Sliding Grids and Adaptive Grid Refinement for RANS Simulation of Ship-Propeller Interaction

Patrick Queutey, Ganbo Deng, Jeroen Wackers, Emmanuel Guilmineau, Alban Leroyer, Michel Visonneau

Sliding Grids and Adaptive Grid Refinement for RANS Simulation of Ship-Propeller Interaction

By Patrick Queutey[1,*], Garbo Deng[1], Jeroen Wackers[1], Emmanuel Guilmineau[1], Alban Leroyer[1] & Michel Visonneau[1]

ABSTRACT

The paper describes a computational approach for a numerical propulsion test. Key techniques concern the computation of free-surface viscous flows around propellers using the sliding grid technique and accurate wave capturing around the hull. An advanced numerical technique resolves very small-scale flow motion such as tip vortex. Efficiency and accuracy are balanced using an adaptive mesh refinement technique. This approach is validated against the KCS test case proposed for the Tokyo 2005 and Gothenburg 2010 CFD validation workshops.

Key words: CFD, Grid, Propulsion, RANSE

1 Introduction

Accurate prediction of the flow 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 and its interaction with the flow coming from the hull. While turbulence modelling for the bilge vortex is the principal issue to ensure an accurate prediction for the wake flow, more physical models are involved in a propeller simulation: transition, cavitation, turbulence, ventilation etc. Advanced numerical techniques are also required to capture very small scale flow motion such as tip vortices.

The topic of the presented research is the simulation of ship propellers in extreme operating conditions, with accurate modelling of the relevant physics involved. This requires the capability to simulate a propeller rotating behind a ship hull, combined with effects of free-surface deformation, ventilation and cavitation. These capabilities are being developed for inclusion in the unstructured finite-volume flow solver ISIS-CFD, *Duvigneau et al.* (2003) and *Queutey and Visonneau* (2007).

An essential building block for these simulations is a sliding grid technique, which allows a part of the grid containing propeller to rotate within the main grid while keeping a connection

[1] CNRS, Ecole Centrale de Nantes, Nantes, France

[*] *Corresponding author*. Present address: patrick.queutey@ec-nantes.fr

between the two parts, Fig. 1. An additional topic is automatic adaptive grid refinement, aiming at accurate resolution of phenomena induced by highly localised low pressure zones.

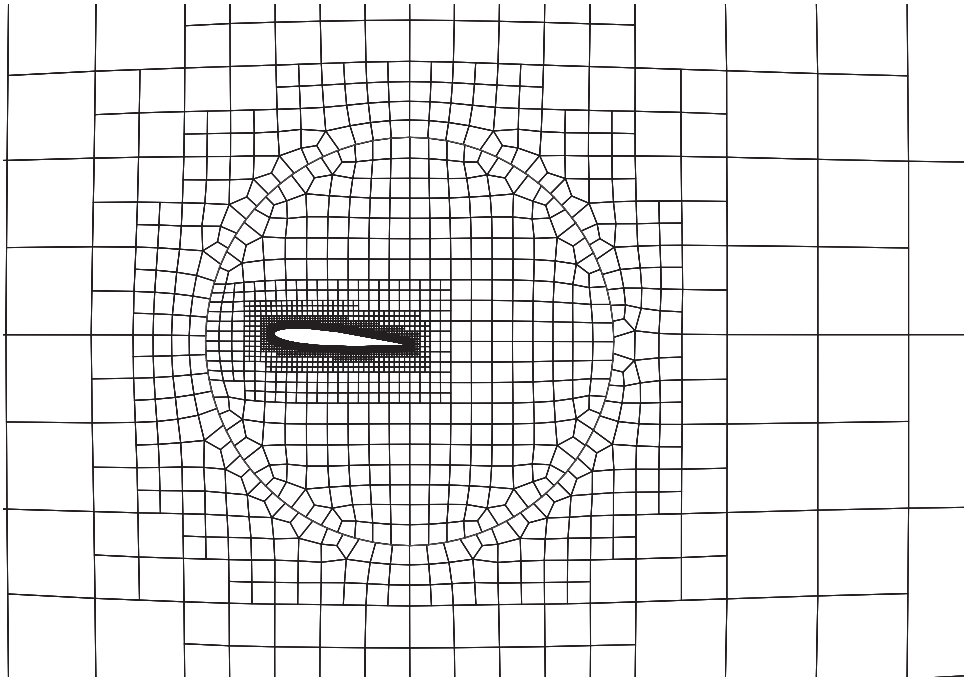

Fig. 1: Example of a sliding grid around a pitching airfoil including sliding interface.

However, the sliding grid approach and the adaptive grid refinement have to be general enough to work together. Recently developed sliding grid capability was specifically constructed to work together with our existing grid refinement method, *Wackers et al.* (2010b) and *Wackers et al.* (2010a). Both techniques are powerful enough to deal with the unstructured hexahedral grids currently used.

Sliding grid methods for unstructured grids are far more complicated than for structured grids. Among others, the connection between the two subdomains of the grid has to be often reconstructed, because it does not follow a regular pattern. In addition, it is not easy to ensure flux conservation over the interface. A connection between the domains was adopted 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.

An adaptive mesh refinement technique is available with different refinement criteria based on the pressure Hessian and the free-surface volume fraction. The method works in parallel with the sliding grid approach and the different subdomains can be distributed arbitrarily over the processors. The combination with adaptive grid refinement is obtained by treating the sliding faces in the same way as standard boundary faces during refinement. The cell sizes are not explicitly synchronised 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 is an incompressible unsteady Reynolds-averaged Navier-Stokes (RANS) method, *Duvigneau et al.* (2003) and *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 and Chow* (1983) 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 cell 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, and 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 solving 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 such as the Explicit Algebraic Stress Model with rotation correction, *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 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 Reconstructions on Faces

To compute the fluxes, the quantities on the cell faces are reconstructed from those in the cell centres, Fig. 2. For the diffusive fluxes and the coefficients in the pressure equation, the quantities on a face and the derivatives in the normal direction to the faces 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-orthogonality corrections are added which use the gradients computed in the cell centres. For the convective fluxes, the AVLSMART scheme is used, *Przulj and Basara* (2001), in the NVD context for unstructured grids, established by Jasak, 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).

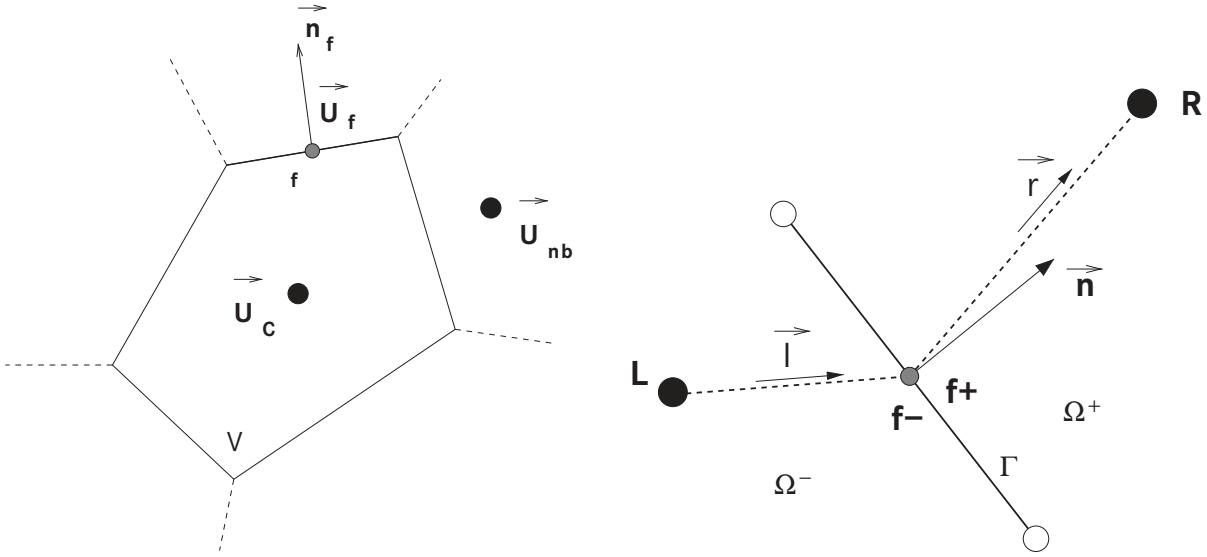

Fig. 2: Cell, faces and neighbours (left), reconstruction on a face (right).

For the following discussion, the essential point is that all fluxes on a face can be reconstructed from the states and gradients in the centres of its two neighbour cells, as well as the positions of these centres. In the developed domain decomposition approach for parallel computing, this neighbour cell information is the only data exchanged over the interface faces between the partitions.

3 Sliding Interface Implementation

To compute the fluxes over a sliding interface between two subdomains, connections between cells on the two sides of this interface are to be established. 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 a face on the interface, the cell centre is found in the other subdomain that matches best the face. This cell is then used as neighbour cell for a flux computation in exactly the same way as for standard cells.

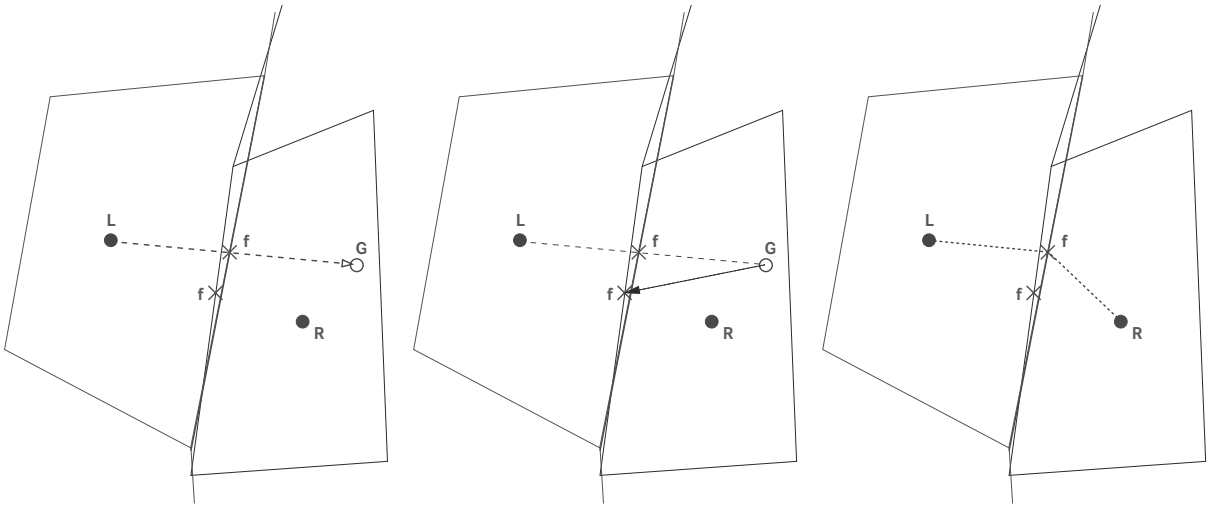

Fig. 3: Sliding faces: construction of ghost points (left), searching the global faces (centre) and the new neighbour cell (right).

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 reference during 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 the outside neighbours for the local sliding faces. If the neighbour is on another processor, MPI communication is established in the same way as for the usual 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.

Thus, for parallel computations, we do not explicitly split the grid in two parts, assigning a number of processors to each subdomain and partitioning each subdomain separately. Instead, the grid can be arbitrarily spread over available processors, and 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 the proposed approach, redistribution is fully flexible.

4 Automatic Grid Refinement

The automatic adaptive grid refinement technique included in the solver ISIS-CFD is, for example, described in *Wackers et al.* (2010b), *Wackers et al.* (2010a) and *Wackers et al.* (2012). The technique is intended to be used for 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.* (2012). The grid refinement is fully parallel and includes an automatic dynamic load balancing, in order to redistribute the refined grid over the available processors when some partitions have been refined more than the others.

4.1 Coupling with Sliding Grids

Since the connectivity between sliding faces is recomputed before each time step, no connectivity 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, in the same way as 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 (in some way) from the flow field, which is smooth over the interface. Thus, if the refinement criterion is computed correctly, a smooth variation of the cell size over the sliding interface is expected. This implicit guarantee of a smooth cell size transition over the interface is experimentally verified later in this paper.

A second point is the dynamic load balancing. As part of this procedure, the grid is repartitioned using ParMETIS. This tool is a graph partitioner that searches a balanced distribution of the graph nodes and a minimum number of graph edges cut; its input graph has the cell centres as nodes and the cell connections to neighbour cells as edges.

To construct the graph, no graph edges are created for the sliding faces: all connections via sliding faces are ignored. Thus, ParMETIS 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. These solvers generally have a two-loop structure; the innermost loop consists of (for example) a single Gauss-Seidel sweep, and in the outer loop between these sweeps the data on both sliding faces and interface faces between processors are updated. The fewer the number of states changed in this outer loop, the faster will be the convergence. Thus, it is advantageous to have sliding faces and inter-partition interface faces coincide, as this keeps their total number low.

4.2 Refinement Criteria

For the following computations, two different refinement criteria are employed. The first of these, the free-surface criterion, refines in the neighbourhood of the water surface. Directional refinement is employed to refine the grid in the direction normal to the surface only. Where the free surface is diagonal to the grid directions, isotropic refinement is used, but where the surface is horizontal, directional refinement is chosen. The resulting zone of directional refinement includes the undisturbed water surface, as well as smooth wave crests and troughs. This is essential to keep the number of grid cells low, because the water surface is often nearly undisturbed in most of the domain. Fig. 4 gives an illustration of this refinement principle.

The second criterion is based on the Hessian matrix of second spatial derivatives of the pressure, which is similar to criteria being used for tetrahedral grid refinement *Alauzet and Loseille* (2010). This criterion detects the presence of, for example, vortices, stagnation points and suction

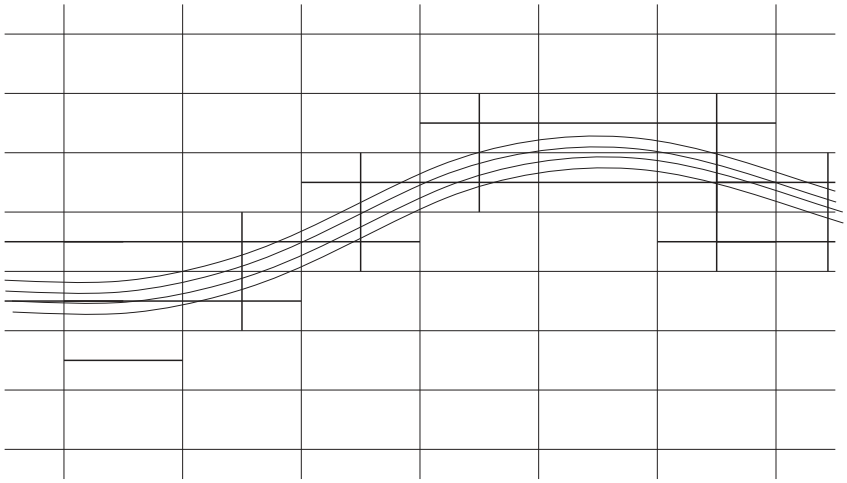

Fig. 4: Isotropic and directional refinement at the free surface; curves represent volume fraction isolines.

peaks. The reason for basing the criterion on the pressure is its behaviour in boundary layers. There, the pressure varies little, while the velocity gradients are very large. Thus, a velocity-based criterion would produce refinement everywhere, while the pressure criterion refines only at the locations where the external flow creates a pressure gradient. In those cases where the flow outside the boundary layers is the main interest of the computation, costly and unnecessary refinement in all boundary layer cells can be prevented by using the pressure criterion.

The major difficulty in using the Hessian tensor as a refinement criterion is the accurate evaluation of the second derivatives, independent of the mesh *Wackers et al.* (2012). If the criterion computation is perturbed by the local grid refinement, the criterion may react more to existing grid refinement than to the pressure field. To exclude this undesired effect, numerical errors in the computed second derivatives must be significantly smaller than the derivatives themselves in all cells. A least-squares method based on third-order polynomials is used to compute this criterion. In each cell, the polynomial is computed that best fits the pressure in the cell, its neighbours and its neighbours' neighbours, in the least-squares sense. The approximated Hessian is constructed from the second derivatives of this polynomial. The least-squares procedure guarantees that the difference between the approximating polynomial and the real pressure is at least fourth-order. Hence, the approximated second derivatives are second-order accurate, independent of the mesh geometry.

5 Numerical Tests

In this section, numerical test cases of propeller open-water flow and ship-propeller interaction are presented. These are meant both to show that the combination of sliding interfaces and grid refinement is successful and to assess the performance of these techniques individually for propeller-related flows.

5.1 Pitching Airfoil

The first test case is intended to study in detail the interaction of sliding interfaces and grid refinement. The case is a two-dimensional flow around a pitching NACA0012 airfoil, placed in a uniform flow and then rotated continuously with respect to its trailing edge, thus shedding strong counter-rotating vortices. The Reynolds number based on the chord and the free-stream velocity is $Re = 40$. The case is modelled with a circular subdomain around the airfoil, which rotates inside the outer grid, Fig. 1. Grid refinement is based on the Hessian criterion, because this case does not involve free surface.

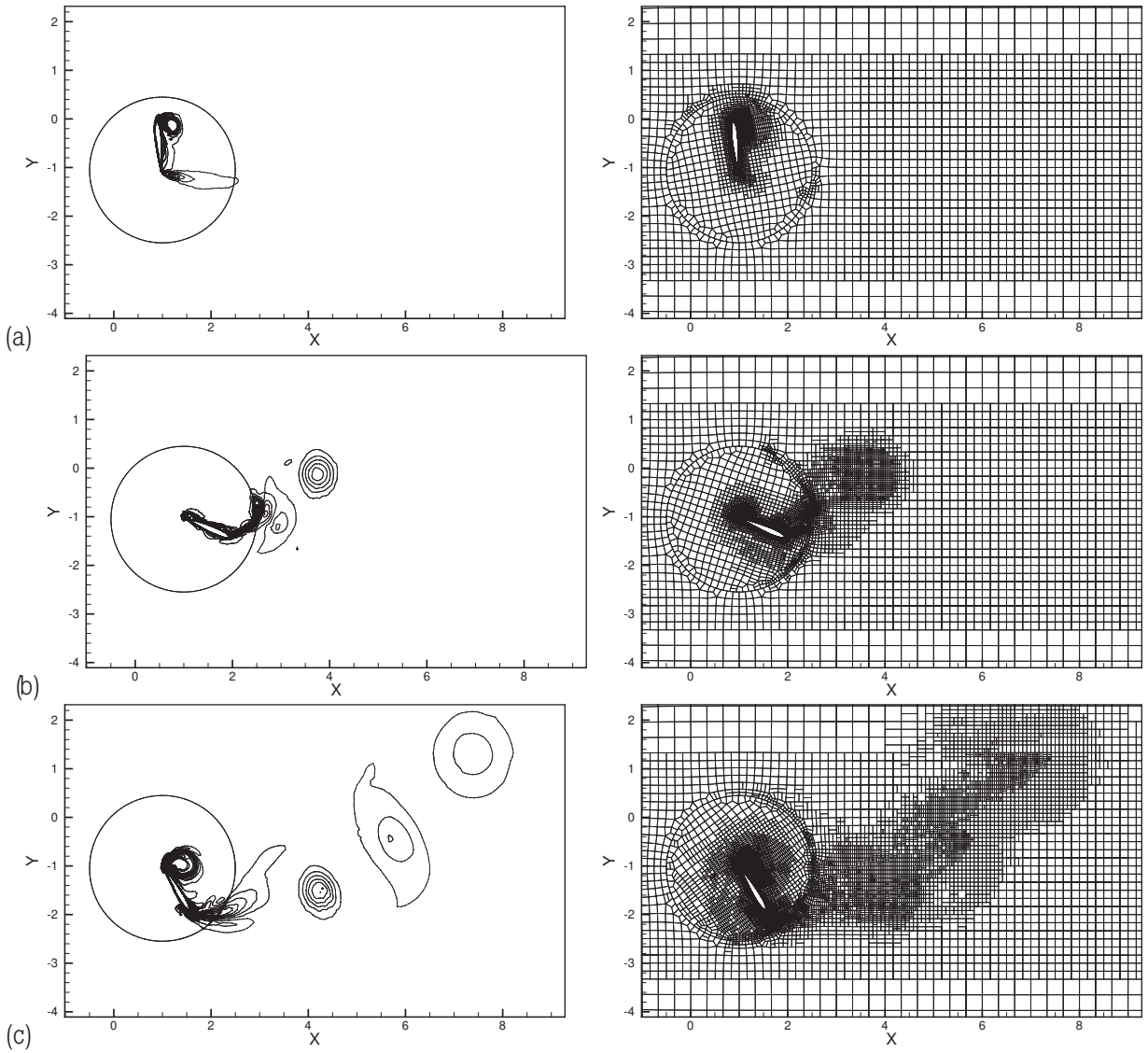

Fig. 5: Vorticity contours and refined meshes for the pitching airfoil case at $t = 0.7$ (a) and 1.05 (b) and during the second revolution at $t = 2.2$ (c).

Fig. 5 shows snapshots of the flow at three different time instants. The turning airfoil creates both isolated vortices and trailing vortex sheets. Even if some minor perturbations occur, these structures cross the sliding interface without disturbances and they persist in the far field. The meshes show that the criterion correctly identifies the vortex zones, both close to the airfoil and far away. Without any explicit adjustment, the grid size varies smoothly over the sliding interface. Besides, earlier refinements are undone correctly: no refinement remains on the upstream side of the sliding grid, which was refined when it was on the downstream side.

5.2 Propeller with Free Surface

The combination of sliding faces and free-surface grid refinement is tested on the case of a ventilating propeller. The objective is not to simulate a surface-piercing propeller of high-speed craft, designed to operate in partially submerged conditions, but to simulate the ventilation of a conventional propeller designed to operate in fully submerged conditions. Ventilation can occur when such a propeller operates too close to the free surface, or in heavy sea states. Such ventilation results in thrust loss and may induce violent impact loading, leading to propeller damage. Hence, successful prediction of such ventilation phenomena is useful for practical applications.

The same test case is used that was investigated by *Califano* (2010). Experimental studies have been conducted with *9* different combinations of free-stream velocity U and shaft frequency n. Only

one of them is simulated in the present preliminary study, namely the case with $U = 0.35$ m/s and $n = 14$ Hz, which gives an advance ratio $J = 0.1$. The submergence ratio is $h/R = 1.64$ (h is the distance between the shaft centre and the free-surface and R is the radius of the propeller).

The propeller with a diameter $D = 0.25$ m is included in a rotating cylindrical subdomain with a diameter of 0.3 m and a thickness of 0.2 m. The original grid contains only about 1.54 million cells for this subdomain, with suitable grid resolution on the walls for low-Reynolds turbulent flow simulation. The second, fixed subdomain contains about 0.36 million cells and extends in the ranges -7 m $< X < 3$ m, -2 m $< Y < 2$ m and -3 m $< Z < 1$ m. The grid is partitioned in four computational blocks. The time step employed is 0.0005 s, which corresponds to about 142 time steps per revolution. The free-stream velocity and the propeller rotation rate are accelerated from zero to their maxima in 0.2 s, while in the measurement, the acceleration period is about 0.42 s. The time resolution is reduced to decrease the computational time for this preliminary investigation.

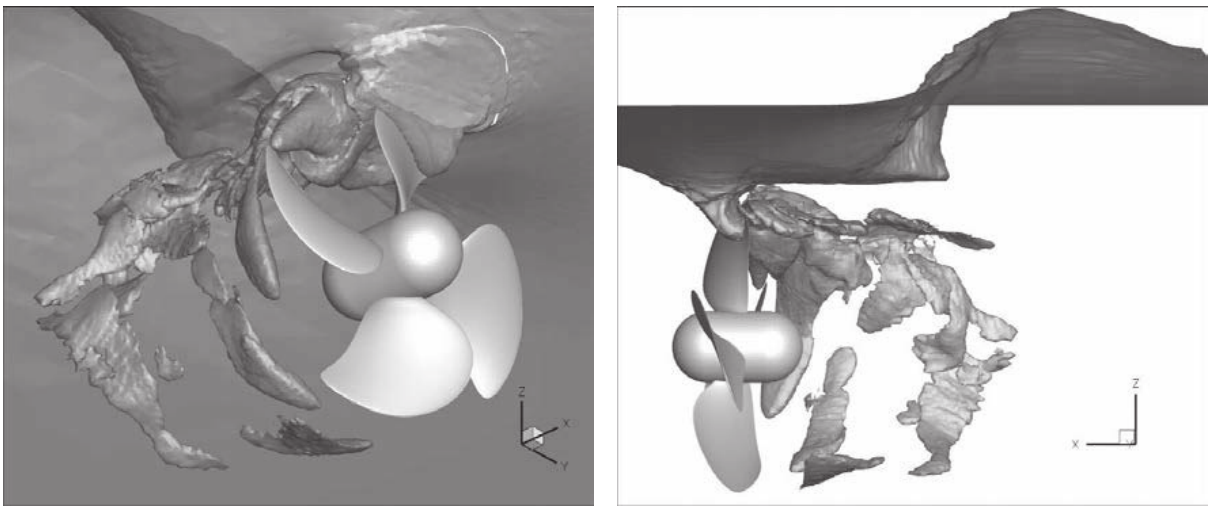

Fig. 6: Overview of the computed free surface after three rotations of the propeller.

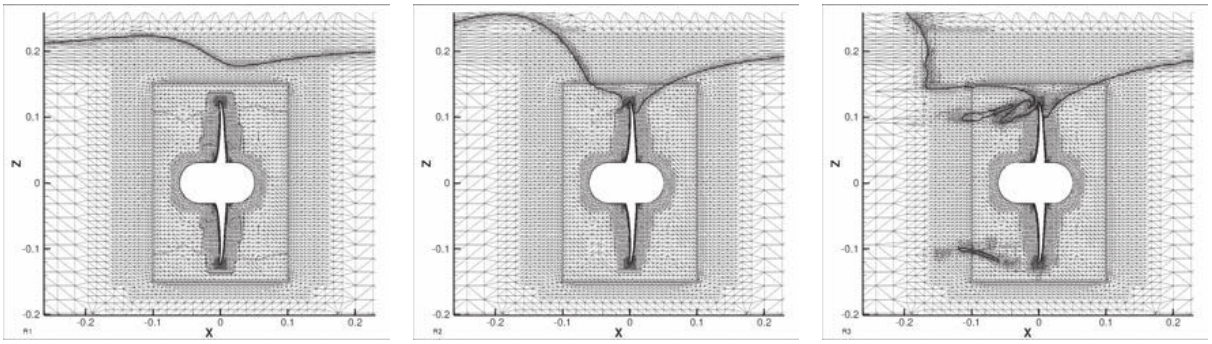

Fig. 7: Grid density and water/air interface in the median plane cut $y = 0$ for the ventilating propeller, after 1 rotation (left), 2 rotations (centre) and 3 rotations (right). The sliding interface corresponds to the limits of the rectangular region surrounding the propeller.

Figure 6 shows the instantaneous free surface after only three revolutions of the propeller when significant air entrainment is already occurred. The strong free-surface deformation seen behind the propeller is created by the rapid acceleration used to spin up the propeller from rest. This deformation moves out downstream later. The flow is not established after three revolutions. Figure 7 focuses on the median plane $y = 0$ at three successive time steps: the thick line represents the 0.5 iso-surface of the volume fraction. The grid cuts in the figure are obtained from triangulation during the computation and are only representative of the density of volume grid, which is purely hexahedral. At the third rotation the grid size has increased by 15% only and contains 2.2 million

cells. The current result demonstrates therefore the efficiency of the refinement procedure together with the sliding interface approach.

5.3 INSEAN E779A Propeller in Open Water

The individual performance of the sliding grid and automatic grid refinement procedures for simulations of ship propellers is studied by computing the open-water flow around the INSEAN E779A propeller. This model propeller has a diameter $D = 0.227$ m and was investigated at a rotational speed of $n = 11.79$ rps and advance ratio $J = 0.895$. In the computations, the water density is $\rho = 1000$ kg/m^3 and the kinematic viscosity is $\mu = 1.1099 \cdot 10^{-3}$ kg/(ms). Tests have been performed in the INSEAN middle towing tank, with the propeller axis $1.5D$ below the free surface. However, in the present computations the free surface is not taken into account.

The non-dimensional coefficients to be considered are: advance ratio $J = U/(nD)$, thrust coefficient $K_T = F_X/(\rho n^2 D^4)$, torque coefficient $K_Q = M_X/(\rho n^2 D^5)$ and open-water efficiency coefficient $\eta = (JK_T)/(2\pi K_Q)$. Force F_x and moment M_x are computed on the propeller blades only. This is not exactly the same as in 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. In the simulations, 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. Besides, 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, at 180 time steps per revolution, continues to fluctuate in time. The mean forces are averaged over the last computed revolution.

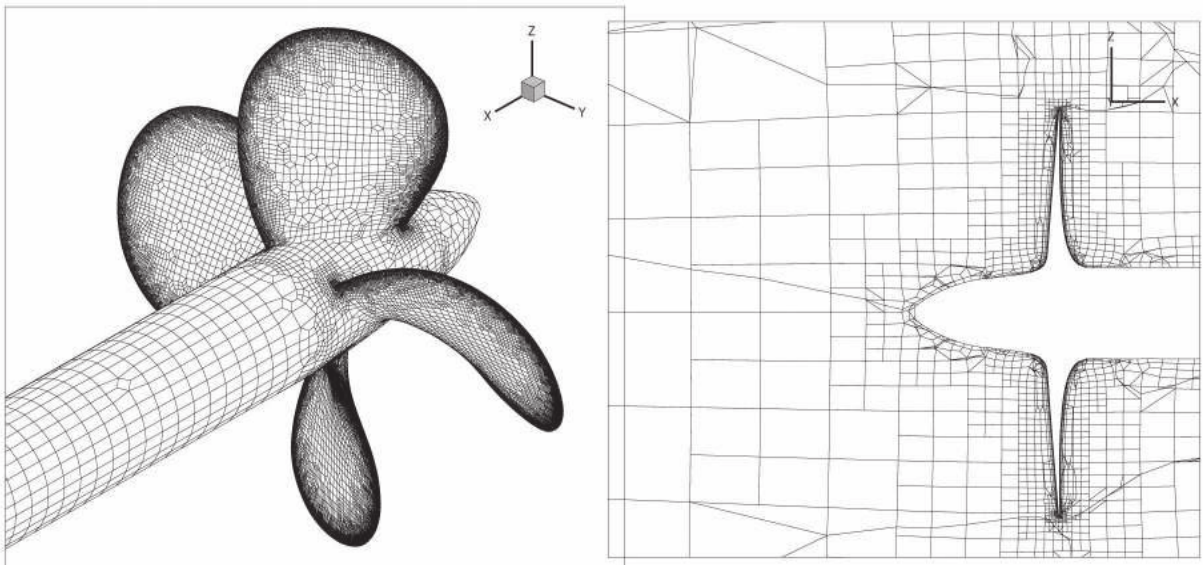

Fig. 8: Surface grid (left) and $Y = 0$ cut (right) for mesh 0.

Grid Convergence

As a reference for the following computations, a grid convergence study was conducted on a series of meshes generated with HEXPRESS for a single rotating domain. The computational domain is a cylinder extending from $X = 1.25D$ to $4.0D$ 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, and prescribed pressure on the outflow wall. Five meshes are created with a number of cells of 0.775 million (mesh 0), 1.87 million (mesh 1), 3.86 million (mesh 2), 6.97 million (mesh 3) and 11.4 million (mesh 4). In all cases, the boundary layer grid is made with a prescribed first-cell thickness of $1.53 \cdot 10^{-4}$; 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. 8.

Table 1 illustrates the convergence of the force and moment coefficients. Apart from grid *3*, where the boundary layer grid changed thickness from *10* to *8* cells, the convergence of K_T and K_Q is monotone. Assuming (somewhat heuristically) that the numerical uncertainty due to the mesh size is about *1.5* times the difference with grid *4*, then the numerical uncertainty is of the order of *4%* for grid *1*. The difference between the experimental value for F_X and the result on grid *4* is larger, thus modelling errors are larger than numerical errors for this and finer grids. The grid density of the current grid 1 will therefore be retained for all further computations.

Tab. 1: Grid dependency at $J = 0.895$.

Mesh	K_T	Diff. M4	K_Q	Diff. M4	η	Diff. M4
0	*0.14685*	*4.19%*	*0.031966*	*5.62%*	*0.6544*	*−1.35%*
1	*0.14403*	*2.19%*	*0.031082*	*2.70%*	*0.6601*	*−0.50%*
2	*0.14240*	*1.03%*	*0.030599*	*1.10%*	*0.6629*	*−0.08%*
3	*0.13958*	*−0.97%*	*0.030052*	*−0.71%*	*0.6616*	*−0.27%*
4	*0.14095*		*0.030266*		*0.6634*	
Exp.	*0.150*	*6.42%*	*0.0294*	*−2.86%*	*0.7267*	*9.55%*

Simulation with Sliding Grids

Another simulation was conducted to validate the sliding grid methodology, with a grid density similar to the one for grid 1. Fig. 9 presents a cut $Y = 0$ through the grid with the sliding interface outlined. Inside the cylinder limited by the sliding interface the grid is rotating and outside of this cylinder it is fixed. The boundary conditions and the numerical settings used are the same as those for the previous case with a monolithic rotating grid. Table 2 contains the results of the force and moment of the simulations without and with sliding grids. It is also used to compare the maximum difference in force between two blades $\Delta f_{X,blade}$, and the strength of the fluctuations in time $\tilde{f}_{X,t}$ computed by taking standard deviation of the signals.

Tab. 2: Differences between blades and between solutions on different grids (*1* monolithic grid *1*, SG with sliding grids).

Mesh	K_T	$\Delta F_{X,blade}/F_X$	$\tilde{F}_{X,t}/F_X$	K_Q	$\Delta M_{X,blade}/M_X$	$\tilde{M}_{X,t}/M_X$
1	*0.1440*	*0.022*	$5.8 \cdot 10^{-4}$	*0.0311*	*0.019*	$3.7 \cdot 10^{-4}$
SG	*0.1442*	*0.032*	$1.6 \cdot 10^{-3}$	*0.0310*	*0.024*	$1.2 \cdot 10^{-3}$

The mean forces between the two computations are close to each other with a relative difference of about *0.1%* for the force and about *0.3%* for the moment. The differences between the blades are similar, between *2%* and *3%*, but the fluctuations in the time are more pronounced – *3* times larger – in the computation with the sliding grid approach. The reason 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, some discontinuity appears as the grid is changing between two time steps when connection between sliding faces changes.

Simulation with Adaptive Grid Refinement

The final computation is performed on a single-domain grid with automatic grid refinement. The pressure Hessian-based criterion is used to control the grid adaptation since it reacts to high pressure variations associated with vortical structures. Here, the tip vortex on each blade is considered, where a strong suction peak could induce cavitation. The forces were compared with the original grid *1*

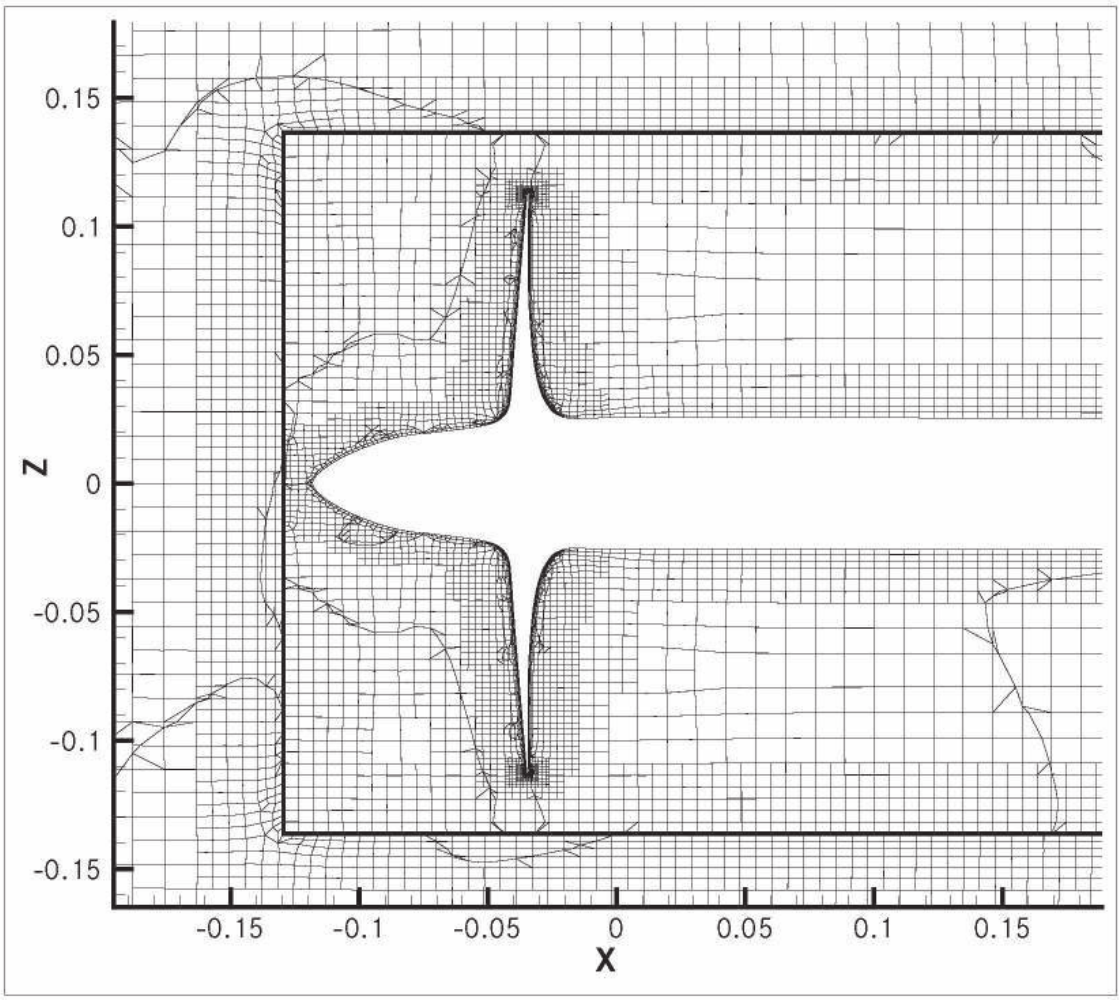

Fig. 9: $Y = 0$ cut for mesh *1* with sliding interface; the sliding interface is indicated by the thick line.

results and with the results obtained on an automatically refined grid starting from the coarsest grid *0*. The minimum cell size for the adaptive grid is set to *0.35* mm.

Fig. 10 compares the iso-surface of pressure value $p = -2$ kPa. The effect of automatic refinement is clearly detected in the shape of the surface pressure. Note that grid *1* has *1.86* million cells, which is close to the number of cells in the adapted grid (about *2.01* million cells) at the end of the simulation. With the automatic refinement, the computed values of K_T and K_Q increased by *0.7%* and *0.4%*, respectively, Table 3. This is surprising because the meshes are not at all similar; the finest cells in the refined mesh are much smaller than even the cells of grid *4*.

Two effects play a role for forces on the propeller. First, on coarse meshes, vortex strengths are under-predicted. Thus, the suction peak is expected to be weaker on the grid *1* than in reality, which leads to under-prediction of both K_T and K_Q. The vortex is much better predicted on the refined grid. Second, without transition modelling, the turbulent boundary layers become too thick on fine meshes, which leads to under-prediction in the forces. It is possible that the boundary layers are better resolved on the refined grid than on grid *1* and thus the two effects may cancel.

Tab. 3: Forces, deviations and differences between blades (*1* original grid *1*, *Ref* with grid refinement).

Mesh	K_T	$\Delta F_{X,blade}/F_X$	$\tilde{F}_{X,t}/F_X$	K_Q	$\Delta M_{X,blade}/M_X$	$\tilde{M}_{X,t}/M_X$
1	*0.144*	*0.022*	$5.8 \cdot 10^{-4}$	*0.0311*	*0.019*	$3.7 \cdot 10^{-4}$
Ref	*0.145*	*0.019*	$5.6 \cdot 10^{-4}$	*0.0312*	*0.017*	$4.7 \cdot 10^{-4}$

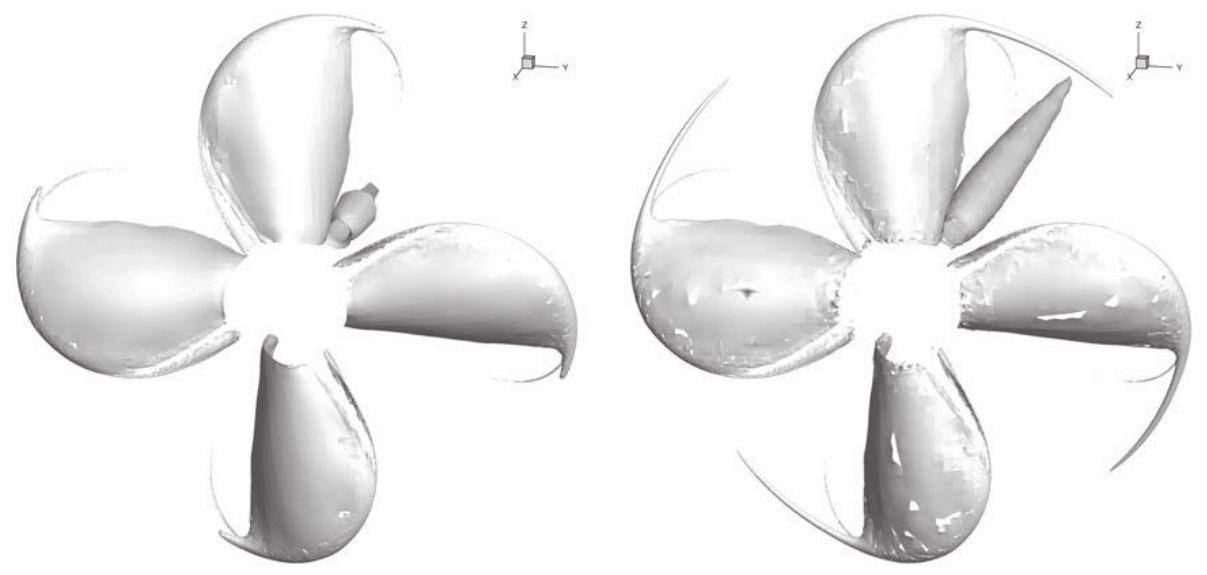

Fig. 10: Iso-pressure surface $p = -2$ kPa for $J = 0.880$ from grid *1* (*1.86* million cells) on the left and from the adapted grid (*1.92* million cells) starting with grid *0* on the right.

5.4 MOERI Container Ship (KCS)

The KCS container ship, tested by the Korea Research 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 case corresponds to a self-propelled model. The sliding grid approach is employed, taking into account ship-propeller interaction.

Mesh Generation

The mesh is generated with a sliding interface to separate the propeller and hull subdomains. The propeller grid contains about *2* million cells to agree with the grid *1* settings from the convergence analysis on the INSEAN propeller. A semi-spherical cap is added to the hub. The up- and downstream sliding grid interfaces are located at $0.114D$ and $0.249D$, respectively, from the centre of the propeller, while the outer sliding grid interface is located at the radius of $0.6D$. The grid for the hull with appropriate refinement near the bow, the stern and the free surface contains about *3.5* million cells. Meshes were generated by the hexahedral unstructured grid generator HEXPRESS.

Computation and Results

Time-accurate simulation is performed for the whole transition procedure of a running propeller. The model is fixed in trim and sinkage. It is first accelerated to the maximum speed with a frozen propeller, until a nearly steady state is reached (after about *800* time steps). A large time step of *0.03* s is employed during this period. After that, propeller rotation is started and time step is reduced to the value used in the open water computation, *0.000526* s (*200* time steps per period for $n = 9.5$ rps). It turns out that such a procedure is extremely time consuming. The propeller thrust begins to stabilise after about *12* revolutions, i.e. *1.26* s physical time. The model resistance takes more time to converge. It decreases from a maximum value of *91.5* N, observed at about the same time as when the propeller thrust is stabilised, to a nearly converged value of about *88.4* N *1.5* s later. The cost of this computation is about *10* times higher than for a resistance test run.

Axial velocity contours and cross-flow vectors are shown downstream of the propeller at $X/L_{pp} = 0.991$, Fig. 11 and 12, respectively. The axial velocity contours are characterised by two regions inside and outside the propeller disk. Inside the propeller disk, one notices a characteristic

asymmetric behaviour with a crescent-like region of high velocity (between $u/U = 1.1$ and 1.2). The overall agreement of the axial velocity contours with the full RANS simulation is excellent. A more detailed analysis of the transversal evolution of the three components of the velocity is also possible due to detailed measurements performed at $X/L_{pp} = 0.9911$, $Z/L_{pp} = -0.03$, Fig. 12. The analysis of the cross-flow vectors reveals a strong main vortex, obviously generated by the propeller-induced flow. The first disagreement is the too low value of longitudinal velocity close to the plane of symmetry: about 0.5, while the experimental value is about 0.7. Outside of this region, the agreement on the axial velocity component asymmetry is very good. The same holds for the vertical and transverse velocity components. While 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.

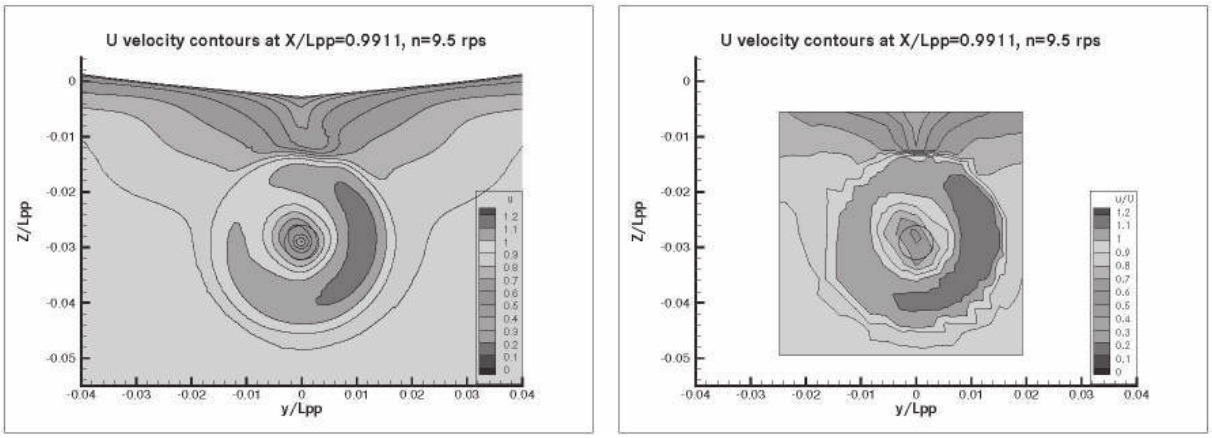

Fig. 11: Computed (left) and measured (right) iso-contours of axial velocity at $x/L_{pp} = 0.9911$, $n = 9.5$ rps.

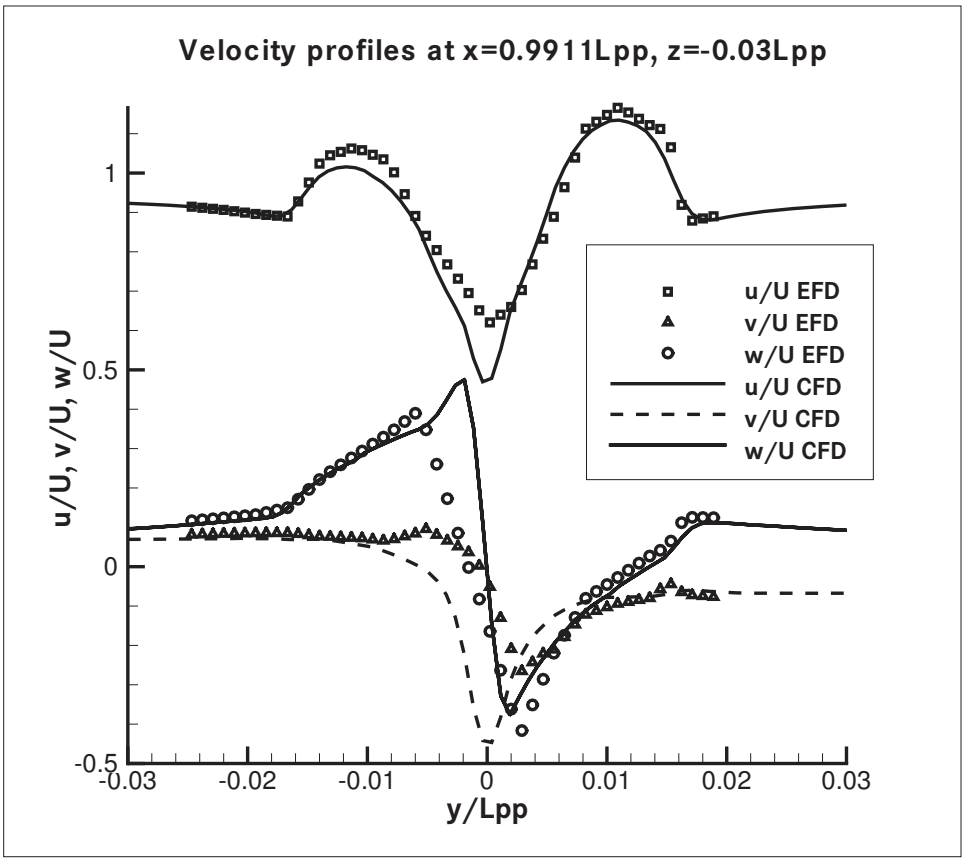

Fig. 12: Velocity profiles at $x/L_{pp} = 0.9911$, $z/L_{pp} = -0.03$, $n = 9.5$ rps.

6 Conclusion

The objective of this paper is the computation of free surface viscous flows around propellers and self-propelled ships with the help of a sliding grid technique. The method works in parallel, and different subdomains can be distributed arbitrarily over the processors. No explicit interpolation is used to find the states on the sliding faces; instead, the coupling algorithm identifies real cells that are used as neighbours for the cells in the other subdomain.

The combination with adaptive grid refinement is obtained by treating the sliding faces in the similar way as the standard boundary faces during refinement. The cell sizes are not explicitly synchronised over the sliding faces; instead, the continuity of the refinement criterion over the interface guarantees smoothly varying cell sizes. When distributing the multi-domain grid over the processors, the partition interfaces naturally fall on the sliding faces, which improves convergence.

Two initial test cases show that the combination of sliding interfaces and grid refinement is successful for the simulation of flows with vorticity and the free surface. After that, an isolated propeller is computed to compare the sliding grid approach with a monolithic rotating grid. The deviations of the mean values of the forces between these two cases are below *0.3%*. The fluctuations in time are rather pronounced and multiplied by a factor *3* for the computation involving the sliding grid approach. These oscillations are likely due to the interpolation on the sliding faces; an interpolation that varies more smoothly in time could reduce them. Finally, the adaptive grid refinement method was applied to the case. It automatically improves the prediction of the tip vortex.

The study of the flow around the container ship allows assessment of the flow solver to predict hull-propeller coupling in self-propulsion conditions. Considering the complexity of this exercise, the results obtained are very promising.

References

Alauzet, F. and Loseille, A. *(2010).* High-order sonic boom modeling based on adaptive methods. *J. Comp. Phys., 229(3):561–593*

Califano, A. *(2010). Dynamic loads on marine propellers due to intermittent ventilation. Ph.D. thesis, Norwegian University of Science and Technology (NTNU), Trondheim*

Duvigneau, R., Visonneau, M. and 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):1–25*

Hino, T. *(2005).* NProceedings CFD Workshop Tokyo, NMRI report, Tokyo

Leroyer, A. and Visonneau, M. *(2005).* Numerical methods for RANSE simulations of a self-propelled fish-like body. *J. Fluid & Structures, 20(3):975–991*

Przulj, V. and Basara, B. *(2001).* Bounded convection schemes for unstructured grids. *AIAA Paper No. 2001-2593*

Queutey, P. and Visonneau, M. *(2007).* An interface capturing method for free-surface hydrodynamic flows. *Computers & Fluids, 36(9):1481–1510*

Rhie, C. M. and Chow, W. L. *(1983).* A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation. *AIAA J., 17:1525–1532*

Wackers, J., Deng, G. B. and Visonneau, M. *(2010a).* Tensor-based grid refinement criteria for ship flow simulation. *In 12-th Numerical Towing Tank Symposium (NuTTS), Duisburg*

Wackers, J., Said, K. A., Deng, G. B., Mizine, I., Queutey, P. and Visonneau, M. *(2010b).* Adaptive grid refinement applied to RANS ship flow computation. *In 28-th ONR Workshop on Naval Hydrodynamics, Pasadena*

Wackers, J., Deng, G. B., Leroyer, A., Queutey, P. and Visonneau, M. *(2012).* Adaptive grid refinement for hydrodynamic flows. *Computers & Fluids, 55:85–100*