Thomasset, F.: *Implementation of finite element methods for Navier-Stokes equations*. Springer, Berlin, Germany, 1981.
49. Taylor, C. and Hood, P.: A numerical solution of the Navier-Stokes equations using the finite element method. *Comput. Fluids* 1 (1973), pp. 73–100.

50. Franca, L.P., Hughes, T.J.R., Loula, A.F.D. and Miranda, I.: A new family of stable elements for the Stokes problem based on a mixed Galerkin/least-squares finite element formulation. In: T.J. Chung and G. Karr (eds): *Proceedings 7th International Conference Finite Elements in Flow Problems*, Huntsville, AL, 1989, pp. 1067–1074.
51. Tezduyar, T.E., Shih, R., Mittal, S. and Ray, S.E.: Incompressible flow computations with stabilized bilinear and linear equal-order interpolation velocity-pressure elements. University of Minnesota Supercomputing Institute (UMSI) Report 90, 1990.
52. Franca, L.P. and Frey, S.L.: Stabilized finite element methods: II. The incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Eng.* 99 (1992), pp. 209–233.
53. Sweby, P.K.: High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM J. Numer. Anal.* 21 (1984), pp. 995–1011.
54. Sussmam, M. and Puckett, E.: A coupled level set and volume of fluid method for computing 3D and axisymmetric incompressible two-phase flows. *J. Comput. Phys.* 162 (2000), pp. 301–337.
55. Löhner, R.: Some useful renumbering strategies for unstructured grids. *Int. J. Num. Methods Eng.* 36 (1993), pp. 3259–3270.
56. Löhner, R.: Renumbering strategies for unstructured-grid solvers operating on shared-memory, cache-based parallel machines. *Comput. Methods Appl. Mech. Eng.* 163 (1998), pp. 95–109.
57. Yang, C., Löhner, R. and Yim, S.C.: Development of a CFD simulation method for extreme wave and structure interactions. *Proceedings 24th International Conference on Offshore Mechanics and Arctic Engineering*, OMAE2005-67422, June 2005, Halkidiki, Greece, 2005.
58. Yang, C. and Löhner, R.: Computation of 3D Flows with violent free surface motion. *Proceedings International Society of Offshore and Polar Engineering (ISOPE) Conferenc*, June 2005, Pusan, Korea, 2005.
59. Löhner, R., Yang, C. and Oñate, E.: On the simulation of flows with violent free surface motion. *Comput. Methods Appl. Mech. Eng.* 195 (2006), pp. 5597–5620.
60. Sommerfeld, A.: Vorlesungen über theoretische Mechanik; Verlag Harri Deutsch, Frankfurt, Germany, 1976.
61. Zienkiewicz, O.C. and Taylor, R.: *The finite element Method*. McGraw Hill, New York, NY, 1988.
62. Bungartz, H.-J. and Schäfer, M. (eds): Fluid-structure interaction: Modeling, simulation, optimization. *Lectures Notes in Computational Science and Engineering*, Vol. 53. Springer, Berlin, Germany, 2006.
63. Löhner, R., Yang, C., Cebral, J., Baum, J.D., Luo, H. Pelessone, D. and Charman, C.: Fluid-structure interaction using a loose coupling algorithm and adaptive unstructured grids. American Institute of Aeronautics and Astronautics, AIAA-95-2259, 1995.
64. Maman, N. and Farhat, C.: Matching fluid and structure meshes for aeroelastic computations: A parallel approach. *Computers and Structures* 54: 4 (1995), pp. 779–785.
65. Brakkee, E., Wolf, K., Ho, D.P. and Schüller, A.: The COupled COmmunications LIBrary. *Proceedings 5th Euromicro Workshop on Parallel and Distributed Processing*, 22–24 January 1997, London, UK, IEEE Computer Society Press, Lo Alamitos, CA, 1997, pp. 155–162.
66. Cebral, J.R. and Löhner, R.: Conservative load projection and tracking for fluid-structure problems. *AIAA J.* 35: 4 (1997), pp. 687–692.
67. Cebral, J.R. and Löhner, R.: Fluid-structure coupling: Extensions and improvements. American Institute of Aeronautics and Astronautics, AIAA-97-0858, 1997.