Computation of Free-Surface Viscous Flows around Self-propelled Ships with the Help of Sliding Grids

Michel Visonneau, LHEEA ECN/CNRS, Nantes/France, michel.visonneau@ec-nantes.fr
Patrick Queutey, LHEEA ECN/CNRS, Nantes/France, patrick.queutey@ec-nantes.fr
Gan Bo Deng, LHEEA ECN/CNRS, Nantes/France, ganbo.deng@ec-nantes.fr
Jeroen Wackers, LHEEA ECN/CNRS, Nantes/France, jeroen.wackers@ec-nantes.fr
Emmanuel Guilmineau, LHEEA ECN/CNRS, Nantes/France, emmanuel.guilmineau@ec-nantes.fr
Alban Leroyer, LHEEA ECN/CNRS, Nantes/France, alban.leroyer@ec-nantes.fr
Benoit Mallol, NUMECA Int., Brussels/Belgium, benoit.mallol@numeca.be

Abstract

Accurate prediction for a self-propelled ship is a challenging task for CFD computation. While turbulence modeling is a crucial issue, more physical modeling is involved in a propeller simulation: transition, cavitation, turbulence, ventilation, etc. More advanced numerical technique is also required to capture very small scale flow motion such as tip vortex. The ISIS-CFD RANSE code has been fully validated for resistance and wake flow prediction using advanced turbulence modeling. An adaptive mesh refinement technique with different refinement is available and a sliding grid approach is recently implemented. This paper is devoted to the validation of the ISIS-CFD code for self propulsion computation. The test case chosen is the well known KCS test case proposed for the Tokyo 2005 and Gothenburg 2010 workshop.

1. Introduction

Propeller-hull interaction is an important topic for ship design. Accurate prediction for a self-propelled ship is a challenging task for CFD computation. It requires not only an accurate prediction of resistance and wake flow, but also a good prediction for the propeller. While turbulence modeling for the bilge vortex is a crucial issue to ensure an accurate prediction for the wake flow, more physical modeling is involved in a propeller simulation: transition, cavitation, turbulence, ventilation, etc. More advanced numerical technique is also required to capture very small scale flow motion such as tip vortex.

Currently, one of the main goals of the research team is the simulation of ship propellers in extreme operating conditions, with accurate modelling of all the physics involved. This requires the capability to simulate a rotating propeller behind a ship hull, combined with effects of free-surface deformation, ventilation, and cavitation. These capabilities are being developed for inclusion in ISIS-CFD, *Duvigneau et al. (2003)*, *Queutey and Visonneau (2007)*, the unstructured finite-volume flow solver. This flow solver code has been fully validated for resistance and wake flow prediction using advanced turbulence models such as the Explicit Algebraic Stress Model with rotation correction.

An essential building block for the simulations including propellers is a sliding grid technique, which allows a part of the grid (containing the propeller) to rotate within the main part of the grid, while keeping a connection between the two parts, Fig.1. Also, since many of the phenomena to be studied originate from highly localised low pressure zones, the accurate simulation of these phenomena can be obtained by automatic adaptive grid refinement.

However, the sliding grid approach and the adaptive grid refinement have to be general enough to work together. We have recently developed a sliding grid capability, that has been specifically constructed to work together with our existing grid refinement method, *Wackers et al. (2010a,b)*. Both techniques are powerful enough to deal with the fully unstructured hexahedral grids that we generally use.

Sliding grid methods can be constructed for unstructured grids, these procedures are far more

complicated than for structured grid solvers. Among others, the connection between the two subdomains of the grid has to be reconstructed often, as it does not follow a regular pattern. Also, it is not easy to ensure flux conservation over the interface. We have chosen a connection between the domains that does not explicitly guarantee conservation; it is based on connectivities between the cells and faces on the interface that mimic as closely as possible the connectivities for all other cells in ISIS-CFD.

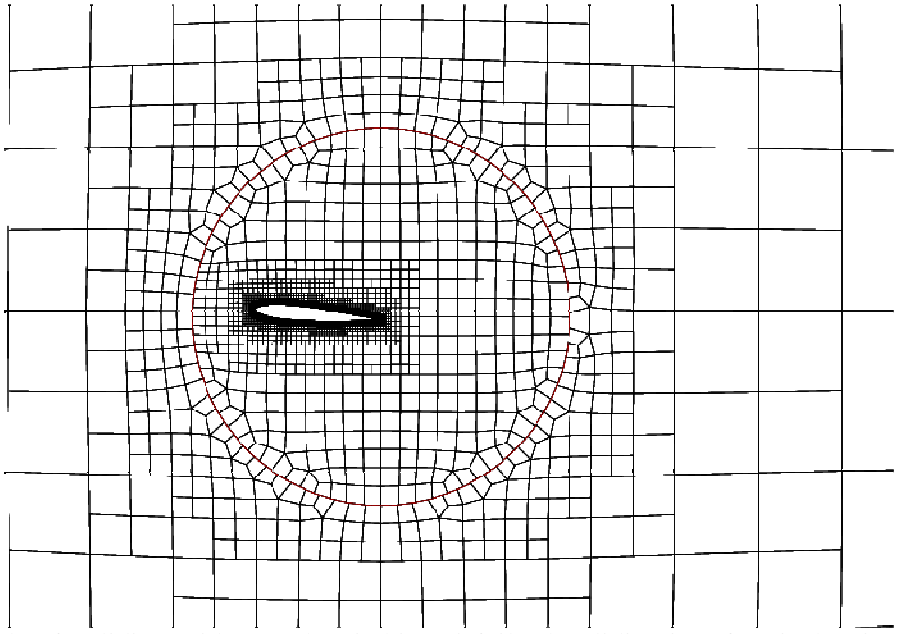

Fig.1: Example of a sliding grid around a pitching airfoil. The sliding interface is the circle between the two non-matching parts of the grid

An adaptive mesh refinement technique with different refinement criterion based on pressure Hessian and free-surface volume fraction is available. The method works in parallel with the sliding grid approach and the different sub-domains can be distributed arbitrarily over the processors. The combination with adaptive grid refinement is obtained by treating the sliding faces like standard boundary faces, when refining. The cell sizes are not explicitly synchronized over the sliding faces, instead the continuity of the refinement criterion over the interface guarantees smoothly varying cell sizes.

2. The ISIS-CFD flow solver

ISIS-CFD, available as a part of the FINE^{TM}/Marine computing suite, is an incompressible unsteady Reynolds averaged Navier-Stokes (RANS) solver, *Duvigneau et al. (2003)*, *Queutey and Visonneau (2007)*. The solver is based on the finite volume method to build the spatial discretisation of the transport equations. Pressure-velocity coupling is obtained through a Rhie & Chow SIMPLE type method: in each time step, the velocity updates come from the momentum equations and the pressure is given by the mass conservation law, transformed into a pressure equation.

The discretisation is face-based. While all unknown state variables are cell-centred, the systems of equations used in the implicit time stepping procedure are constructed face by face. Fluxes are computed in a loop over the faces and the contribution of each face is then added to the two cells next to the face. This technique poses no specific requirements on the topology of the cells. Therefore, the grids can be completely unstructured; cells with an arbitrary number of arbitrarily-shaped faces are accepted.

Free-surface flow is simulated with a multi-phase flow approach: the water surface is captured with a conservation equation for the volume fraction of water, discretised with specific compressive

discretisation schemes, *Queutey and Visonneau (2007)*. Furthermore, the method features sophisticated turbulence models, *Duvigneau et al. (2003),* and 6 DOF motion simulation for free moving ships, *Leroyer and Visonneau (2005).*

Parallelisation is based on domain decomposition. The grid is divided into different partitions; these partitions contain the cells. The interface faces on the boundaries between the partitions are shared between the partitions; information on these faces is exchanged with the MPI (Message Passing Interface) protocol.

2.1. Reconstruction on the faces

The face fluxes are computed from the quantities in the cell centres reconstructed to the faces, Fig.2. For the diffusive fluxes and the coefficients in the pressure equation, the quantities on a face and the normal derivatives are computed with central schemes using the L and R cell centre states; if these centres are not aligned with the face normal, then non-orthogonal corrections are added which use the gradients computed in the cell centres. For the convective fluxes, we use the AVLSMART scheme, *Przulj and Basara (2001)*, in the NVD context for unstructured, where limited schemes are constructed based on a weighted blending of the central difference scheme and an extrapolation using the gradient in the upwind cell. The reconstructions are detailed in *Queutey and Visonneau (2007).*

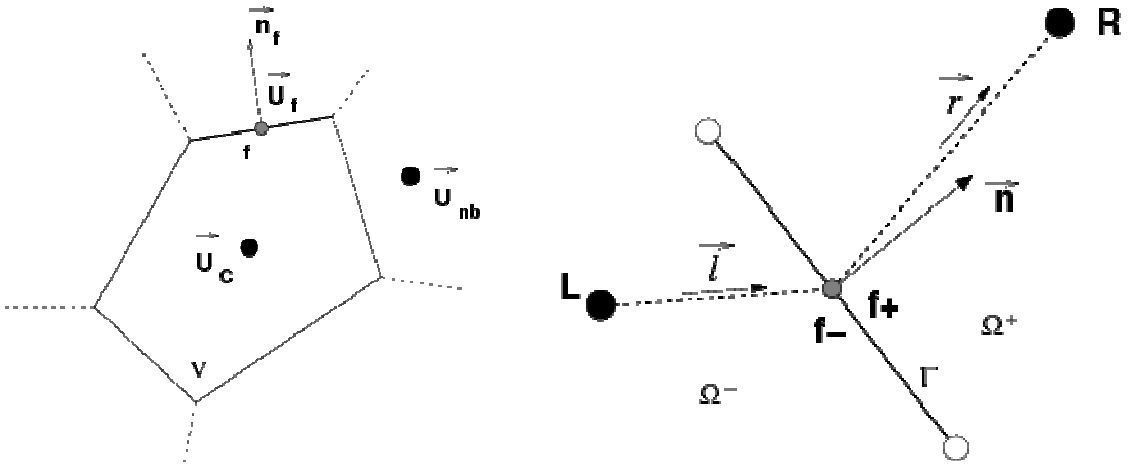

Fig.2: Cell, faces and neighbours (left), reconstruction on a face (right)

For the following discussion, the essential point is that all fluxes on a face can be constructed from the states and gradients in the centres of its two neighbour cells, plus the positions of these centres. In our domain decomposition approach for parallel computing, this neighbour cell information is the only thing which is exchanged over the interface faces.

2.2. Sliding interface implementation

To compute the fluxes over the sliding interface, we need to establish connections between cells on the two sides of the interface. The procedure to connect these cells is performed at each time step in order to account for the rotation of the two subdomains with respect to each other. This procedure is chosen to remain as close as possible to what is done for standard cells. Thus, no specific interpolations are used. Instead, for a cell and face on the interface, we search the cell centre (in the other subdomain) which best matches the face. This cell is then used as neighbour cell for a flux computation exactly like in section 2.1.

The matching neighbour steps are searched in three steps, Fig.3:

1. A temporary ‘ghost’ point is constructed on the outside of each sliding face. This point is the mirror image of the inside neighbour cell centre, except near sharp corners of the sliding interface where the normal vector to the face is used. Ghost points are constructed on each

side of the interface, for the two subdomains (Fig.3 shows only the point for the left subdomain). The ghost points are not used for interpolation, only for the remainder of the search.

2. The current position of the sliding faces is gathered over all partitions to form a global table. Then, in each partition, a search algorithm is used to find the global sliding face closest to each local ghost point.
3. The inside neighbour cells of the faces found are used as outside neighbours for the local sliding faces. If the neighbour is on another processor, an MPI communication is established just like the one for the normal domain decomposition. If the two cells are on the same processor, the communication is performed locally. As opposed to the normal domain decomposition, a cell on a sliding interface may be a neighbour for more than one cell or for none at all.

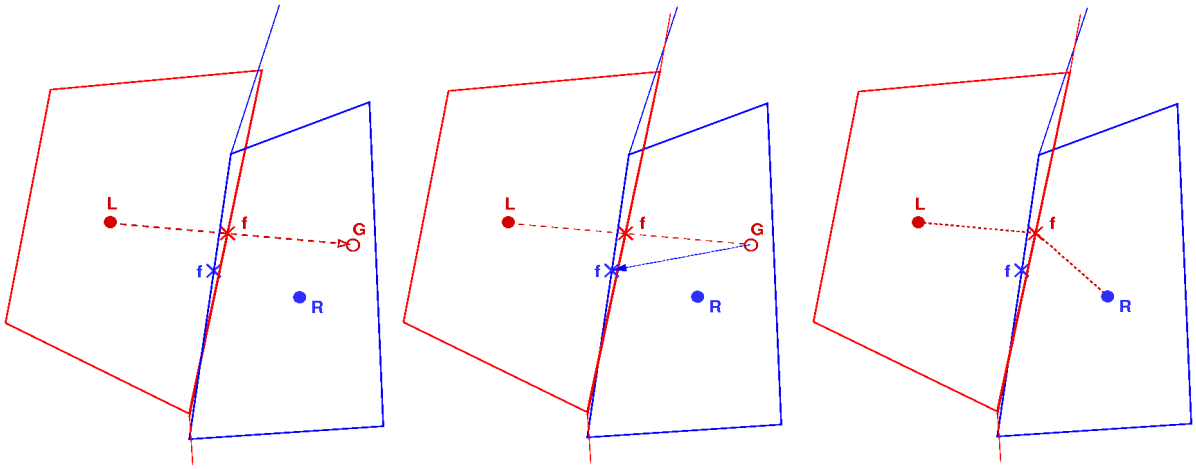

Fig.3: Sliding interface: construction of ghost points (left), searching the global faces (centre), the new neighbour cell (right)

As one sees, we do not explicitly split the grid in two parts by assigning a number of processors to each subdomain and partitioning each subdomain separately. Instead, the grid can be arbitrarily spread over the processors; a 'colour' is assigned to each cell to indicate to which subdomain it belongs. This gives the flexibility to run the code on a single processor and also to perform grid refinement; when a part of the grid is refined, the balance would be lost if each subdomain were assigned to a constant number of processors. With our approach, we can redistribute freely.

2.3. Automatic grid refinement

The automatic adaptive grid refinement technique included in the solver ISIS-CFD is for example described in *Wackers et al. (2010a,b)*, *Wackers et al. (2011)*, The technique is meant to be used for all the different applications of the flow solver and has therefore been made as general as possible. The method supports the isotropic and anisotropic refinement of unstructured hexahedral meshes, i.e. cells can be refined by dividing them in all directions or in one direction only. Earlier refinements can be undone in order to adapt the grid to unsteady problems. The refinement criterion, which indicates where the grid must be refined, can be modified very easily; different refinement criteria have already been tested, *Wackers et al. (2010b)*. And finally, the grid refinement is fully parallel and includes an automatic dynamic load balancing in order to redistribute the refined grid over the processors when some partitions have been refined more than the others.

When computing the interaction of a propeller with the water surface, the volume fraction equation which gives the surface position needs to be accurately resolved. The precise computation of the vortical structures around the propeller is also of prime importance; our preliminary research suggests for example that the onset of ventilation is largely determined by the minimum pressure on the propeller blades. Thus, the grid must be refined both at the water surface and below the surface, in

order to get good accuracy. Therefore, we choose a refinement criterion which is a combination of two sensors. The first creates anisotropic grid refinement around the free surface. The second is based on the Hessian matrix of second derivatives of the pressure, which is similar to criteria being used for tetrahedral grid refinement, *Alauzet and Loseille (2010)*. This second criterion detects the presence of for example vortices. The final criterion is taken as the (approximate) maximum of these two sensors. A more complete description of the combined criterion is given in *Wackers et al. (2012)*.

2.4. Refinement and sliding grid coupling

Since the coupling between sliding faces is recomputed before each time step, no coupling information needs to be preserved when the grid is refined. Therefore, the coupling between the two techniques is relatively straightforward. However, several points deserve attention.

The first is the refinement of the sliding faces. Unlike the interface faces between partitions, which require a complex refinement procedure to preserve the connection to the faces on the other processor, the coupling between sliding faces is not kept during refinement. Therefore, the sliding faces are refined independently, just like boundary faces such as wall, inflow, or outflow faces. There is no explicit guarantee that the resulting refined cells on the two sides of the sliding interface have the same size. However, the refinement criterion which imposes the cell sizes is computed from the flow field, which is smooth over the interface. Thus, we may also expect a smooth variation of the cell size over the sliding interface.

A second point is the dynamic load balancing. As a part of this procedure, the grid is repartitioned using *PaRMeTiS,* http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview. This tool is a graph partitioner that searches a balanced distribution of the nodes and a minimum number of edges cut; its input graph has the cell centres as nodes and the cell – neighbour cell connections as edges. To construct the graph, no edges are created for the sliding faces: all connections via sliding faces are ignored. Thus, ParMeTiS naturally tends to put the interfaces between partitions on the sliding faces, as this costs nothing in terms of edges cut. This is an advantage for the convergence of the linear solvers in ISIS-CFD, since data on both sliding faces and interface faces between processors are updated in our linear solvers, but not in the innermost loop. Thus, faster convergence is obtained if these faces coincide, as this keeps their total number low.

3. Simulations

3.1. The INSEAN E779A propeller

In a first step, computations have been performed to assess the capacity of ISIS-CFD in open-water simulations for ship propellers with hexahedral grids as generated by HEXPRESS. The test case is the INSEAN E779A propeller, *Salvatore (2007)*. The tests are performed in the INSEAN middle towing tank, with the propeller axis at 1.5 times the propeller diameter below the free surface. However, as is customary, in most of the present computations the free surface is not taken into account.

3.1.1. Grid convergence

A grid convergence study has been conducted for the computational conditions listed in Table 1.

Table 1: Computational conditions

Propeller diameter D	0.2272727 m
Reference chord	0.0870 m
Rotational speed n	11.7881 rps
Water viscosity μ	$1.1099 \cdot 10^{-3}$ kg/(ms)
Water density ρ	1000 kg/m^3

The non-dimensional coefficients to be considered are written in Table 2 and the advance coefficient ratio of the convergence study is J=0.895. The force Fx and the moment Mx are computed on the propeller blades only. Note that this is not exactly the same as for the experiments, where the force on the blades and on the hub is measured and the force on the hub without blades is subtracted from this value.

Table 2: Non-dimensional coefficients

Advance ratio	Thrust	Torque	Open-water efficiency
$J = \frac{U}{nD}$	$K_T = \frac{F_X}{\rho n^2 D^4}$	$K_Q = \frac{M_X}{\rho n^2 D^5}$	$\eta = \frac{UF_X}{2\pi M_X} = \frac{nJK_T}{2\pi K_Q}$

A series of mesh involving a single rotating is generated with HEXPRESS. The computational domain is a cylinder from X = −1.25D to X = 4.0D with a diameter of 2.942D. Boundary conditions are a law of the wall on the propeller blades and hub, far field Dirichlet condition on the inflow and side faces, prescribed pressure on the outflow wall. Five meshes are created with a number of cells of 775k for mesh 0, 1.87M, 3.86M, 6.97M, and 11.4M for mesh 4. In all cases, the boundary layer grid is made with a prescribed first-cell thickness of 0.000152533. Note that between mesh 2 and 3, the number of layers in the boundary grid decreases from 10 to 8. Two images of a typical mesh can be found in Fig.4.

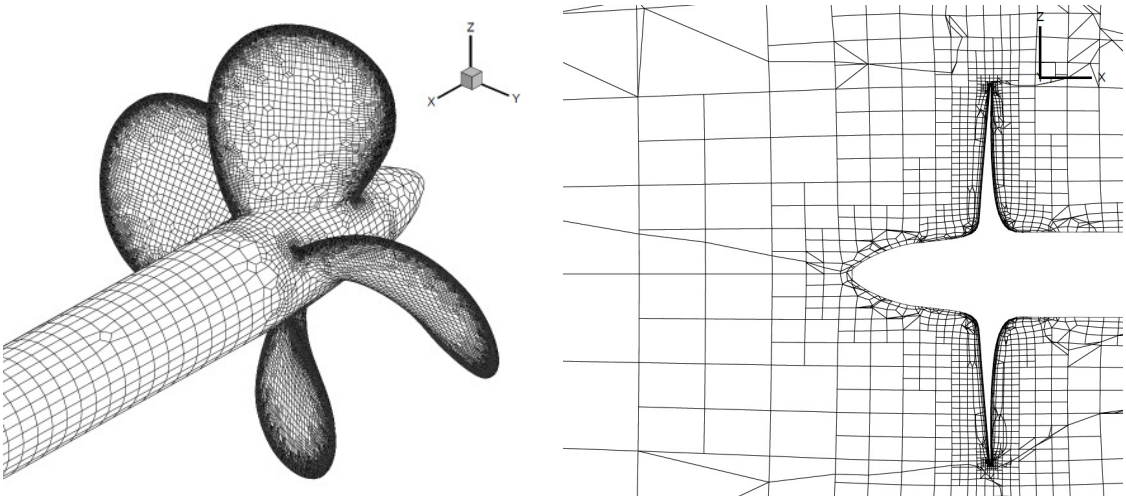

Fig.4: Surface grid (left) and Y = 0 cut for mesh 0 (right)

The full propeller is modelled. Because the geometry used is the one of the real model propeller which is not perfectly symmetric, the mesh on the different blades is not identical. Apart from the classical convergence of the forces, we study the convergence of the difference in force between the blades:

$$\Delta F_{X,blade} = 4\ (\ \max(F_{X,i}) - \min(F_{X,i})\ ) \quad i = 1 \cdots 4,$$
$$\text{and } \Delta M_{X,blade} = 4\ (\ \max(M_{X,i}) - \min M_{X,i})\ ) \quad i = 1 \cdots 4,$$

where $F_{X,i}$ and $M_{X,i}$ are the force and moment on blade i.

Since the computations are performed in an earth-fixed reference frame, they are unsteady even when converged. The solution (computed by time integrating the unsteady flow equations) continues to fluctuate in time. Thus, the forces are averaged over the last 180 time steps (last computed period). Furthermore, the strength of the fluctuations is computed by taking the standard deviation of the signal:

$$\tilde{F}_{X,t} = \left(\frac{1}{N} \sum_{end-N}^{end} \left(F_X(t) - F_X \right)^2 \right)^{1/2}$$

Fig.5 shows the convergence of these quantities, normalised (for better or worse) by the F_X force and the M_X moment computed on grid 4. Table 3 gives the convergence of the main force and moment coefficients. Roughly speaking, apart from grid 3 where the boundary layer grid jumped from 10 to 8 cells thickness, the convergence of KT and KQ is monotone. If we assume that the numerical uncertainty due to the mesh size can be estimated as the difference with grid 4, plus a little extra, then this uncertainty is about 3% for grid 1 (1.87M cells).

Thus, we (somewhat optimistically) find a numerical uncertainty of about 4% for grid 1. The difference between the experimental value for F_X and grid 4 is larger, thus we conclude that modelling errors are larger than numerical errors from this grid on.

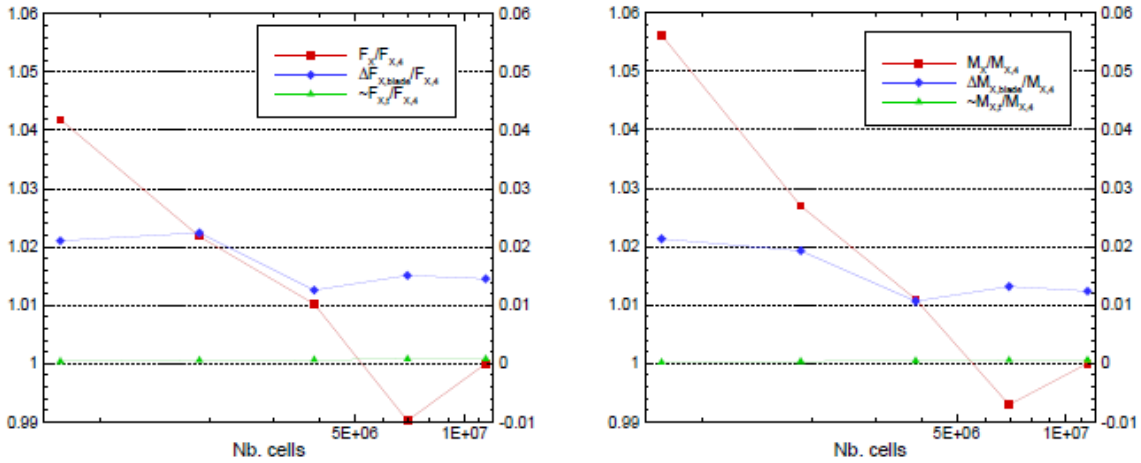

Fig. 5: Convergence of force (left) and moment (right) coefficients. The force and moment are shown on the left axes, the dependence on time and the difference between blades on the right axes

Table 3: Grid convergence, J = 0.895

Mesh	K_T	Diff. M4	K_Q	Diff. M4	η	Diff.M4
0	0.14685	4.19%	0.031966	5.62%	7.7139	-1.35 %
1	0.14403	2.19%	0.031082	2.70%	7.7809	-0.50 %
2	0.14240	1.03%	0.030599	1.10%	7.8140	-0.08 %
3	0.13958	-0.97%	0.030052	-0.71%	7.7987	-0.27 %
4	0.14095		0.030266		7.8202	
Exp.	0.150	6.42%	0.0294	-2.86%	8.567	9.55 %

3.1.2. Simulation with sliding grids at J=0.895

A simulation at J=0.895 has been conducted to validate the sliding grid methodology with a grid density similar to the grid density obtained with grid 1. Fig.6 presents a cut Y=0 in the grid with the sliding interface outlined. Inside the cylinder limited by the sliding interface the grid is rotating and outside it is fixed. The boundary conditions and the numerical settings used are the same as those used for the case with a monolithic rotating grid (previous sections).

Table 4 contains the results of the force and moment of the simulation without and with sliding grids. It is also used to compare the difference in force between the blades, $\Delta f_{X,blade}$,and the strength of the fluctuations in time computed by taking the standard deviation of the signal $\tilde{f}_{X,t}$.

Table 4: Forces, deviations and differences between blades (1G: one grid, SG: with sliding grids)

case	F_X(N)	K_T	$\Delta F_{X,blade}/F_X$	$\tilde{F}_{X,t}/F_X$	M_X(Nm)	K_Q	$\Delta M_{X,blade}/M_X$	$\tilde{M}_{X,t}/M_X$
1G	53.3987	0.1440	0.022	$5.8\ 10^{-4}$	-2.6190	0.0311	0.019	$3.7\ 10^{-4}$
SG	53.4569	0.1442	0.032	$1.6\ 10^{-3}$	-2.6115	0.0310	0.024	$1.2\ 10^{-3}$

The mean forces between the two computations are close together with a relative difference of about 0.1% on the force and about 0.3% on the moment. The differences between the blades are in the same order between 2% and 3% but the fluctuations in time are more pronounced and multiplied by a factor 3 with the computation involving the sliding grid approach. The reason given for this increase in fluctuations in time is the change of connection to the neighbouring cells of the sliding interface. This change occurs from time to time between the rotating cells and the fixed cells. Even if the interpolation is second order accurate, it seems as if the grid is changing between two time steps when a connection changes between these two time steps. One possible option is to have a smooth in time and still accurate interpolation of the neighbouring cells independent of the mesh. This has already been implemented for the computation of the first and second order derivatives of the pressure for adaptive grid refinement, *Wackers et al (2012)*, on the basis of third-order least-squares approximations.

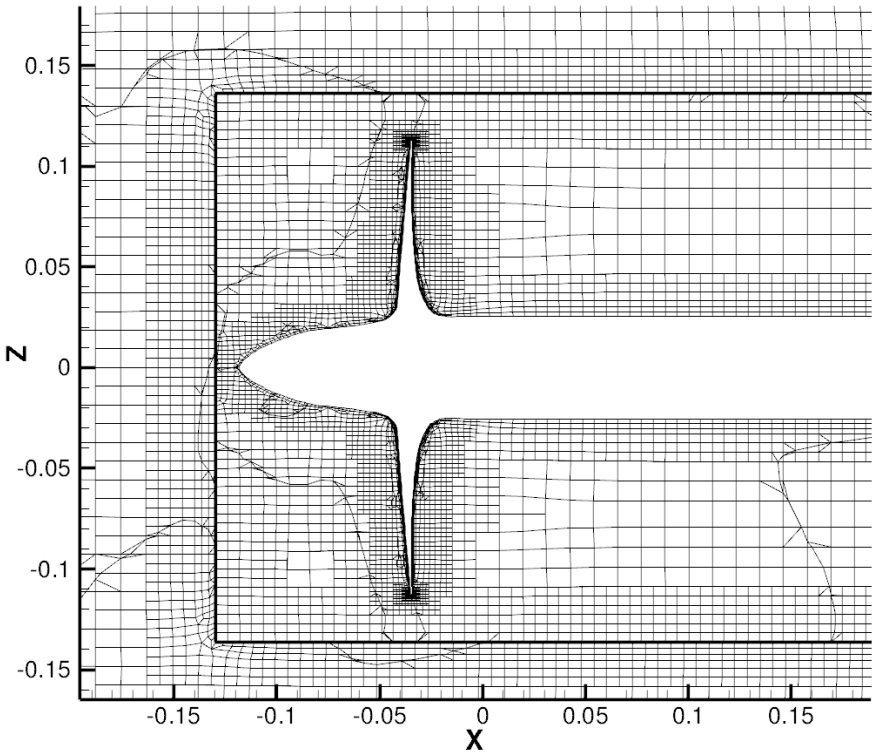

Fig.6: Y = 0 cut for mesh 1 with sliding interface (thick line)

3.1.4. Simulation with adaptive grid refinement at J=0.880

The advance coefficient J=0.880 corresponding to a rotational speed *n*=25rps with advance velocity *U*=5m/s has been selected to study the effect of adaptive grid refinement. This specific point is retained since PIV measurements are available, *Salvatore (2007)*. The pressure Hessian-based criterion is used to control the grid adaptation since it reacts to high pressure variations associated with the intense vortical structures, *Wackers et al (2011,2012)*. Here, the targeted vortical structures correspond to the tip vortex on each blade where low pressure peak can induce the cavitating phenomenon.

Only the forces have been compared between the results obtained with the grid 1 and the results obtained with an automatically refined grid starting from the lightest grid 0. The minimum cell for the adaptive grid is set to 0.35mm. Fig.7 compares the iso-surface pressure corresponding to the computed pressure value p=10kPa. The effect of automatic refinement is clearly detected from the shape of the surface pressure. It is interesting to note that grid 1 has about 1.86M cells which is equivalent to the number of cells in the adapted grid; about 1.92M cells at the end of the simulation.

With the automatic refinement the computed F_X force is increased by 5% and the magnitude of the moment M_X by 4%. This is explained with the fact that on coarse or non-adapted meshes, vortex strengths are under-predicted. Thus, we may assume that the suction peak is weaker in the grid 1 than in reality, leading to an under-prediction of both K_T and K_Q.

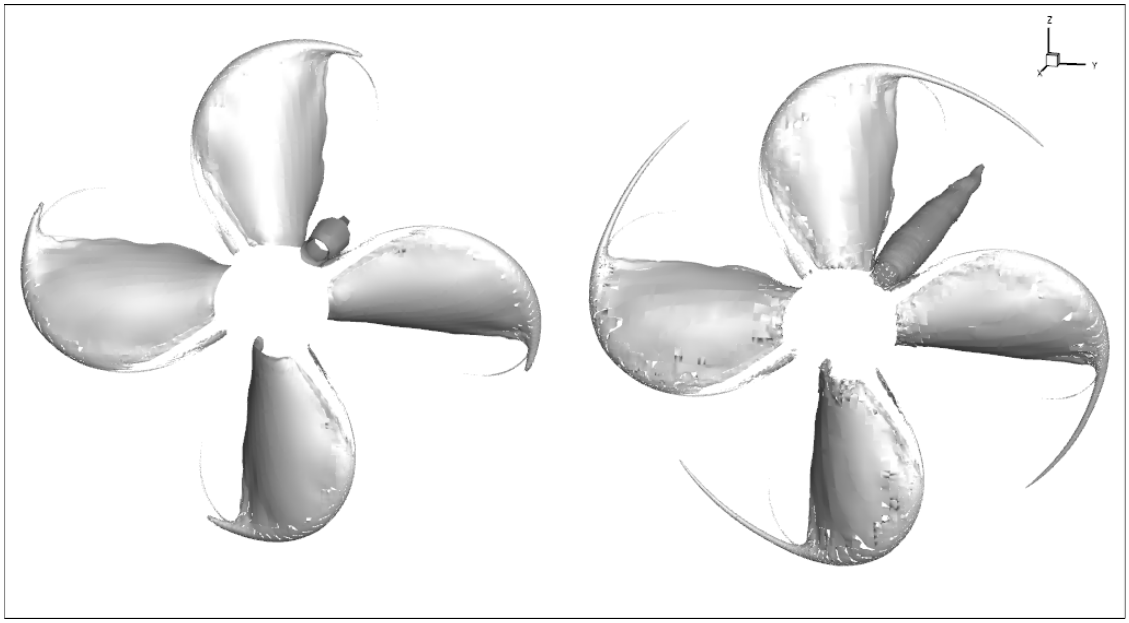

Fig.7: Iso-pressure surface p=10 kPa for J=0.880: from grid 1 (1.86M) on the left; from the adapted grid (1.92M) starting with grid 0 on the right

3.1. The MOERI container ship (KCS)

The KCS tanker experimented by the Korean Institute for Ships and Ocean Engineering (now MOERI) was conceived to provide data for both explication of flow physics and CFD validation for a modern container ship. The present study focuses on the case 2.3a, *Hino (2005),* used for the last Gothenburg workshop. This corresponds to a self propulsion case at model scale. Open water, resistance test, and self-propulsion at ship point in calm water computations are used to evaluate various coefficients.

3.2.1. Mesh generation

At least three different computations are required for a self propulsion computation at ship point in model scale: an open water propeller computation, a resistance computation, and a propulsion computation. A double model computation is also required if the skin friction correction is to be determined by the computation as well. As the purpose of the present computation is to validate the CFD computation, the same value of skin friction correction as in the measurement is applied in the computation. Sliding grid approach is employed for all computations. The same mesh for the propeller is employed both for the open water computation and the propulsion computation. A semi-spherical cap is added to the hub. The upstream and downstream sliding grid interfaces are located at 0.114*D* and 0.249*D* from the centre of the propeller respectively, while the outer sliding grid interface is located at 0.6*D*, *D* being the diameter of the propeller. It contains about 2 million cells to agree with

the grid 1 settings from the convergence analysis on the INSEAN propeller. The same grid for the hull is also employed both for the resistance computation and for the propulsion computation. With appropriate refinement near the bow, the stern and the free-surface, the grid contains about 3.5 million cells. In the resistance computation, the domain containing the propeller is replaced by domain containing a dummy hub. While for the open water computation, an external cylindrical domain is added with the inlet, outlet and external boundaries located at 2*D*, 3.6*D* and 2.4*D* respectively. All meshes have been generated by the hexahedral unstructured grid generator HEXPRESS^{TM}.

3.2.2. Open water computation

Several computations have been performed to obtain the open water characteristics for the propeller. Rate of revolution n=9.5 is fixed for all computation. Propeller advancing speed is adjusted so that the advancing coefficient for different computations are J=0.1, 0.2…, 0.9. Effect of gravity is not taken into account. Zero pressure condition is applied at the outlet, while uniform velocity is applied at other external boundaries. One propeller revolution is performed with 200 time steps with 8 non-linear iterations per time step, to reduce the residual by about 60. The propeller is accelerated to its maximum speed in one revolution. Convergence is achieved after about 4 revolutions. Predicted open water characteristics are compared with measurement results obtained at NMRI and SVA in Fig.8. The propeller size and the rate of revolution of both measurements are different. The setup of the present computation corresponds to the measurement condition in NMRI. The predicted results agree fairly well with the measurement data in spite of relatively low numerical resolution both in space and in time as well as relatively low time convergence.

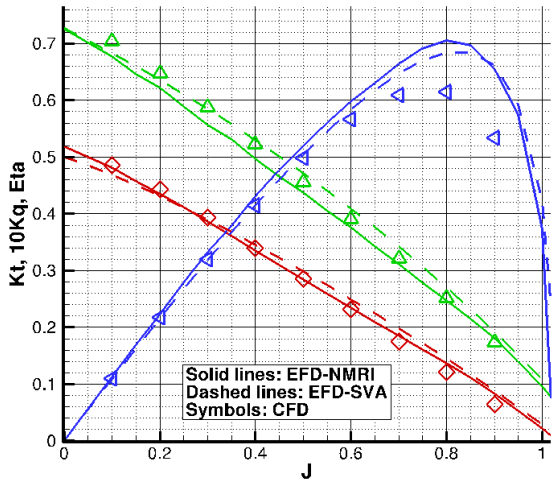

Fig.8: Open-water performance of KCS propeller

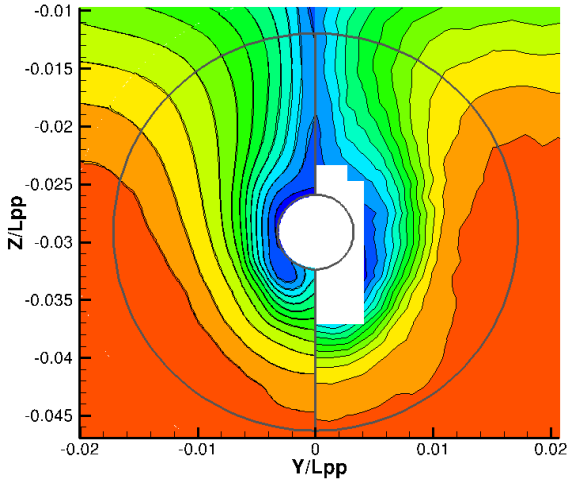

Fig.9: Nominal wake of the KCS container ship at x/Lpp=0.9825: computed (left) and measured (right)

3.2.3. Resistance test computation

Ship position is fixed in the resistance computation with Reynolds number Re=1.36x10^{7} in agreement with the experimental condition. Water density is set to 998.4kg/m³. With ship speed V=2.196m/s, the predicted ship resistance Rt(tow)=79.79N using the K-ω SST model. Among it, friction resistance is 63.63N. Normalized with wetted surface S_0=9.4379m², the resistance coefficient C_t(tow)=3.511x10^{-3}, about 1% smaller than the measurement value (3.55x10^{-3}). The friction resistance coefficient C_f=2.80x10^{-3}, while the ITTC'57friction line gives C_{f0}=2.846x10^{-3}. More difference is observed on the nominal wake coefficient. The predicted value is 0.745, while the measurement value is 0.686. With a higher nominal wake coefficient, it is expected that the thrust of the propeller with the same rate of revolution will be smaller than the measurement value. Higher rate of revolution is expected for numerical simulation under the self-propulsion condition. Fig.9 shows the nominal wake compared with the measurement. Bilge vortex does not seem to exist. Conventional

turbulence model should be capable to predict the wake flow with accuracy. Yet, difference between the computation and the measurement is quite noticeable. Further investigations are required to explain this issue.

3.2.4. Self-propulsion test computation

For the self-propulsion computation, time accurate simulation is performed for the whole transition procedure of a running propeller. Ship position is fixed. The boat is first accelerated to its maximum speed with a frozen propeller until reaching a nearly steady state in about 800 time steps with 4 non-linear iterations per time step. A large time step Δt=0.03s is employed during this period. Then, we switch to a simulation with rotating propeller using the time resolution as for the open water computation, namely Δt=0.000526s, 200 time steps per period for n=9.5rps. It turns out that such procedure is extremely time consuming. Propeller thrust begins to stabilize after about 12 revolutions, namely 1.26 second physical time. However, free surface deformation due to the transition from a non-rotating propeller to a rotating propeller can not be stabilized in such a short time. We continue the computation up to 2 seconds. The ship resistance keeps decreasing from a maximum value of about 91.5N to 89.5N. The cost of this not yet converged computation is about 8 times a resistance test run.

In a self-propulsion computation, following equilibrium is expected: Rt(sp) = T + SFC, where Rt(sp) is the total resistance of the ship in the self-propeller condition, T is the propeller thrust, and SFC is the skin friction correction.

As the objective of the paper is to validate the CFD computation with measurement data, the same value SFC=30.3N used in the measurement at NMRI is employed in the computation. Care must be taken when one determines the propeller thrust. Usually, in the measurement, the forces acting on the whole propulsion device, including the blade and the hub is measured. An additional measurement with a rotating dummy hub only is also measured. The difference between both measurements is then considered as the thrust of the propeller. In the numerical computation, only one computation corresponding to the first measurement is performed. The thrust of the propeller is obtained by integrating forces on the blade of the propeller. Forces acting on the hub of the propeller are considered as part of ship resistance. As the radius of the hub is not constant, the thrust thus computed will depend on pressure reference level. To avoid any ambiguity, a modified hub with constant radius R=45.2mm is employed in the computation. As the surface of the blade is decreased, it is expected that the computed thrust is slightly smaller than the measurement value.

The first computation with n=9.5rps gives ship resistance Rt(sp)=89.3N and propeller thrust T=55.3N. With SFC=30.3N, the unbalance is 3.7N. As expected, the rate of revolution of the propeller needs to be increased to achieve the equilibrium. Based on the first run (Run1), a new run with a rate of revolution n=9.748rps is performed (Run2 not as converged as Run1). Results under self-propulsion condition can be interpolated from these two runs. Details are given in Table 5.

Table 5: Interpolation for self-propulsion point

Quantities	Run1	Run2	Self-propulsion
n (rps)	9.5	9.748	9.67
Rt(sp) (N)	89.3	90	89.77
T (N)	55.3	61.56	59.47
Rt(sp)-(T+SFC) (N)	3.7	-1.86	0
Q (Nm)	2.6	2.84	2.76

Based on these results, the computed and measured self-propulsion parameters are summarized in Table 6. The ways they have been obtained are fully detailed below:

- Thrust coefficient $K_T=T/(\rho n^2D^4)=59.47/(998.4*9.67^2*0.25^4)=0.163$
- Torque coefficient $K_Q=Q/(\rho n^2D^5)=2.76/(998.4*9.67^2*0.25^5)=0.0303$
- Thrust deduction coefficient 1-t = 1-(T+SFC-Rt(tow))/T=1-(59.47+30.3-79.79)/59.47=0.832
- Advance ratio J=0.7233, determined by using the thrust identity method as shown in Fig.10
- Torque coefficient in open water $K_Q(0)=0.03052$ is also determined by using the thrust identity method
- Propeller advance speed Va=JnD=0.7233*9.67*0.25=1.749m/s
- Taylor wake fraction $w_t=(U-Va)/U=(2.196-1.749)/2.196=0.204$
- Effective wake coefficient $1-w_t=0.796$
- Open water efficiency $\eta_0=JK_T/(2\pi K_Q(0))=0.7233*0.163/(2\pi*0.03052)=0.615$
- Relative rotative efficiency $\eta_R=K_Q(0)/K_Q=0.03052/0.0303=1.007$
- Propulsive efficiency $\eta=\eta_0\eta_R(1-t)/(1-w_t)=0.615*1.007*0.832/0.796=0.674$

Table 6: Computed (CFD) and measured (EFD-NMRI) self-propulsion parameters

Self-propulsion parameters	CFD	EFD
Propeller thrust coefficient K_T	0.163	0.170
Propeller torque coefficient K_Q	0.0303	0.0288
Thrust deduction coefficient (1 - t)	0.832	0.853
Effective wake coefficient from thrust identity method (1 - w_T)	0.796	0.792
Propeller open water efficiency η_o	0.615	0.682
Relative rotative efficiency η_R	1.007	1.011
Advance ratio from thrust identity method J, Fig.11	0.723	0.728
Propeller rate of revolution n [rps]	9.67	9.50
Propulsive efficiency η	0.674	0.740

The underestimation of the computed thrust coefficient compared to the measured value is consistent with the fact the computed mean velocity observed at the propeller plane, Fig.9 for the resistance test, is high compared to the mean experimental value. Irrespective of the uncertainty about the not fully converged solution of Run2, we can conclude that the lower predicted thrust is mainly due to the higher wake velocity.

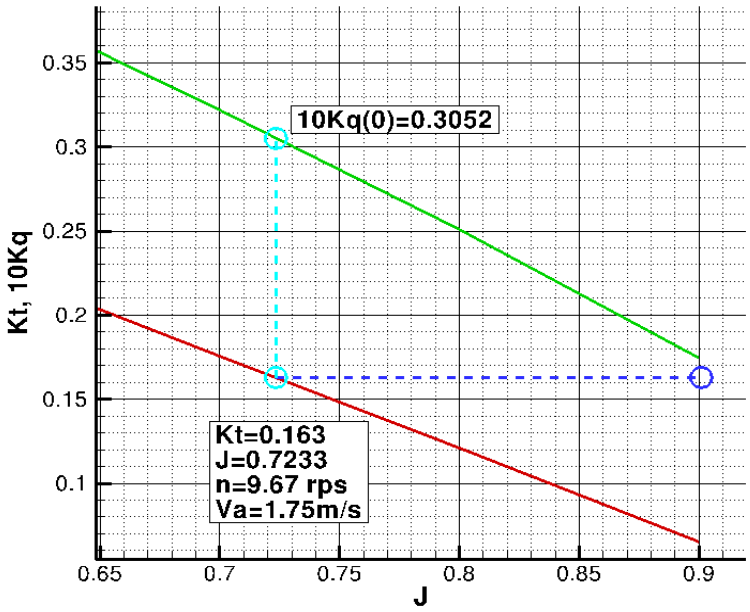

Fig.10: Thrust identity method from computed results

The axial velocity contours and cross-flow vectors are available downstream of the propeller at x/L_{pp}=0.991, Figs.12 and 13. The axial velocity contours, Fig.12, are characterized by two regions inside and outside the propeller disk. Inside the propeller disk, one notice a characteristic asymmetric behavior with a moon crescent-like region of high velocity (between u/U=1.1 and 1.2). The asymmetry of the flow flied is not surprising when considering the computed pressure distribution on the propeller, Fig.11, and the global agreement on the axial velocity contours from the full RANSE simulation is excellent.

Fig.11: Rear view of instantaneous computed dynamic pressure distribution on hull and propeller

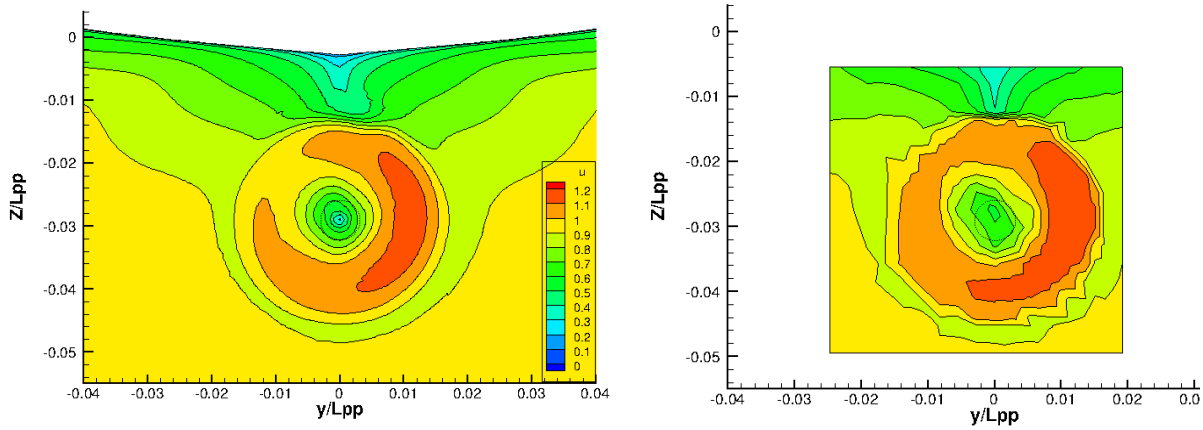

Fig.12: Iso axial velocity contours at x/Lpp=0.9911, n=9.5rps. Computed (left) and measured (right)

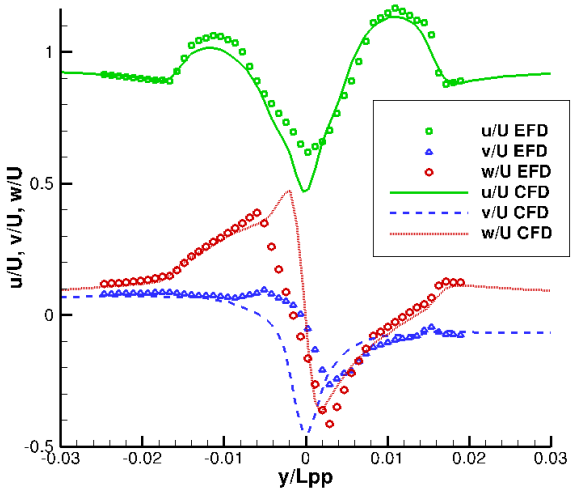

Fig.13: Velocity profiles at x/Lpp=0.9911, z/Lpp=-0.03, n=9.5rps

A more detailed analysis on the transversal evolution of the three components of the velocity is also possible thanks to detailed measurements performed at x/L_{PP} = 0.9911, z/L_{PP} = -0.03, Fig.13. The analysis of the cross-flow vectors reveals the existence of a strong main vortex obviously due to flow induced by the propeller. The first disagreement is the too low value of U close to the plane of symmetry, about U=0.5, while the experimental value is around U=0.7. Outside of this region, the agreement on the axial velocity component asymmetry is very good. The same remarks hold for the vertical and transverse velocity components. If the change from positive to negative values across the vertical plane of these velocity components is too abrupt in the computation, the minimum and maximum values are in rather good agreement.

4. Conclusion

The objective of this paper is the computation of free surface viscous flows around a self-propelled ship with the help of a sliding grid technique. This challenging case is the very first computation performed with the ISIS-CFD flow solver for a full RANSE simulation of a propeller operating behind a ship. The applied method works in parallel and the different sub-domains can be distributed arbitrarily over the processors. A grid convergence analysis has been conducted first about an isolated propeller in order to evaluate the minimum settings for the generation of a hexahedral mesh. The same propeller, at a particular point of the open water test, has been used to check the sliding grid approach with a grid generated on the basis of the grid convergence study. In both cases (unique rotating grid or sliding grids) the fluctuations of the forces over the last computed period of rotation have been analysed to point out that if the mean values agree below 0.3%, the fluctuations in time are more pronounced and multiplied by a factor 3 with the computation involving the sliding grid approach. Starting from a coarse mesh, the adaptive grid refinement method has been applied to a rotating propeller. It automatically improves the prediction of the tip vortex with a significant consequence on the prediction of the forces. The study of the flow around the container ship gave us the opportunity to assess the flow solver to predict hull/propeller coupling in self-propulsion conditions. Considering the complexity of this exercise, the results obtained for this first try are very promising.

Acknowledgements

This work was granted access to the HPC resources of IDRIS under the allocation 2011-21308 made by GENCI (Grand Equipement National de Calcul Intensif).

References

ALAUZET, F.; LOSEILLE, A (2010), *High-order sonic boom modelling based on adaptive methods*, J. Comp. Phys. 229/3, pp.561-593

DUVIGNEAU, R.; VISONNEAU, M.; DENG, G.B. (2003), *On the role played by turbulence closures in hull shape optimization at model and full scale*, J. Marine Science and Technology 8/1, pp.1-25

HINO, T. (Ed.) (2005), *Proceedings of CFD Workshop Tokyo 2005*, NMRI report
(see also http://www.gothenburg2010.org/instructions_KCS/Case2.3a/Case_2-3a.htm)

LEROYER, A.; VISONNEAU, M. (2005), *Numerical methods for RANSE simulations of a self-propelled fish-like body*, J. Fluid & Structures 20/3, pp.975-991

QUEUTEY, P.; VISONNEAU, M. (2007), *An interface capturing method for free-surface hydrodynamic flows*, Computers & Fluids 36/9, pp.1481-1510

SALVATORE, F. (2007), *The INSEAN E779A Propeller Experimental Dataset*, VIRTUE Project, Deliverable 4.1.3 of the VIRTUE Project

WACKERS, J.; AIT SAID, K.; DENG, G.B.; MIZINE, I.; QUEUTEY, P.; VISONNEAU, M. (2010a), *Adaptive grid refinement applied to RANS ship flow computation*, 28th ONR Workshop on Naval Hydrodynamics, Pasadena

WACKERS, J.; DENG, G.B.; VISONNEAU, M. (2010b), *Tensor-based grid refinement criteria for ship flow simulation*, 12th Numerical Towing Tank Symposium (NuTTS '10), Duisburg

WACKERS, J.; DENG, G.B.; VISONNEAU, M. (2011), *Combined tensor-based refinement criteria for anisotropic mesh adaptation in ship wave simulation*, ADMOS 2011, Paris

WACKERS, J.; DENG, G.B.; LEROYER, A.; QUEUTEY, P.; VISONNEAU, M. (2012), *Adaptive grid refinement for hydrodynamics flows*, Computers & Fluids 55, pp.85-100