Multi-component Modeling with the Dual Mesh Connect Approach

Kaare A. Sørensen

Cassidian, 85077 Manching, Germany

Abstract. An approach for the treatment of CFD simulations on configurations consisting of more than one discretized components is described. The approach interfaces the dual meshes of the different blocks through cutting with an interface common for the blocks and is named the Dual Mesh Connect (DMC) approach. This approach is spatially conservative and capable of treating blocks with intersecting geometries in a natural manner. A description of the current implementation is given and the application of the method a wing geometry is shown.

Keywords: multi-component modeling, intersecting geometries, dual mesh connect.

1 Introduction

Most configurations considered in industry consist of a combination of several clearly identifiable components. In some cases, these components are moved in relation to each other to fulfill an important task for the configuration. For an aircraft, the control surfaces are typically a collection of distinct aerodynamic devices which are moved in relation to the airframe in a more or less rigid manner. Some devices are aerodynamically disconnected from the other components through gaps and do not have large surface interfaces with the remaining configuration. For other control surfaces, the gaps are so small that the components may be considered as a connected aerodynamic entity. Using the traditional CFD approach, each change in the geometry requires a new mesh to be created. For control surfaces which are directly connected with the surrounding geometries, a significant amount of CFD work may also be required to re–build the interfaces of the moved components. Mesh generation and CAD is user intensive and may significantly increase the time required for the numerical analysis of an aircraft. There are, however, alternatives which have the potential of considerably reducing the effort for such configurations. One approach is to use mesh deformation to move the pre-existing nodes in a mesh to conform with the new configuration. This approach works well for many cases, but will normally reduce the mesh quality or render a mesh invalid for large deformations. If there are narrow gaps in the original mesh and thus little room for the original elements to deform, the maximum allowable geometry change may be restrictively small. For cases where the control surface geometry is intersecting other components, movement

N. Kroll et al. (eds.), *Computational Flight Testing*, Notes on Numerical Fluid Mechanics and Multidisciplinary Design 123,
DOI: 10.1007/978-3-642-38877-4_18,

normally results in the appearance and disappearance of surface parts which is difficult to accurately represent with a pure deformation of the mesh. An improved approach combines mesh deformation with a local remeshing algorithm in regions where the deformation fails [1,2,3]. For time–accurate simulations where the control surface deployment varies between timesteps, this approach requires interpolation between the timesteps. Another problem with the deformation/remeshing approach is the treatment of intersecting geometries since a remeshing on the surface requires a valid CAD geometry. Another widely used approach for multi–component modeling is the Chimera method in which the various mesh blocks are combined though interpolation boundary conditions [4]. By introducing extrapolation boundary conditions, this approach can also be applied on configurations where the geometries of the various blocks intersect [5].

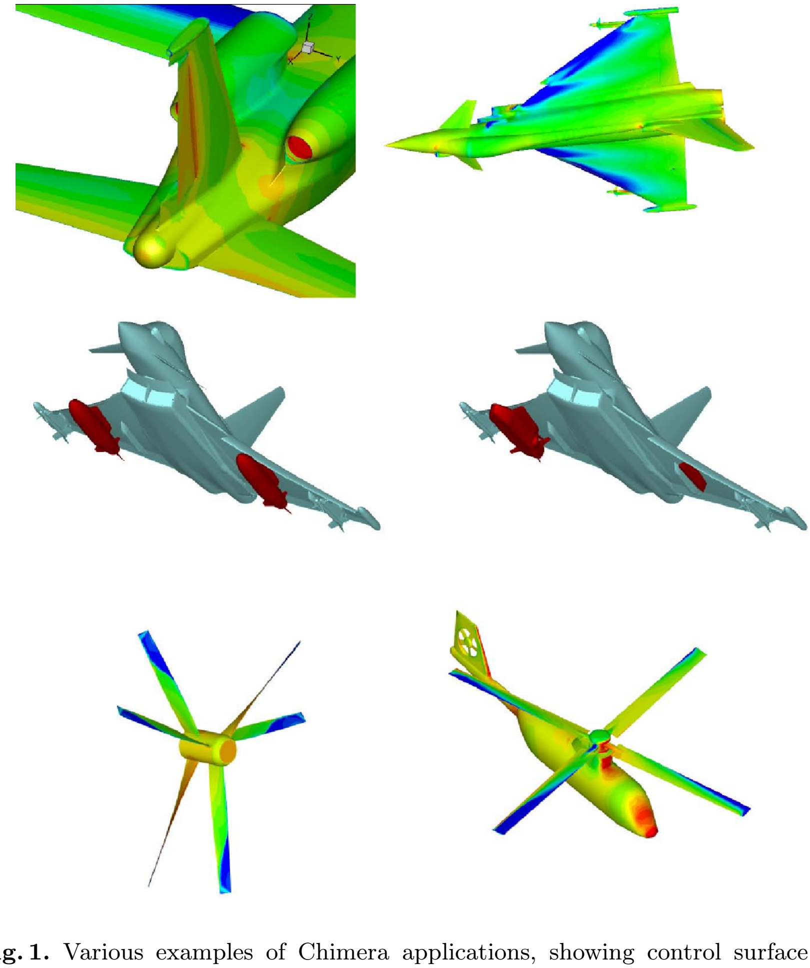

Fig. 1. Various examples of Chimera applications, showing control surface deployment, manoeuvre simulations, rapid store configuration changes, propeller and helicopter computations. Surface intersections occur in all the examples shown.

This approach has been successfully applied on a wide range of cases, Fig. 1. Such implementations are however not trivial and introduce local errors which may significantly disturb the flowfield locally if the meshes are too coarse. This may become a problem for very sensitive flowfields, such as for laminar wings, and significantly increases the total mesh size if large geometry intersections are present.

The approach considered in this paper has a similar user–interface with the Chimera approach, but is based on a completely different underlying algorithm, designed to eliminate the problems associated with extrapolation boundary conditions on surface intersections. Instead of employing interpolation or extrapolation boundary conditions, this algorithm attempts to interface the blocks directly through a cutting algorithm. Such an approach has previously been implemented for a cell–centered finite volume method on polygonal meshes [6]. In the current paper, an implementation working on the dual mesh of a cell–vertex finite volume approach is proposed. For this scheme, no new nodes are introduced through the cutting, as would have been the case if the cutting algorithm would work directly on the primary mesh for a cell–vertex method. More importantly, the averaging performed in the dual mesh flux coefficient construction has a smoothing effect on the discrete system, allowing for control volumes of irregular appearance (within limits) without detrimental effect on the accuracy or stability of the scheme. Since the cutting operations result in a consistent (merged) dual mesh, the CFD–solver runs in normal mode and does not 'know' that the discrete system stems from a collection of disjoint meshes. The discretization is thus automatically conservative and all dual mesh based algorithms (such as graph partitioning) work without any changes.

2 Current Algorithm

The DMC algorithm takes as input two or more primary mesh blocks and produces a consistent dual mesh through a cutting algorithm. In addition, a consistent hybrid primary mesh is produced for plotting purposes. The current algorithm can be summarized as follows:

1. The iso wall distance fields for each primary mesh block is constructed. For each node, the distance to the closest wall node on the same block, as well as the minimum distance to the wall nodes of the other blocks is found through a combined octree–closest neighbour search routine. The nodes are then categorized into two groups dependent on whether the closest wall node belongs to the same block or not. Elements containing both categories of nodes are defined as interface elements, and the element faces containing only nodes with the closest wall nodes on the same block are taken as the interface surface definition. An iterative algorithm, locally expanding the interface element layer, is applied to ensure that the interface surface is simply connected. Quadrilateral faces are split into triangles.
2. The dual element candidates to be cut are constructed. Element candidates with higher block–ID's that may intersect the interface surface are identified

and explicitly split up into a simplex dual mesh. The search algorithm is based on a binary tree with interval arithmetic for the bounding boxes of the interface surface.

3. The element candidate dual meshes are cut by the interface surface. The cutting operation is performed separately for each dual mesh subtetra. First, all topology nodes are registered and sorted according to the following types:
   – Interface triangle cuts a subtetra edge
   – Subtetra face cuts interface surface triangle edge
   – Subtetra contains an interface surface node
   – Special case nodes where the interface triangle node lies on a subtetra edge

   In addition to this, the subtetra faces and edges that are active are stored, as well as the active interface surface edges and nodes. Each subtetra/interface triangle combination is analyzed. There are around 60 allowed topologies for the cuts. The remaining regions of the subtetra faces are meshed using combinatorics, resulting in one, or several, triangular surface hulls for the original subtetra. These hulls are then filled with tetrahedra. Element subtetra which are located on the wrong side of the interface surface are deleted. To ensure the consistency of the cutting operation, each floating point operation is only performed once in the code. This implies that cuts on edges and faces must be communicated to neighbouring subtetra and elements which have not yet been cut. A typical cut case is shown in Fig. 2.
4. The interface surface containing the triangles on the interface, resulting from the cutting operation, is constructed.
5. The subset of the interface element candidates which are active in the cutting procedure are split into dual meshes and matched with the previously cut duals on the opposing block. On the outer edge of the active cut surface, element faces occur which only partially lie on the interface. These faces are also connected to avoid hanging nodes in the dual mesh.
6. A second interface surface consisting of element faces on block outer boundaries still active after the first cut (i.e. they are closer to a block–own wall node) is constructed.
7. The element candidates for the second cut operation are constructed. This is performed as above, by first creating a binary tree of the second interface surface bounding boxes and then selecting all active elements which have intersecting boundary boxes.
8. The second cut element candidates are cut with the second cut surface. This is performed by using the same routines as for the first cut, Fig. 3.
9. The elements on the other, non cut, side of the cutting surface are matched with the elements from the second cut to construct a complete, explicit and consistent dual mesh in the interface regions of the computational domain.
10. A hybrid primary mesh for plotting purposes is constructed. Since visualization is performed on the primary mesh, such a mesh is constructed by using the dual mesh triangles in the interface regions and matching them with the remaining primary mesh through a layer of split elements. In the interface region, this mesh contains elements of very poor quality resulting

from the cutting and matching operations and should not be used as basis for computations. The interpolation coefficients for the new nodes appearing for the visualization are constructed.

11. A consistent dual mesh for the entire computational domain is implicitly constructed. Initially, all dual meshes for the entire primary mesh blocks are constructed and stored in an edge based datastructure. The parts outside of the computational region for the block, i.e. on the wrong side of the interface surfaces, are identified and deleted. The explicitly created dual mesh tetras in the interface regions are distributed between the control volumes through criteria attempting to optimize their regularity. The current algorithm attempts to place control volume interfaces mid–way between the nodes, and, through a weighted algorithm, reduce the surface area of the control volumes. For the connections already present in the dual mesh, i.e. node pairs with an edge between them, the new contributions are added to the edge coefficient. For node pairs sharing a control volume interface without a pre-existing edge, e.g. node pairs from different blocks, a new edge is constructed. As a final check, the edge coefficients are summed up to confirm the consistency of the new dual mesh representation, and guaranteeing that the discretization is conservative.

The above described procedure is mostly topological, that is, if all special cases have been considered the method can not fail. This is however not the case for the parts which involve floating point operations, where the same computation can result in different topological results depending on the order of the operations involved, introducing inconsistencies in the dual. This issue influences the

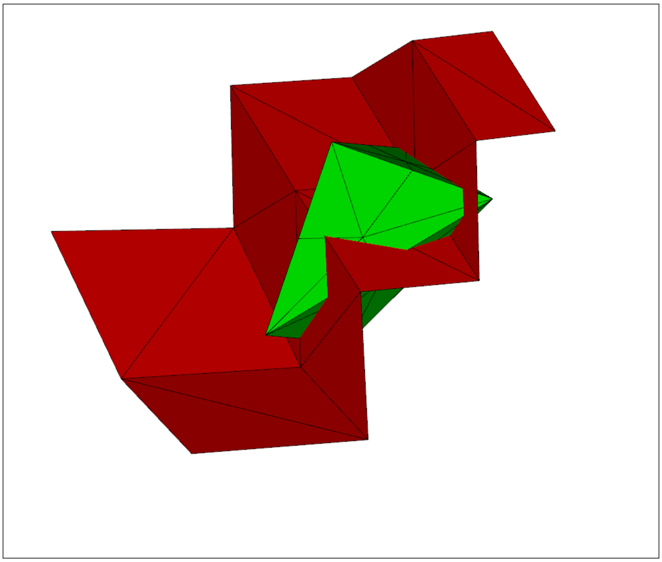

Fig. 2. A typical cut case between the dual of a tetrahedral element and a triangular interface surface

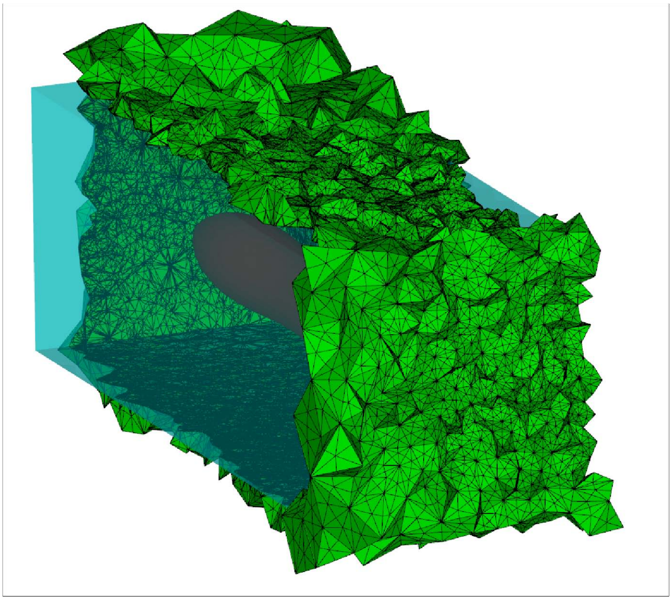

Fig. 3. Result of a second cutting operation where the background dual mesh has been cut with the outer block surface remaining after the first cut

robustness of the creation of the consistent dual mesh. An additional robustness issue is the stability of the solver on the dual mesh resulting from the DMC procedure.

For the DMC algorithm, there are essentially two parts which involve floating–point operations, namely the cutting of the subtetra with the interface triangles and the construction of a volume tesselation of the resulting hull. These two potential problems are solved in the following way in the current implementation:

1. To avoid inconsistencies in the cutting operations, each floating–point computation is performed only once. Neighbouring subtetra and elements therefore must be informed of pre–computed cutting operations. Since not only the coordinates, but also the type of cuts, with auxiliary data, are stored, the topologies can be checked for inconsistencies during the cutting. If a topology results which is not mathematically possible, optional topologies are considered. This is performed through the introduction of what has been termed 'tolerance nodes'. These nodes appear in cutting cases where, due to the limited accuracy of floating point operations, it is unclear whether a cut is present or if a cutting face node is within the subtetra in question. The various combinatorical possibilities of introducing the various tolerance nodes are then considered until a configuration with a valid topology is found. Even though it has yet to occur in the test examples, it is theoretically possible that even this procedure fails. In such cases, a method has been implemented

in which the cutting surface is perturbed by a very small amount and the cuts are re–computed. Since cutting faces may also lie on the geometrical surface of the configuration, it is possible that very slight changes are introduced in the geometrical representation of the case. However, in using double precision operations, the perturbations can be kept at a level much smaller than the surface mesh conformity of the geometry.

2. The cutting operations on the tetra result in one or more hulls made up of triangular faces. These hulls must be filled with tetrahedra for the generation of an explicit dual mesh representation in the interface regions. The filling operation does not need to consider the resulting quality of the tetrahedra generated. This increases the speed and robustness of the operation considerably. The current implementation consist of first attempting to fill the hulls by a simple and fast connection approach, in which seed nodes on the hull are selected and connected with as many hull triangles as possible. This approach has shown itself to work for around 99 percent of the hulls that originate from tetrahedral elements, and 95 percent of the hulls that originate from boundary layer prisms. If this approach fails, a node insertion approach is applied, in which new dual mesh nodes are introduced inside the hulls and connected with the hull triangles. In extreme cases, for example where the hull describes a flat volume, the point insertion strategy may also fail. If this is the case, an 'enforced closure' approach is applied to the hull. This approach is conservative and not critical for the generation of the dual mesh. The degenerated elements this procedure may introduce for the hybrid visualization mesh are also not of major concern since no numerical computations should be performed on this mesh.

The above procedure has been implemented into the Cassidian **SimServer** [7] multidisciplinary simulation environment, employing the DLR TAU code [8] as solver.

3 Examples

Around 15 testcases have so far been computed with the DMC approach. The meshes used are however kept relatively small since the parallelization of the software is not yet completed. The cases considered so far are mostly wing configurations in which mesh blocks containing spoilers slats and external fuel tanks are added in various positions. Both Euler and RANS computations have been performed and the configurations include intersecting geometries. Even though the code has not been tested to a level warranting clear statements of robustness, accuracy and speed, an indication of these characteristics can be extracted from the testcases presented here.

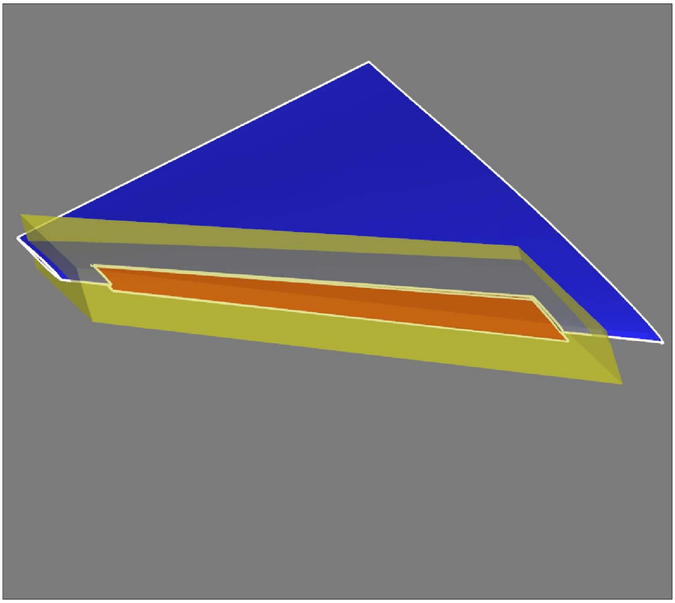

Fig. 4. Wing–slat configuration used for the Euler DMC computation. The outer block boundary for the slat component is shown.

3.1 Wing with Attached Slat

In this section the configuration shown in Fig. 4 is considered. The flow is inviscid with a Mach number of 0.5, the angle of attack 15 degrees and the slat deployment angle is 20 degrees. Details of the hybrid (plotting) mesh are shown in Fig. 5. A mesh convergence study has been performed on the configuration, in which mesh cascades for both consistent meshes and multiblock DMC meshes have been made. Every attempt has been made to obtain the same point distribution for each coarseness level in the cascades and the meshes have not been explicitly refined in the intersection regions. The plots for the convergence study is shown in Fig. 6. For the finest mesh level, a small oscillation in the wingtip region with an amplitude of around one percent of the integral values for both the consistent and DMC computations occurs. Here the average values have been taken for the mesh convergence analysis. It is seen that already for the coarsest meshes, the difference in the results between the consistent and DMC approaches is very small, and that the error decreases monotonously with the global mesh refinement. In Fig. 7, the surface pressures of the finest cascade level is shown.

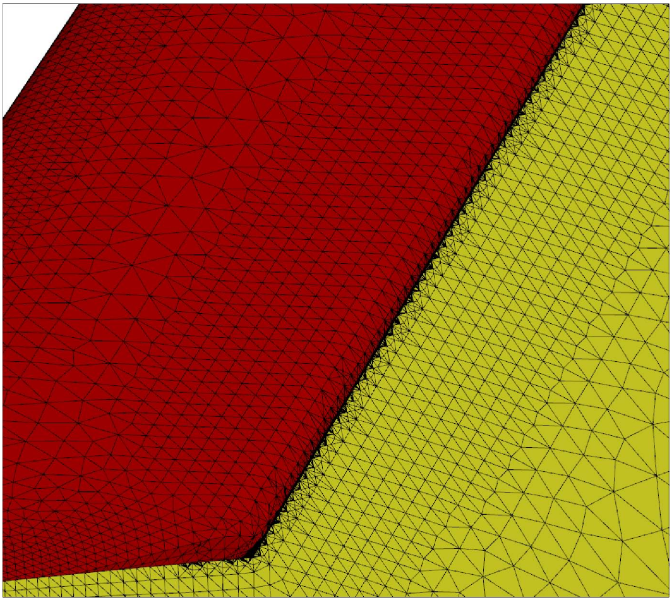

Fig. 5. Hybrid surface mesh detail for the wing–slat Euler DMC computation

3.2 Wing with Intersecting Tank

To demonstrate the DMC approach on a small RANS problem, a comparison between consistent and multiblock computations for the wing with tank configuration shown in Fig. 8 is presented. The Reynolds number is two million, the Mach number 0.5 and the angle of attack 5 degrees. The meshes consist of around one million nodes. In Fig. 9 the convergence of the two meshes is compared. It is observed that the DMC computation surprisingly converges somewhat faster, an effect that can probably be contributed to different coarse level multigrid meshes due to the different numbering of the dual mesh nodes and is not believed to be intrinsic to the DMC approach for general applications. The pressure distributions of the two computations is shown in Fig. 10.

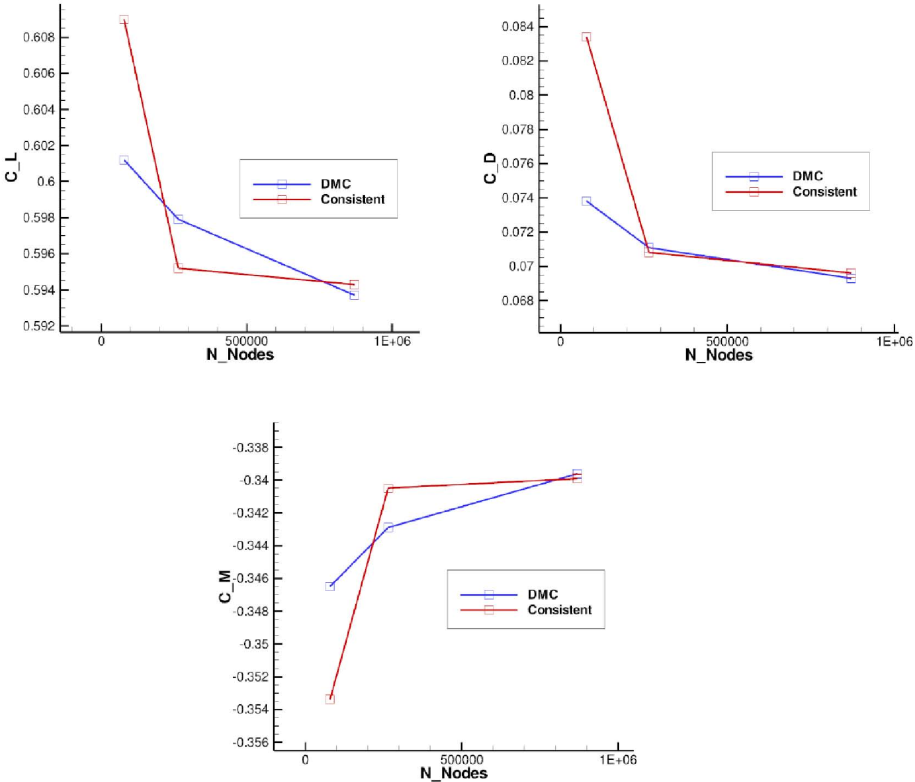

Fig. 6. Mesh convergence comparison for lift drag and pitching moment for the wing–slat Euler computations

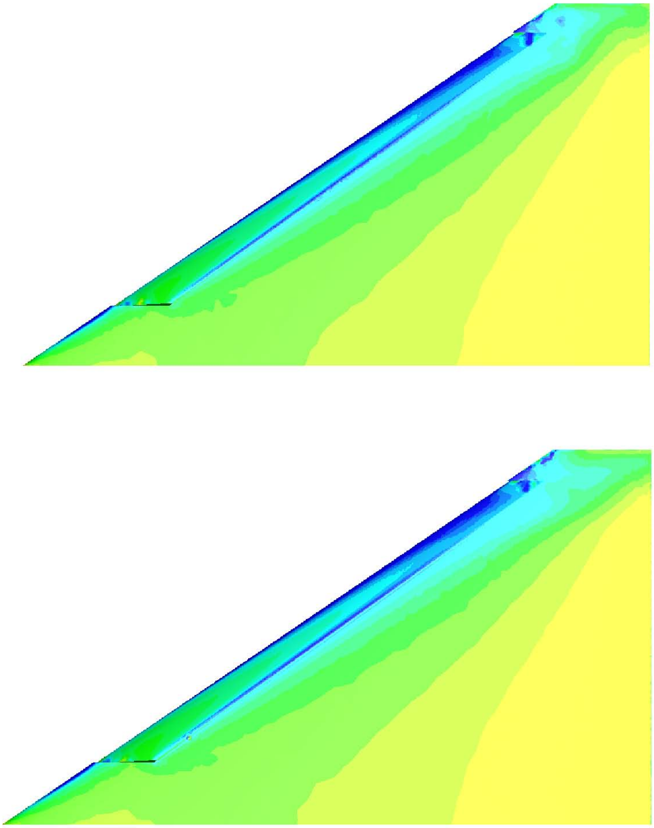

Fig. 7. Pressure comparison of the consistent (upper) and DMC (lower) wing–slat Euler computations

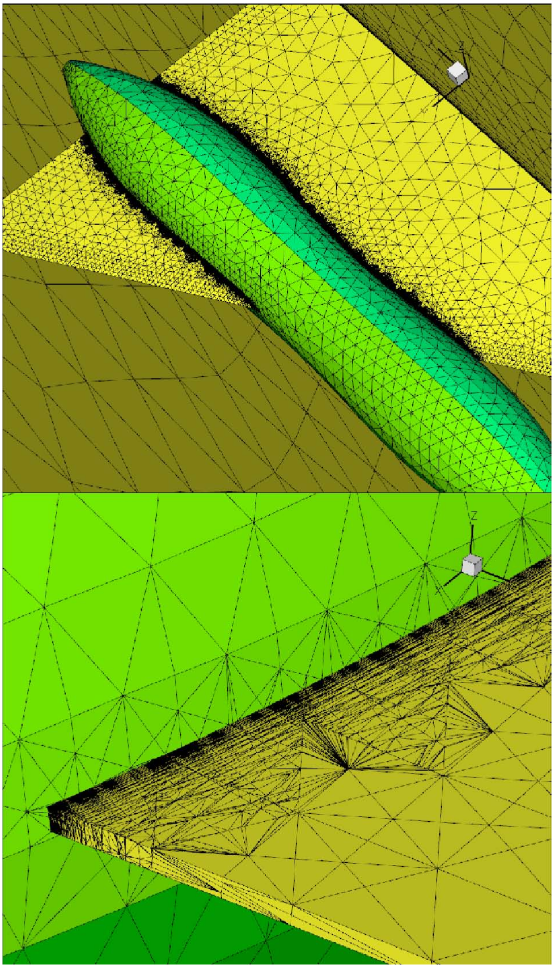

Fig. 8. Hybrid surface mesh for the wing–tank RANS DMC computation

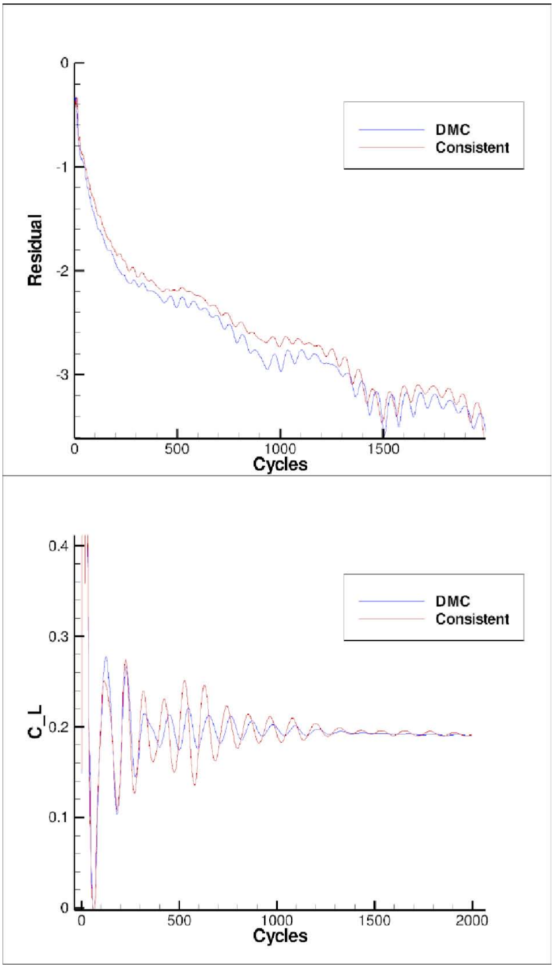

Fig. 9. Density residual and lift convergence of the wing–tank RANS computations

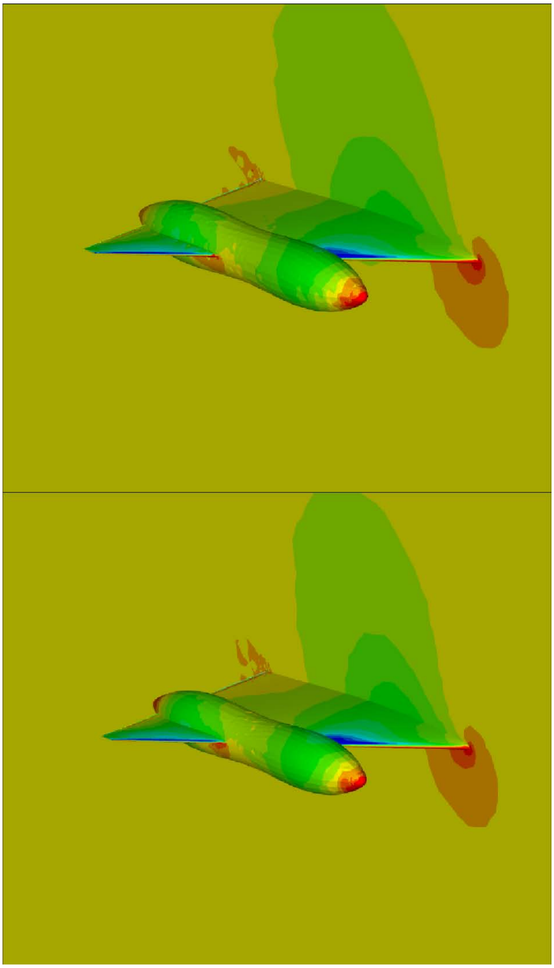

Fig. 10. Comparison between the pressure field of the consistent (upper) and multi–component DMC (lower) computations for the wing–tank case

4 Conclusions

A new approach for the multi–component modeling of configurations with intersecting surfaces has been presented. In the approach, a consistent dual mesh is created through cutting operations, resulting in a conservative numerical scheme. The current implementation has not yet reached a maturity level allowing for clear conclusions regarding the performance of the method, the examples computed thus far do however indicate that the approach could represent a very accurate and efficient way of simulating fluid flow on a wide variety of multi-component configurations of industrial complexity.

References

1. Tremel, U.: Parallel unstructured adaptive remeshing for moving boundary problems. Ph.D. Thesis, University of Wales, Swansea (2005)
2. Hassan, O., Sørensen, K.A., Morgan, K., Weatherill, N.P.: A Method for Time Accurate Turbulent Compressible Fluid Flow Simulation with Moving Boundary Components Employing Local Remeshing. Int. J. Numer. Meth. Fluids 52, 1243–1266 (2007)
3. Tremel, U., Hitzel, S.M., Sørensen, K.A., Weatherill N.P.: JDAM–Store Separation From an F/A–18C - An Application of the Multidisciplinary SimServer–System AIAA 2005–5222 (2005)
4. Steger, J.L., Dougherty, F.C., Benek, J.A.: Advances in grid generation, vol. 5. American Society of Mechanical Engineers (1983)
5. Sørensen, K.A., Tremel, U., Rieger, H., Hitzel, S.M.: Simulation of Manoeuvring Fighter Aircraft with the Unstructured Chimera Approach. AIAA 2007–0123 (2007)
6. Brenner, P.: Three Dimensional Aerodynamics with Moving Bodies applied to Solid Propulsion. AIAA 91–2304 (1991)
7. Tremel, U., Deister, F., Sørensen, K.A., Rieger, H., Weatherill, N.P.: The SimServer – a Parallel Multidisciplinary Simulation Environment. DGLS–2003–099, Munich (2003)
8. Gerhold, T., Friedrich, O., Evans, J., Galle, M.: Calculation of Complex Three–Dimensional Configurations Employing the DLR–Tau Code AIAA 97–0167 (1997)