VII International Conference on Computational Methods in Marine Engineering
MARINE 2017
M. Visonneau, P. Queutey and D. Le Touzé (Eds)

ON THE USE OF EULER AND CRANK-NICOLSON TIME-STEPPING SCHEMES FOR SEAKEEPING SIMULATIONS IN OPENFOAM®

Sopheak Seng*, Charles Monroy, Šime Malenica

Bureau Veritas, Marine & Offshore Division
Research Department, 92571 Neuilly-sur-Seine, France
*e-mail: sopheak.seng@bureauveritas.com

Key words: Seakeeping, OpenFOAM, VOF, Crank-Nicolson, Euler

Abstract. The open-source CFD software package OpenFOAM has reached a maturity level such that it is possible to perform seakeeping simulations using the included VOF-based free-surface URANS (Unsteady-Reynolds-Averaged Navier-Stokes) solver (a.k.a interDyMFoam). This paper describes results of seakeeping tests and the experiences obtained while selecting an appropriate combination of spatial and temporal schemes for wave and seakeeping simulations in a regular head sea condition. Particular attention has been paid to the accuracy level and the convergence rate of temporal schemes `Euler` and `CrankNicolson` since an accurate temporal discretization is known to be very important for wave propagations. Here the numerical results confirm the need for at least a 2^{nd}-order temporal scheme. To improve the stability and the robustness of the existing `CrankNicolson` scheme we customized the code and performed a numerical experiment on the modified `CrankNicolson` scheme where the off-centering parameter c_o is non-uniformly distributed in the domain. The results show that when using a simple distribution of the c_o parameter the stability of the `CrankNicolson` scheme can be restored without having to degrade significantly the numerical order of the scheme. This new approach allows stable simulations to be performed where the incident wave field is propagated more acceptably with a very small decay and, at the same time, keeps the time step large enough to allow the simulations to finish at a reasonable CPU time.

1 INTRODUCTION

The study of ship motion in waves is commonly performed using tools based on potential flow theory which are very efficient in linear waves but cannot account sufficiently for highly nonlinear waves, strong body nonlinearity, high Froude-number and the viscous effects. In the hope of overcoming these issues simulations based on a free-surface URANS (Unsteady Reynolds-Averaged Navier-Stokes) solver is becoming increasingly more popular since the solver and the necessary seakeeping capabilities are available in commercial codes as well as in open-source codes. One of the critical components for seakeeping simulations is the capability to generate waves and propagate them accurately toward the ship. Hence, one of the preliminary

steps to evaluate a free surface URANS code for its seakeeping capability is to establish practical experiences on how wave systems and perhaps more fundamentally the dynamics and kinematics of the free surface can be simulated efficiently and accurately. Physically, besides the incident wave field there are several important wave systems generated by the presence of the ship and its motion. For example, at a certain combination of low Froude numbers and wave frequencies the radiated and diffracted waves are propagating upstream ahead of the ship creating a complicated interfering pattern with the incident wave field. These wave systems travel very far from the ship. Since most of the free surface URANS codes need a finite computational domain the wave systems may cause stability problems, numerical artifacts at wave generation boundaries and a certain level of artificial wave reflection from the truncated boundaries.

The work done in this paper uses the open-source software package OpenFOAM [1, 2] with a necessary customized third-party wave library based on [3] to introduce waves into the computational domain. The objective of the work is to evaluate from a practical point of view the robustness, the efficiency and the accuracy of the public domain OpenFOAM code for seakeeping simulations. A seakeeping case in a head sea condition in a long crested regular wave is selected for the purpose. The configuration of the seakeeping mesh is selected appropriately to include sufficient refinements within the free surface zone, the wave propagation zone, the wave damping zone and the near-hull local regions. The errors related to the propagation of the regular incident wave field on this selected mesh configuration are first evaluated before performing the full seakeeping simulation. The flow solver uses the FVM (Finite Volume Method) discretization for solving URANS equations and the free surface is captured by a VOF (Volume of Fluid) method. The capability to generate and absorb waves is added according to [3]. The mathematical formulation and the important aspects of this solver are presented in Sec. 2 below. Two test conditions are presented. The first test condition is for the propagations of the incident wave field on a 2D mesh with a similar configuration as for the 3D seakeeping mesh. This test condition shall provide a good insight on how well the simulated incident wave field can be propagated from the wave inlet boundary toward the ship. The second test condition is the 3D seakeeping test performed with similar numerical schemes and configuration taken from the best known 2D wave tests. The general numerical setup is described in Sec. 3 and the results of the simulations are discussed in Sec. 4.

2 NUMERICAL MODEL

The OpenFOAM software package is a collection of C++ libraries which provide core functionalities for solving partial differential equations. One of its most developed functionalities is the FVM discretization using unstructured polyhedral meshes. Several basic solvers are included in the package. The flow solver used in this work is a VOF based incompressible two-phase solver. It solves the URANS equations and a modified transport equation of the phase fraction field. Details related to the basic principle of VOF can be found in [6]. Air and water are considered immiscible and treated as viscous and incompressible. While the computational domain includes both air and water with a density ratio up to approximately 1000, the mass and momentum conservation are formulated under the assumption that the density ρ and viscosity μ of the effective fluid can be computed using the volume phase fraction field as weighting factor as in

Eq. (1) below:

$$\rho = \alpha\rho_w + (1-\alpha)\rho_a \tag{1}$$
$$\mu = \alpha\mu_w + (1-\alpha)\mu_a$$

where the subindices "a, w" indicate air and water, respectively. The volume phase fraction field α must be bounded between 0 and 1 since it describes the volume fraction of water in the control volume. Here the surface tension is considered negligible. The governing equations for the laminar model (the turbulence model is runtime selectable) including the modified transport equation read

$$\nabla \cdot \boldsymbol{u} = 0 \tag{2}$$

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot \alpha \boldsymbol{u} + \nabla \cdot [\alpha(1-\alpha)\boldsymbol{u}_r] = 0 \tag{3}$$

$$\frac{\partial \rho \boldsymbol{u}}{\partial t} + \nabla \cdot \rho \boldsymbol{u}\boldsymbol{u}^T - \nabla \cdot \left[\mu(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)\right] = -\nabla p_{rgh} - (\boldsymbol{g} \cdot \boldsymbol{x})\nabla\rho \tag{4}$$

The pressure field p_{rgh} is defined in relation to the static pressure p as $p_{rgh} \equiv p - \rho \boldsymbol{g} \cdot \boldsymbol{x}$, where $\boldsymbol{g}$ is the vector representing the gravitational acceleration and $\boldsymbol{x}$ is the position vector. The velocity/pressure coupling is resolved in time using a modified PISO algorithm known to the OpenFOAM community as the `PIMPLE` algorithm which is a unified implementation of the PISO algorithm (pressure-implicit with split-operator [7]) and the SIMPLE algorithm (Semi-Implicit Method for Pressure Linked Equations). Through user-selected run-time controlling parameters the `PIMPLE` algorithm can be configured to operate in pure PISO or SIMPLE mode. More importantly it enables a combined use of SIMPLE/PISO mode (a.k.a `PIMPLE` mode) to achieve a more efficient time stepping since in this mode the time step size shall be less constrained than in the PISO mode. The `PIMPLE` mode has been used in all simulations performed this work.

To keep the α-field bounded between 0 and 1, Eq. (3) is solved using the semi-implicit MULES solver (Multidimensional Universal Limiter with Explicit Solution). Although MULES has been introduced into the OpenFOAM library almost a decade ago the literature on this solver is sparse. The work done in [8, 9] are examples of recent efforts to describe the MULES solver. The ability of MULES in keeping the α-field bounded is documented in [9]. The artificial term in Eq. (3) (the third time on the left-hand-side) is added in an attempt to prevent smearing of the interface due to numerical diffusion in the α-field. This term is designed to compress the interface by adding artificial flux to the equation and requires an empirical model for $\boldsymbol{u}_r$ (see also [10]), the relative velocity between the air and water, which in theory should be zero but in practice is different from zero due to the numerical smearing and the difficulty of representing the interface exactly in the VOF formulation. The treatment of the transport equation for the α-field using this artificial term in a combination with the MULES solver can be classified as an algebraic VOF algorithm. While it is possible to obtain good results with this algorithm the conclusion put forward by [9] is that MULES is inferior to geometric VOF algorithms. Since the OpenFOAM software package does not provide other options, the current work uses exclusively the semi-implicit `MULES` in all simulations.

The FVM discretization of the governing equations is managed by the OpenFOAM core library which follows the standard FVM practice of converting volume integrals to surface integrals using Gauss's theorem. The integration and interpolation scheme for each term in the

equations are made runtime selectable. These include the time derivative term for the momentum equation and the transport equation, the divergence and the gradient operators. The 2^{nd}-order upwind scheme has been selected for the convective term in the momentum equations and the 2^{nd}-order central differencing scheme for the diffusive term. The α-field always has a steep gradient near the free surface. Hence, the 2^{nd}-order high-resolution shock capturing scheme `van Leer` is selected for the convective term in the transport equation for the α-field. For the time derivative terms OpenFOAM core library provides three different schemes named: `Euler`, `backward`, and `CrankNicolson`. The Euler scheme in OpenFOAM is an implementation of the backward Euler integration algorithm which is implicit and known to be very stable but only 1^{st}-order accurate. The numerical results of wave propagations which are presented and discussed in Sec. 4.1 show that the Euler scheme, although very stable, is 1^{st}-order accurate and is not sufficient for propagating waves from the wave generation boundaries toward the ship. To maintain an acceptable wave condition within the vicinity of the ship the order of the temporal scheme needs to be at least 2^{nd}-order accurate which can be achieved using either `backward` or `CrankNicolson`. In OpenFOAM `backward` is the 2^{nd}-order three-time-steps backward differentiation formula and `CrankNicolson` is a modified Crank-Nicolson scheme where an off-center parameter with a value between 0 and 1 has been introduced to control the weight toward either standard Crank-Nicolson scheme or the implicit Euler scheme. It appears that the `backward` or the `CrankNicolson` schemes can be readily applied to obtain a 2^{nd}-order integration in time. However, the `backward` scheme is known to be unbounded. Hence it cannot be applied to the transport equation for the α-field since it will cause extreme difficulty to control a bounded solution of the α-field to within 0 and 1. The option to use the `backward` scheme in MULES is simply disabled. The `CrankNicolson` scheme, on the other hand, can be made bounded and is supported by MULES to obtain a consistent 2^{nd}-order temporal accuracy not only for the momentum equations but also for the transport equation of the α-field. The simulation performed in this work, however, quickly reveals that the `CrankNicolson` scheme may cause the numerical solution to have an oscillatory convergence. When the scheme is used directly in FSI (fluid structure interaction) simulations where the body motion is resolved through a partitioning scheme where the flow solver and the body-motion solver are executed alternately (with fixed under-relaxation of the body acceleration) the oscillatory flow solutions are having a strong negative influence on the convergence behavior of the FSI iterations. When the FSI converges, the number of FSI iterations increases significantly. Under certain circumstances the FSI iterations diverges. The oscillatory convergence behavior of the `CrankNicolson` scheme is well known and the oscillatory behavior is associated with a too large Courant number ($C \gg 1$). Under normal circumstances the Courant number is computed as the global maximum which is conservative since the Courant number is critically high at a few local cells only. These cells are typically inside local refinement regions where the cells are very small due to either physical constraints such as boundary layer flow and the selected turbulence model or technical issues related to automatic mesh generations. The latter seems evident when using automatic mesh generation tools which are based on unstructured split-hexahedra. These are e.g. the freely available `snappyHexMesh`[1] or similar commercial alternatives such as `HEXPRESS`™ or `STAR-CCM+`.

[1] A utility provided by the OpenFOAM software package to generate unstructured 3-dimensional meshes containing mostly hexahedra and spitted hexahedra

Assuming that these critical small cells region are an unavoidable product of the automatic mesh generation tools and the physics of the flow within these regions do not require the cells to be so small it is natural to think of a solution to trade on the unnecessarily high spatial resolution for an increasing numerical stability. One of the easiest solution which is put under a numerical experiment in this paper is to adjust locally the off-center parameter found in the `CrankNicolson` scheme. In OpenFOAM the implementation of this solution is straightforward costing no more than a few lines of codes added to the source code of the flow solver. Here this solution shall be described in detail.

Recall that the standard Crank-Nicolson scheme can be viewed as a half-and-half weighting between the forward and backward Euler-schemes, the weight coefficient can be considered as an adjustable parameter which controls the numerical behavior of the scheme. In OpenFOAM this weighting coefficient has been implemented as described below. Consider the following generic 1st-order ODE (ordinary differential equation)

$$\dot{y} = f(t, y) \tag{5}$$

According to the forward Euler scheme the integration from time step n to $n+1$ is approximated by $y_{n+1} - y_n \approx h f_n$ where $h = t_{n+1} - t_n$ is the time step size and $f_n = f(t_n, y_n)$. For the backward Euler scheme the approximation is $y_{n+1} - y_n \approx h f_{n+1}$. A weighting coefficient $\gamma = [0, 1]$ has been introduced as follows:

$$y_{n+1} - y_n \approx \gamma h f_{n+1} + (1 - \gamma) h f_n \tag{6}$$

For $\gamma = 1/2$ the scheme is equivalent to the standard Crank-Nicolson scheme. The scheme can be made equivalent to the forward explicit Euler scheme by selecting $\gamma = 0$. For stability reason, however, it is advisable to keep γ between $1/2$ and 1, where $\gamma = 1$ is equivalent to the fully implicit 1^{st}-order backward Euler scheme. In OpenFOAM γ can be adjusted within the range of $\gamma = [1/2, 1]$. The adjustment is made available during run-time through an off-center coefficient c_o which is related to γ as follows

$$c_o \equiv \frac{1 - \gamma}{\gamma} \quad \text{for} \quad \gamma = [0.5, 1] \tag{7}$$

From this definition c_o will have a value between 0 and 1. The standard 2^{nd}-order Crank-Nicolson scheme ($\gamma = 0.5$) is specified as $c_o = 1$, and the implicit 1^{st}-order backward Euler ($\gamma = 1$) as $c_o = 0$. It is noted that a 2^{nd}-order accuracy can be expected only by using $c_o = 1$ (i.e. $\gamma = 0.5$). Clearly if the simulation runs stable at $c_o = 1$ it will not be wise to lower this coefficient. Changing the value of c_o slightly, says $c_o = 0.9$, will increase numerical dissipation and reduce the order toward the fully 1^{st}-order backward Euler. The lost of accuracy shall be judged against the benefit of having an additional numerical dissipation which means better numerical stability. In cases the simulations cannot proceed at $c_o = 1$ due to stability reasons this additional dissipation may become very valuable if it restores fully the numerical stability with a minimal and local impact on the flow features of interest. In seakeeping the flow features of interest, preliminarily, are the wave field at the position of the ship. From this point of view it is justifiable to allow a non-constant spatial dependency of the c_o parameter if the benefit of added numerical stability outweighs the lost of accuracy. Obviously it is not trivial to determine

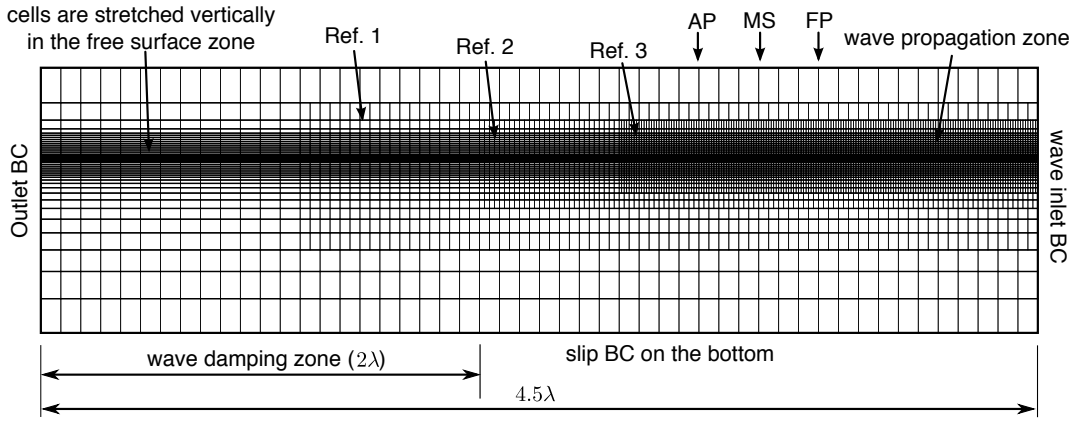

Figure 1: A sketch of the mesh configuration and the boundary conditions (BC) defined in the 2D wave tests (without the ship). The longitudinal position of the ship is indicated by AP (aft perpendicular), MS (midship) and FP (forward perpendicular). The wave length is $\lambda \approx 11.84$ m c.f. Table 1.

an optimum spatial dependency for c_o. The first aims, however, is not to have a fully optimal distribution for c_o but to establish some evidences showing that the benefit of having a non-constant spatial distribution of c_o can be achieved in practice. The numerical experiments constructed in this paper shall provide more insight on this subject.

3 NUMERICAL SETUP

It is generally acknowledged that seakeeping simulations based on a FVM free-surface URANS solver are computationally very expensive which provides a strong motivation for keeping the cell count low. The seakeeping mesh is generated using unstructured body-fitted split-hexahedra mesh. Waves are introduced into the domain by imposing wave kinematics and free surface conditions at the wave generation boundaries. To prevent wave reflection from the outlet boundary we use an explicit relaxation zone technique based on the description in [3]. A study on this wave generation and absorption technique can be found in [4].

Two test conditions are presented. The first test consists of wave propagations in a two-dimensional wave flume and the second test is for three-dimensional simulations of a ship moving in head waves. A mesh for the 2D case is constructed without the ship to have the same refinement configurations as in the 3D seakeeping case. This 2D mesh essentially is a slice of the 3D mesh through a plane parallel to the symmetry plane of the ship. Therefore the numerical results of the long-crested regular waves on this 2D mesh are considered representative for the results on the 3D mesh. Figure 1 shows an illustration of the 2D mesh used in the wave propagation tests. The mesh resolution shown in this figure has been reduced considerably for visualization purposes. Here it is shown that three isotropic volume refinement regions (Refinement box 1, 2 and 3, see Fig. 1) are defined to obtain a reasonable coarsening toward the outlet boundary and a good mesh resolution within the wave propagation zone and around the ship. Cells are stretched vertically away from the free surface region. The total length of the domain in the wave propagation direction is 4.5λ where λ (see Table 1) is the wave length predicted by the nonlinear stream function wave theory [5]. A region next to the outlet boundary

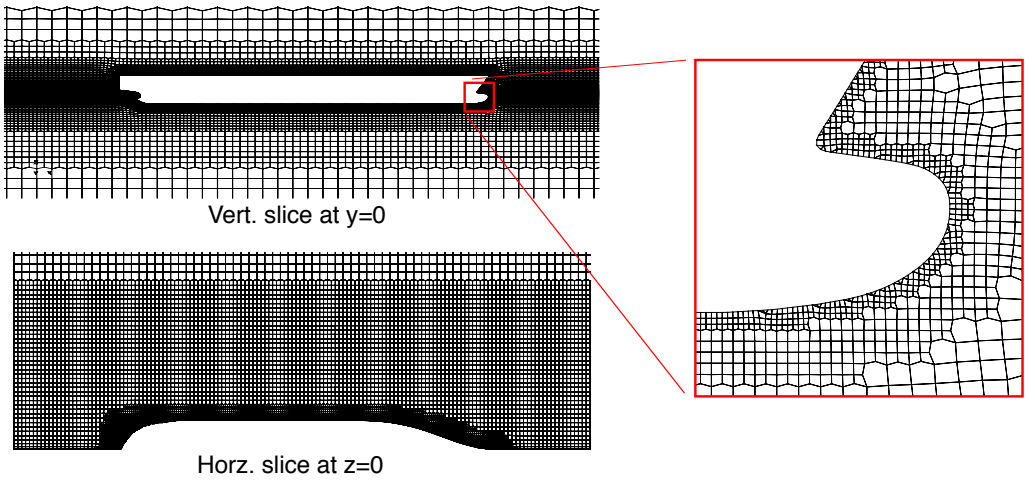

Figure 2: A close-up view of the cells at the vicinity of the hull in the 3D seakeeping mesh

equivalent to 2λ is reserved for wave damping. According to results published in [3] and [4] the selected length of 2λ is more than sufficient to prevent wave reflection from the outlet boundary. The resulting 2D mesh has approximately 28000 cells in total and the wave propagation zone has about 190 cells per λ and 12 cells per wave height. The width to height cell aspect ratio in this zone is approximately 4.

For the seakeeping simulation the container ship KCS in model scale is selected. The experimental data for a head sea condition is publicly available e.g. in [11] and in the proceeding of the workshop on CFD in ship hydrodynamics in Tokyo Dec. 2^{nd}-4^{th} 2015 [12]. The selected test case is label `KCS-CASE-2.10-c5` in [12]. A short summary of the test condition is given in Table 1. The template for the 3D mesh is the same as for the 2D mesh (Fig. 1) including the isotropic volume refinement regions and the stretching toward the free surface zone. Here, the ship is cut out and the cut-cells are refined and fitted to the ship hull. A close-up view of the cells in the vicinity of the hull is shown in Fig. 2. Cells next to the hull within a normal distance of about 4 times the cell length in the wave propagation zone are refined gradually 2 times as shown on the magnified area in Fig. 2. This 3D mesh has been produced using open-source meshing tools including `blockMesh`, `refineMesh`, `snappyHexMesh` and `cfMesh` which (apart from the latter) are readily available in the OpenFOAM package. These tools are known to have difficulty to produce good quality prism layers on the hull. Hence the simulations are performed without the prism layers and the turbulence model has been de-activated assuming a laminar viscous flow passed the ship. The consequence of this configuration is that the viscous shear stress is not correctly captured by the flow solver which affects strongly the predicted total hull resistance. However, heave and pitch motions are dominated by the vertical forces. The viscous shear stress which contributes mainly to the horizontal forces in the direction of the forward speed therefore will have a negligible effect on heave and pitch motion. The symmetric flow in head sea conditions are exploited such that only haft of the ship (and the fluid domain) is modeled. The length, width and height of the 3D computational domain are respectively $\approx (4.5, 1.5, 0.7)\lambda$. The bottom is flat at a water depth of 5.5 m. There are 3λ from the outlet boundary to the

ship whereas 2λ are reserved for wave damping. The wave propagation zone in front of the ship is 1λ and it spans from the wave inlet boundary to the ship. From the ship to the side wall of the computational domain there is 1.5λ. This configuration produces approximately 1.8 millions cells in total.

4 NUMERICAL RESULTS AND DISCUSSION

All simulations performed in this work use a modern HPC (high performance computing) cluster located at Ecole Centrale de Nantes (ECN) in France which consists of 266 computed nodes. Each node provides 24 cores (dual processors Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz). The 2D wave simulations are performed in parallel using 24 cores. This choice is arbitrary and perhaps slightly excessive considering the fact that the 2D mesh has only about 28000 cells in total. The number of outer and inner iterations in the `PIMPLE` algorithm is fixed to 5 and 4, respectively; and the iterative non-orthogonal correction is set to 1. Hence, at each time step, the flow solver solves the predictive momentum equations 5 times and the Poisson's equation for the dynamic pressure 20 times. On 24 cores each time step in the 2D wave simulations is completed within 0.4 to 0.5 seconds real time. The 3D seakeeping simulations (total cell count $\approx$ 1.8 millions) use 72 cores with 20 outer iterations, 3 inner iterations and 1 iterative non-orthogonal correction. As a result 20 predictive momentum equations and 120 Poisson's equations are solved at each time step. The time required to complete one time step in the 3D simulations using the selected 72 cores varies between 38 and 43 seconds real time. The computed nodes spend a large amount of the CPU time to recompute the geometrical properties of each cell and to deform the 3D mesh according to the rigid-body motion of the hull. It is noted that most likely the performance can be improved significantly by selecting a more efficient technique to handle body motions and dynamic meshing in the computational domain.

The wave condition is considered nonlinear and the stream function wave model has been selected as the input wave model at the wave inlet boundary. At the initial time the volume fraction field α and the velocity field $\boldsymbol{u}$ are initialized according to this model. Three numerical wave probes are defined in the 2D domain at AP (aft perpendicular), MS (midship) and FP (forward perpendicular). These numerical wave probes compute the free surface elevation from the volume fraction field by assuming that the free surface is at $\alpha = 0.5$. While this assumption allows a simple and efficient reading of the free surface elevation it must be stressed that the probed values are dependent on the local grid resolution at the probe locations. A primitive test shows that on the meshes used in in the current work the errors in the numerical wave probes affect the resulting 1^{st}-harmonic (normalized by the analytical equivalent) in the order of $O(10^{-3})$.

4.1 Wave propagations in the 2D domain

The free surface elevation at midship after more than $40T_e$ (encountered wave period) is shown in Fig. 3 (left). The crest and trough are not symmetric which indicates that the wave condition is slightly nonlinear. The time step is kept constant at 0.01 s which is equivalent to $T_e/\Delta t \approx 187$. The simulation become unstable very quickly when using the standard Crank-Nicolson scheme (i.e. `CrankNicolson 1`, see Table 2). After a few T_e the Courant number grows from 0.7 to more than 200 and the simulation crashes at about $4.5T_e$. While perhaps the

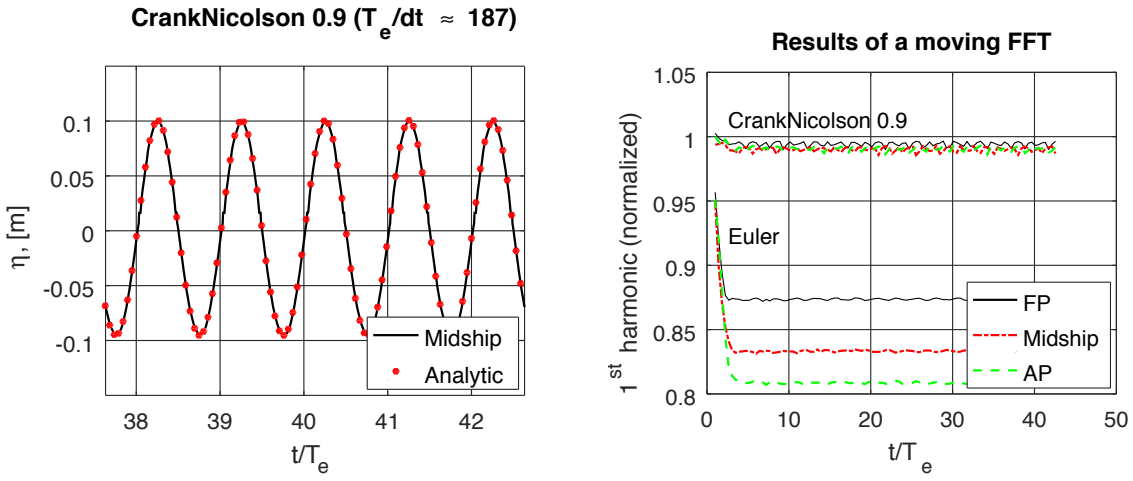

Figure 3: Free surface elevation η at midship after 40 encountered period (left), and the results of a moving FFT (window size of 1 T_e) applied to η at AP, midship and FP (right). The simulations are performed using FVM schemes as shown in Table 2. The time step is constant at 0.01 s ($T_e/\Delta t \approx 187$).

stability can be restored by reducing the time step significantly we have not explored further this option since it may quickly become very resource demanding to reduce the time step much further. An alternative option is to reduce the off-centering coefficient c_o. Here at $c_o = 0.9$ (i.e. `CrankNicolson 0.9`) the simulation runs stable for more 40 encountered period. It is possible to lower the c_o value even further at the cost of additional lost of the wave amplitude as shown in Fig. 4 (left). When the time step is fixed at 0.01 s ($T_e/\Delta t \approx 187$) there is a lost of about 4% at midship at $c_o = 0.7$ against 2% at $c_o = 0.9$. Further improvements can be observed when reducing the time step as shown in Fig. 4.

Figure 3 (right) shows 1^{st} harmonics of the free surface elevations. These 1^{st} harmonics are the results of moving FFTs applied with a window size of $1T_e$. Here it is evident that when using the `CrankNicolson` scheme the resulting the free surface elevations η are very similar at FP, midship and AP. Hence the the wave amplitude does not show sign of a significant decay when it is propagated numerically from FP to AP. The `Euler` scheme too is stable but the wave amplitudes are very different between FP, midship and AP. Indeed waves are decaying strongly over the ship length ($L_{pp} \approx 0.5\lambda$). With the `Euler` scheme the incident wave has lost more than 12% of its 1^{st} harmonic amplitude at FP and 19% by the time it reaches AP. Since there is only about 0.5λ from FP to AP the decay rate is almost 13% per λ which is very significant in most engineering applications. Figure 4 (right) shows the temporal convergence rate for the `Euler` scheme compared to the modified `CrankNicolson` scheme with the off-centering coefficient c_o varied between 0.7 and 0.9. For these the temporal convergence tests the 2D mesh is kept the same and so are the spatial discretization schemes as shown in Table 2. As expected the `Euler` scheme is (approximately) 1^{st}-order. The `CrankNicolson` schemes at $c_o = 0.7$ and $c_o = 0.8$ are confirmed here to be 1^{st}-order also but the level of errors are one order of magnitude lower than the `Euler` scheme. Theoretically the `CrankNicolson` scheme is fully $2^n d$-order accurate only when $c_o = 1$. Since the scheme currently is not stable at $c_o = 1$ the closest results

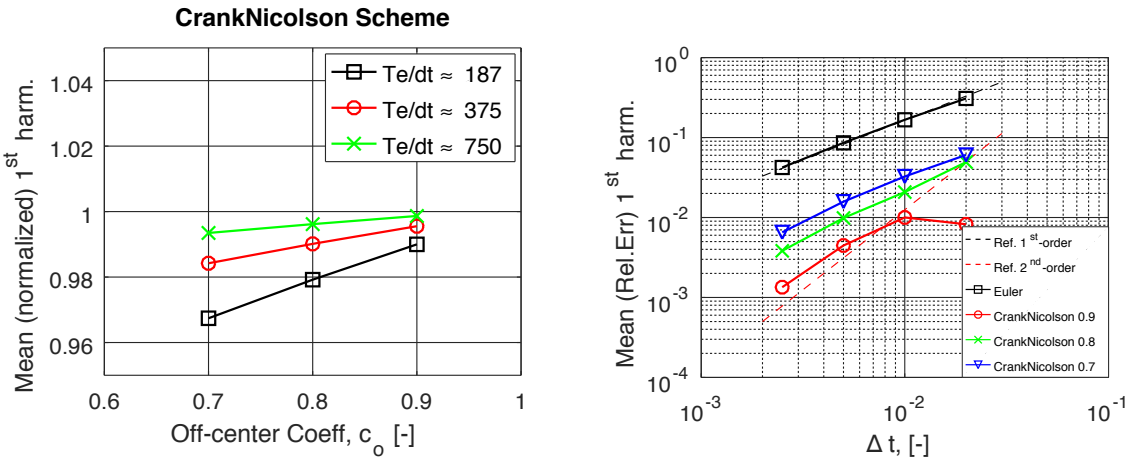

Figure 4: The dependency of the errors on the off-centering coefficient c_o (left). The temporal convergence rate (right). The accuracy of the modified `CrankNicolson` scheme is improving when c_o is approaching its maximum value of 1. At $c_o = 1$ the simulation is not stable.

are with $c_o = 0.9$ where the errors seem to indicate a convergence rate of close-to 2^{nd}-order (≈ 1.7). Nonetheless these results must be evaluated taking into account the previously discussed limitations of the numerical wave probes used to determine the free surface elevation from the volume fraction field. The numerical wave probes produce errors in the order of $O(10^{-3})$ which is the same order as the errors shown for $c_o = 0.9$ in Fig. 4 (right). Hence the errors seen here at $c_o = 0.9$ may very well contain a relatively significant contribution from the errors in the numerical wave probes.

4.2 Seakeeping simulations in regular head sea conditions

When using the `Euler` scheme the seakeeping simulations run stable for more than $17T_e$. The 2D wave propagation tests show clearly that the `Euler` scheme is not appropriated since waves are decaying significantly from FP to AP. The 3D seakeeping mesh has the similar mesh configuration as in 2D and all relevant numerical schemes as well as the time step are kept the same between 2D and 3D simulations. Hence it is expected that the resulting incident waves when using the `Euler` scheme in 3D have the same decay rate as seen on the 2D mesh. When inspecting the resulting heave and pitch motions, however, it is not clear how the condition where waves at FP and AP may have different amplitudes affects the heave and pitch motions. Figures 6 and 7 show results of a moving FFT on heave and pitch motions. Since the simulated incident waves are decaying and therefore have some loss of amplitude the resulting heave and pitch motions are reduced accordingly. The resulting ship motions are expected to be improved when using the `CrankNicolson` scheme.

With `CrankNicolson 0.9` the 2D wave simulation on the 2D mesh is stable. However, the 3D seakeeping mesh has small irregular cells which are produced in the process of cutting and fitting the mesh to the ship hull. These irregular cells are expected to have a negative influence on the overall numerical stability. Furthermore the presence of the ship hull and its motion induces a larger local flow in the vicinity of the hull which causes the local Courant number to

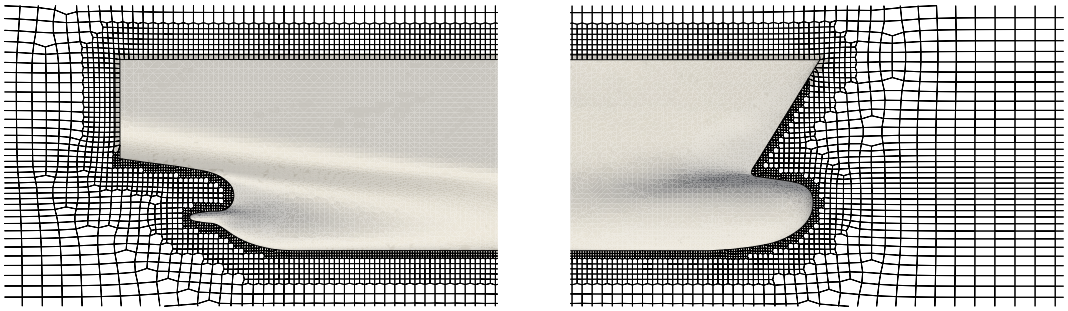

Figure 5: Cells next to the ship hull at a distance of less than 0.0245 m (shaded area) are selected for use with $c_o = 0$ (i.e. the same as the `Euler` scheme). All other cells not included in this selection are integrated using $c_o = 0.9$. Close-up view at stern (left), at bow (right).

increase. It is our discovery that the 3D seakeeping simulation at $c_o = 0.9$ becomes unstable very quickly. Once again the stability can be restored by reducing the value of c_o further toward 0 (i.e. the `Euler` scheme) and at same time decrease the time step (see Fig. 4) considerably to maintain an acceptable accuracy of wave simulations. In the current 3D seakeeping case we are exploring a more efficient solution by reducing c_o not globally but in locally selected cells where the local Courant numbers either are considered too large or have poor geometrical properties. The selection is done based on authors own intuition and convenience rather than on a rigorous optimization. Figure 5 illustrates these local cells (shaded region). The selection is done manually prior to each run using cell selection tools available in OpenFOAM. Here a distance based cell selection tool has been used to selected all cells within a normal distance of 0.0245 m to the ship hull. This short distance covers a few layers of cells which, in some regions, can be very small. The simulation is performed using `CrankNicolson 0.9` in all cells except those selected for used with `CrankNicolson 0` (i.e. equivalent to the implicit backward `Euler` scheme, see also Eq. (7)). With this selection the off-centering coefficient c_o changes abruptly from 0.9 to 0. While it is possible to distribute the c_o values more smoothly no such attempt has been made since the current selection is already very stable. The seakeeping simulations run for more than $17T_e$. The resulting heave and pitch motions are shown in Fig. 6 and 7 with label `CN0.9+Euler`. The time series of heave and pitch motions are processed in to the frequency domain using a moving FFT with a window size of $1T_e$. The evolution of the 1^{st} harmonic confirms that the resulting motions have reached a quasi-steady state in less than $17T_e$. More importantly the predicted amplitude of the 1^{st} harmonics (for both heave and pitch) are significantly closer to the experiments. Some small variations still exist, possibly due to disturbance produced by a non-perfect wave treatments at the outlet boundaries. Table 3 shows a comparison of the amplitudes of the 1^{st} harmonics between the current simulations and other CFD codes reported in [12]. Using the experiments as a reference for the comparison it is evident that the `CN0.9+Euler` scheme is superior than the `Euler` scheme. With respect to results from other CFD codes the percentage errors in the present simulations using `CN0.9+Euler` are approximately at the same level as reported by `UNIZAG-FSB/swenseFoam` and `FORCE/StarCCM+`. While these results are very encouraging for a further study of the `CN0.9+Euler` scheme it shall be stressed that the use of the `CN0.9+Euler` scheme combination is not strictly unambiguous

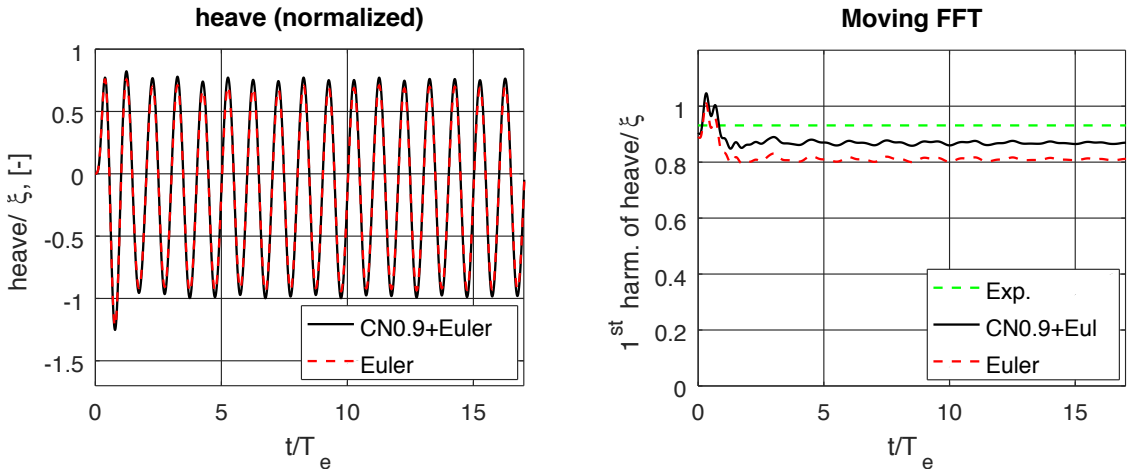

Figure 6: Normalized heave motion (left) and the corresponding results of the moving FFT (right) using a window size of $1T_e$. Heave is normalized by wave amplitude $\xi = H/2$. The time step size is $T_e/\Delta t \approx 187$.

due to the uncertainty on how to select properly the distribution of c_o. Moreover there are other temporal schemes which may perhaps produce results at least at a similar level of accuracy and performance without introducing ambiguous parameters as with the `CN0.9+Euler` scheme. The major barrier in OpenFOAM, however, is that MULES which is the dominating solver for the transport equation of the phase fraction field currently only supports the `Euler` and `CrankNicolson` schemes for the reasons that these two schemes can keep the α-field bounded between 0 and 1.

5 CONCLUSIONS

The results of the 2D wave propagation tests and the 3D seakeeping tests show that OpenFOAM has the capacity to simulate regular head wave conditions reasonably well. Since the test condition presented in this paper only covers one wave condition there is a need for more tests on a larger variety of wave and test conditions in order to draw a firm conclusion on whether or not OpenFOAM can be applied satisfactory in a practical engineering application. Already the results presented here reveal some weaknesses and limitations related to the stability of the available 2^{nd}-order `CrankNicolson` scheme and the accuracy of wave probing tools. The modified `CrankNicolson` scheme provides a better accuracy at an increasing value of c_o, Fig. 4. When c_o is too large, however, the simulation may become unstable. Here it is shown that the stability can be restored by reducing c_o value only at local regions where the cell quality or the flow are considered critical. Without reducing the c_o value globally waves are allowed to propagate more accurately toward the ship which helps to improve significantly the overall accuracy of the predicted ship motions.

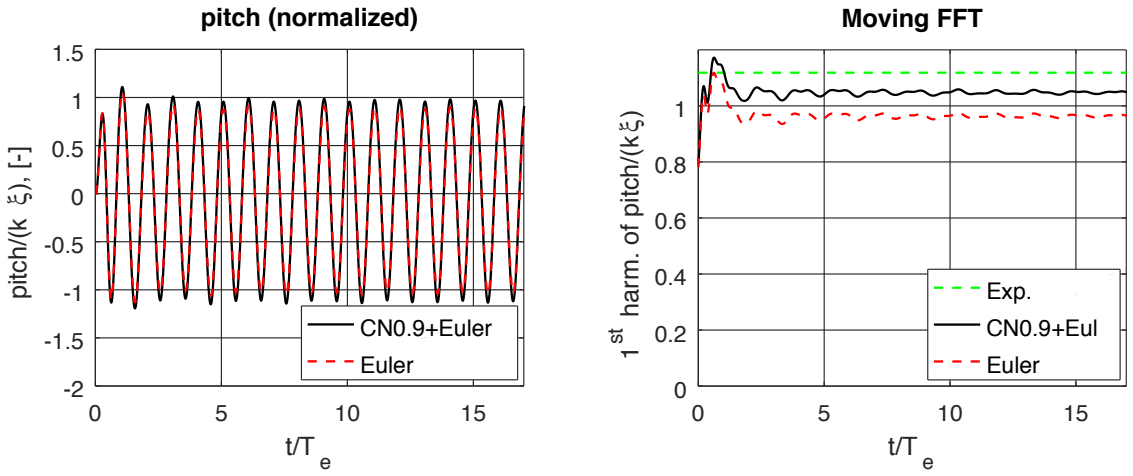

Figure 7: Normalized pitch motion (left) and the corresponding results of the moving FFT (right) using a window size of $1T_e$. Pitch is normalized by $k\xi$, where k is the wave number and ξ is the wave amplitude $\xi = H/2$. The time step size is $T_e/\Delta t \approx 187$.

Table 1: The seakeeping case used in this work c.f. [12]. Wave crest is at FP at $t = 0$ s. Heave motion normalized by wave amplitude $\xi = H/2$, pitch angle by $k\xi$, forces by $0.5\rho U^2 S_0$. Sinkage is positive upwards. Trim is positive bow up.

Hull data (model scale) KCS Model 2, `Case-2.10`	Length btw. perpendiculars, Lpp, [m]	6.0702
	Maximum beam of waterline, B [m]	0.8498
	Draft, [m]	0.2850
	Displacement volume, [m^3]	0.9571
	LCB, [%Lpp, fwd+]	-1.4800
	Vert. CoG, [m, from keel]	0.3780
	Moment of Inertia, [Kxx/B]	0.4000
	Moment of Inertia, [Kyy/Lpp, Kzz/Lpp]	0.2520
Test condition (Towed in regular head wave)	Degree-of-freedom,	heave & pitch
	Towing speed, [m/s]	2.0170
	Wave length, λ [m]	11.840
	Wave height, H [m]	0.1960
	Froude number, F_r	0.2610
	Reynolds number, R_e	$1.074 \cdot 10^7$
	Water depth, d [m]	5.5
Derived wave properties based on [5]	Period, T [s]	2.7581
	Wave frequency, ω [rad/s]	2.2781
	Encounter frequency, ω_e [rad/s]	3.3484
	Wave number, k [1/m]	0.5307
	kd [-]	2.9186
	H/λ [-]	0.0165
	Phase velocity, c [m/s]	4.2929
Experimental results c.f. [12] ampl. $(0^{th}, 1^{st}, 2^{nd}, 3^{rd}, 4^{th})$ phase $(1^{st}, 2^{nd}, 3^{rd}, 4^{th})$	Wave, ampl. [/Lpp] 1^{st}-harm. 0.01611	
	Thrust Coeff., CT, ampl. [10^3], phase [rad]	
	ampl. (10.842, 25.101, 0.655, 0.581, 0.595)	
	phase (-0.3585, 0.0890, 1.5306, -0.8437)	
	Heave, ampl. [/ξ], phase [rad]	
	ampl. (-0.2005, 0.9312, 0.0206, 0.0013, 0.0002)	
	phase (-1.7205, 1.3007, 1.6636, 0.5488)	
	Pitch, ampl. [/($k\xi$)], phase [rad]	
	ampl. (-0.0562, 1.1185, 0.0379, 0.0017, 0.0003)	
	phase (-0.5811, 2.1387, 0.8706, -0.3979)	

Table 2: The description of the FVM schemes used in this work

FVM Scheme (name in OpenFOAM)	Description
`Euler`	1^{st}-order implicit backward Euler scheme
`CrankNicolson` c_o	Modified Crank-Nicolson with off-center coefficient c_o
`backward`	2^{nd}-order three-time-steps backward differencing scheme
`vanLeer01`	2^{nd}-order scheme with van Leer's limiter bounded between $[0, 1]$
`interfaceCompression`	High-resolution scheme for the artificial compression term
`linear`	2^{nd}-order central differencing scheme (linear interpolation)
`linear correct`	Linear interpolation with explicit non-orthogonal correction
`linearUpwind`	2^{nd}-order upwind scheme
Spatial/interpolation schemes	used in all simulations
Divergence operator	$\nabla \cdot (\alpha \boldsymbol{u})$ `vanLeer01` $\nabla \cdot [\alpha(1-\alpha)\boldsymbol{u}_r]$ `interfaceCompression` $\nabla \cdot (\rho \boldsymbol{u})\boldsymbol{u}^T$ `linearUpwind`
Laplacian operator	`linear corrected`
Gradient operator	`linear`
Cell-to-face interpolation	`linear`

Table 3: Amplitudes of the 1^{st} harmonic of normalized heave (z/ξ) and pitch ($\theta/(k\xi)$) motions c.f. [12]. The experimental data (EFD) is from the towing tests performed at FORCE Technology. The errors in percent are computed as $100(S - EFD)/EFD$, where S are the simulated results and EFD are the experiments.

KCS CASE 2.10-C5	heave, z/ξ	Err. [%]	pitch, $\theta/(k\xi)$	Err. [%]
EFD	0.9312	-	1.1185	-
Present Simulations/`CN0.9+Euler`	0.866	-7.00	1.050	-6.12
Present Simulations/`Euler`	0.808	-13.23	0.962	-13.99
FORCE/StarCCM+	0.874	-6.14	1.084	-3.08
HSVA/FreSCO+	0.899	-3.46	1.087	-2.81
IIHR/CFDShip-Iowa v.4.5	0.917	-1.52	1.090	-2.55
KRISO/WAVIS	0.741	-20.43	0.886	-20.79
MARIC/FINEMarine31_3	0.944	1.37	1.163	3.98
NMRI/NAGISA	0.925	-0.67	1.047	-6.39
NUMECA/ISISCFD	0.898	-3.57	1.080	-3.44
UM/OF23x	0.901	-3.24	1.078	-3.62
UNIZAG-FSB/swenseFoam, Grid#1	0.876	-5.93	1.038	-7.20

REFERENCES

[1] The OpenFOAM Foundation Ltd, www.openfoam.org

[2] Weller, H.G.; Tabor, G.; Jasak, H. and Fureby, C.: A Tensorial Approach to Computational Continuum Mechanics using Object Orientated Techniques. *J. Comput. Phys.* 12, 6 (Nov. 1998) 620–631. (doi:10.1063/1.168744)

[3] Jacobsen, N. G.; Fuhrman, D. R. and Fredse, J.: A wave generation toolbox for the open-source CFD library: OpenFoam. *Int. J. Numer. Meth. Fluids* (2012), **70**:1073–1088. doi:10.1002/fld.2726

[4] Monroy, C.; Seng, S. and Malenica, Š.: Développements et validation de l'outil CFD OpenFOAM pour le calcul de tenue à la mer. 15 èmes Journées de l'Hydrodynamique, (22-24 Nov. 2016), Brest.

[5] Fenton, J. D.: The numerical solution of steady water wave problems. *Computers & Geosciences* 14, 3 (1988), 357–368, ISSN 0098–3004, http://dx.doi.org/10.1016/0098-3004(88)90066-0.

[6] Hirt, C.W and Nichols, B.D.: Volume of fluid (VOF) method for the dynamics of free boundaries. *J. Comput. Phys.* 39, 1 (Jan. 1981), 201–225.

[7] Issa, R. I.: Solution of the implicitly discretised fluid flow equations by operator-splitting. *J. Comput. Phys.* 62, 1 (Jan. 1986), 40-65. DOI=10.1016/0021-9991(86)90099-9 http://dx.doi.org/10.1016/0021-9991(86)90099-9

[8] Damian, S. M.: An Extended Mixture Model for the Simultaneous Treatment of Short and Long Scale Interfaces. PhD thesis, Universidad Nacional del Litoral, Santa Fe, Argentina, (2013)

[9] Deshpande, S. S; Anumolu, L. and Trujillo, M. F .: Evaluating the performance of the two-phase flow solver interFoam. *Comput. Sci. Disc.* (2012) 5 014016

[10] Seng, S.; Pedersen, P. T. (Supervisor) and Jensen, J. J. (Supervisor).: Slamming And Whipping Analysis Of Ships. PhD thesis, Technical University of Denmark, Lyngby, Denmark, (2012)

[11] Simonsen, C.; Otzen, J. and Stern, F.: EFD and CFD for KCS heaving and pitching in regular head waves. In Proc. 27th Symp. Naval Hydrodynamics, Seoul, Korea, 2008.

[12] Tokyo 2015: A Workshop on CFD in Ship Hydrodynamics, Dec. 2-4, 2015, www.t2015.nmri.go.jp