ECCOMAS Congress 2016
VII European Congress on Computational Methods in Applied Sciences and Engineering
M. Papadrakakis, V. Papadopoulos, G. Stefanou, V. Plevris (eds.)
Crete Island, Greece, 5–10 June 2016

AN INTEGRATED FRAMEWORK FOR WRAPPING AND MESH GENERATION OF COMPLEX GEOMETRIES

David G. Martineau[1], Jeremy D. Gould[1], and Jacques Papper[1]

[1] ICON Technology & Process Consulting Ltd
Berkshire House, Thames Side Windsor, SL4 1QN
{d.martineau,j.gould,j.papper}@iconcfd.com

Keywords: Mesh Generation, Surface wrapping, Complex geometry.

Abstract. *This paper presents a combined mesh generation and surface wrapping capability which is able to create manifold watertight surfaces for complex industrial model assemblies. The wrapping functionality is able to join disconnected and overlapping surfaces. Large holes in the assembly are closed by performing the wrapping at intermediate stages of the mesh refinement. The ability to perform wrapping in combination with mesh generation ensures that holes are closed without loss of geometric fidelity. The wrapping capability is demonstrated on a range of test cases including an engine block and a highly detailed automotive case.*

1 INTRODUCTION

Thanks to significant advances in computational fluid dynamics (CFD) over the last few decades, and increasing availability of high performance computing resources, engineers are able to apply CFD to more and more complex flow simulations. However, robust and automatic mesh generation still represents a significant challenge in the overall process from CAD to CFD solution in industrial engineering analysis. A major bottleneck in the mesh generation process is the creation of a watertight representation of the geometry from the original CAD model. A watertight geometry representation is required as input to traditional structured and unstructured mesh generation methods, which first discretize the surface of the flow domain before meshing the interior. The translation of the geometry from the native CAD data into another format can often result in problems such as missing or duplicate parts, small gaps and overlaps [1]. Often the input to the simulation process is a model which has been designed for manufacture, not for the purpose of simulation, and therefore contains an unnecessary level of detail and geometry which should not be part of the fluid domain. Resolving these issues, and creating a geometry model suitable for mesh generation requires labour-intensive and time-consuming geometry repair and modification.

Mesh generation approaches such as octree or Cartesian-based methods [3, 4], which start with a volume mesh in the interior of the domain, and then obtain a boundary conforming surface mesh through cell-cutting or snapping procedures, have therefore become increasingly popular in recent years, in part due to their inherent tolerance to poor quality or “dirty” geometry [5]. However, although such meshing approaches do not need a watertight geometry representation, and are able to tolerate geometry issues such as overlaps and poor quality triangulations, they cannot handle so-called ‘fully-resolved gaps’, i.e. gaps in the geometry larger than the local element size [5]. There have been numerous attempts over the last decade to develop methods for automatic repair of holes in discrete geometry models. These methods broadly fall into two main categories: surface-based approaches and voxel or volume-based approaches.

Surface-based approaches operate directly on the discrete geometry input data and resolve defects in the geometry such as holes with open boundary edges. The closing of such holes is a relatively mature field of study, and there are various algorithms described in the literature [6-10]. Holes in relatively planar regions of the model can be easily patched via planar triangulation. To handle more complex holes, Jun [6] proposes a piecewise scheme which divides them into several simpler holes which are filled with a planar triangulation, before applying smoothing and sub-division techniques to improve the quality. Zhao et al [7] employ an advancing front method to close complex holes, and then solve the Poisson equation to reposition the new vertices. Alternative algorithms use radial basis functions [8] or the Level Set Method [10] to construct an implicit surface which covers the hole. Since surface-based methods operate only locally, they can be very efficient and incur only minimal perturbation of the input data. However, they can only close holes with an open boundary, and are not able to handle semantic holes caused by missing components, or gaps between parts in a model assembly.

Volume-based approaches operate by generating a volumetric representation of the surface, and then classifying the cells in the volume as either inside, outside, or intersecting the surface. The shrink wrapping technique developed by Lee et al [1] is one such example of this approach, and is based on the concept of the interior-to-boundary Cartesian mesh generation described by Wang [5]. Volume approaches such as this are typically fully automatic and can be very robust. However, large holes in the geometry can lead to mesh ‘leaking’ into areas of the domain which should not be modelled. Juretić and Putz [12] identify the holes leading to such mesh leaks by solving the heat diffusion equation and looking at regions of high heat flux intensity. However it is left to the user to then manually close these holes and repeat the process until no leaks are present. Kumar and Shih [13] propose a hybrid method, which applies surface-based repair techniques to close holes with simple topologies, and then obtains a watertight surface using the Marching Cubes algorithm [14] to extract the zero-set surface of the numerical solution to the diffusion equation. Whilst they demonstrate good quality results on models with complex topology, their approach relies on consistent orientation of the input geometry, and is not able to deal with semantic holes in an assembly.

This paper describes an integrated approach to wrapping and mesh generation implemented in iconCFD® [15], which is able to handle large gaps in model assemblies, including holes with open boundaries and semantic holes, without significant loss of geometric fidelity. The problems of resampling a geometry (e.g. loss of model features) that are typically encountered with volume-based geometry repair methods are also avoided in our approach by exploiting an adaptively-refined Cartesian grid to simultaneously perform the wrapping of the geometry and generate a boundary conforming, hexahedral-dominant mesh of the flow domain. Section 2 of this paper describes the various stages of the existing mesh generation process which formed the basis for this work. The extension of this capability to include surface wrapping is then described in section 3, and in section 4 the two different modes of operation are described. In section 5 the capability is applied to a number of models to demonstrate the ability of the approach to handle industrial geometries used in external aerodynamic and under-hood thermal management simulations. Finally, the paper finishes with conclusions and a discussion of directions for future work.

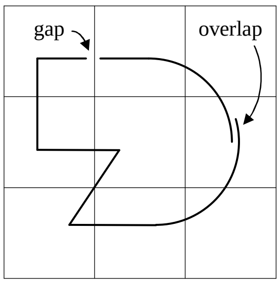
2 OVERVIEW OF MESHING APPROACH

The wrapping capability has been implemented within an existing parallel hexahedral-dominant mesh generator, iconHexMesh, which is part of the iconCFD process, an open source-based CFD software suite developed by ICON, using OpenFOAM® [16] technology. The mesh generation process in iconHexMesh comprises three main stages: (1) refinement of the initial block mesh based on prescribed refinement levels on geometry surfaces and feature lines, (2) creation of a conformal mesh by snapping to geometry surfaces, and (3) insertion of a layer mesh to capture viscous boundary layers. At various points during the mesh generation, the mesh is dynamically re-partitioned to achieve good load-balancing. The refinement and snapping stages are common requirements for both meshing and surface wrapping, and are therefore described in more detail in the following sections.

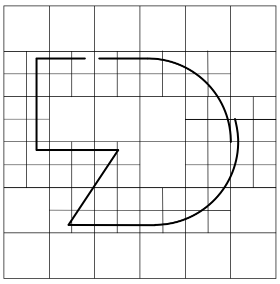
2.1 Refinement

The input to the meshing process is a discretized model of the geometry. In the automotive industry, the STL (stereo lithography) format is commonly used as input to the simulation process as it is a simple portable format which can be generated by most CAD systems and is easy to visualize [5]. The meshing process begins by reading one or more STL files describing the assembly to be meshed, and creates an initial coarse mesh of the domain, typically using a uniform grid of Cartesian-aligned cells which completely encloses the entire geometry, as illustrated in Figure 1(a). Successive iterations of local mesh refinement are then applied, until the required level of mesh refinement is reached (Figure 1(b)). The mesh generator supports a number of different refinement criteria to capture the details of the geometry, including:

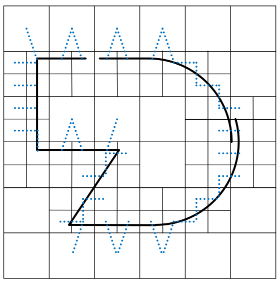
- Uniform refinement of geometry surfaces and feature lines
- Curvature-based surface refinement
- Proximity-based refinement of surfaces and feature lines
- User-defined volumes of refinement



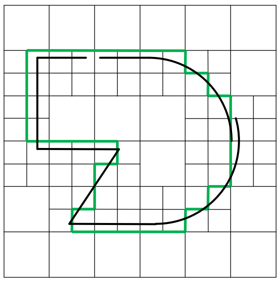
(a) Input 'dirty' geometry with initial mesh



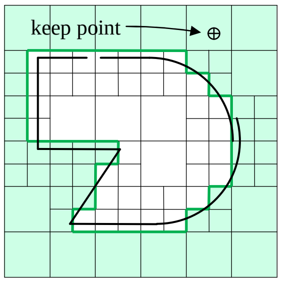
(b) Refined mesh



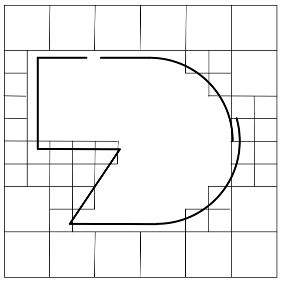
(c) Dual-edges intersecting the geometry



(d) Marked boundary faces



(e) Cells connected to specified keep point



(f) Cells corresponding to fluid domain

Figure 1: Illustration of the refinement process in iconHexMesh.

Once the refinement criteria have all been met, 'dual-edges' of the mesh are tested for intersection with the geometry (Figure 1(c)). A 'dual-edge' corresponding to an interior face of the mesh is the line joining the centres of the two cells neighbouring the face. If this edge intersects the geometry, then the face is marked as a boundary face (Figure 1(d)). By painting cells connected to one or more user-specified keep points without crossing boundary faces, the fluid region to be kept is identified (Figure 1(e)). Any cells on the other side of the set of boundary faces from the keep point are then discarded (Figure 1(f)).

2.2 Snapping

The outer surface of the remaining mesh, which will be castellated at this stage, is then snapped to the geometry to provide a body-conforming volume mesh. This process involves a combination of smoothing of the nodes on the boundary, and projection to the geometry, and as such shares a degree of similarity with the shrink wrapping approach described by Kobbelt et al [17]. A full description of the snapping algorithm is beyond the scope of this paper, but the overall process is summarized as follows:

1. Apply a few iterations of global Laplacian smoothing to the initial castellated surface. This helps to ensure that the subsequent projection onto the geometry does not result in self-intersection of the mesh.
2. Project selected nodes in the surface mesh to feature lines in the geometry. These feature lines are automatically detected based on discontinuities (i.e. sharp edges) in the discretized geometry model, or region boundaries. For each feature line, a

corresponding path of connected nodes is found in the surface mesh, which is then projected to the feature line. The deformation field arising from this projection is smoothed into the surrounding mesh, minimizing the distortion of the surrounding mesh elements.

3. Project all remaining boundary nodes to the geometry. Rather than simply projecting each node to the closest point on the geometry, this process is performed iteratively, and attempts to gradually move and rotate the faces of the surface mesh to match the local normal direction of the geometry.
4. Smooth the final surface locally where the distance from the face centre to the geometry is large. This relaxation of the surface mesh where the snapping has failed to accurately capture the geometry improves the local quality of the surface mesh, and aids in the subsequent insertion of layer mesh.

Throughout the snapping process, the geometric quality of the volume mesh is monitored. Any deformation which would result in degradation of the mesh quality below user-specified limits is locally scaled back to ensure that the mesh quality constraints are satisfied. Similarly, any topological changes to the mesh which could violate these mesh quality constraints can be reversed to guarantee a resulting volume mesh of acceptable quality.

3 WRAPPING IMPLEMENTATION

3.1 Improving tolerance to gaps

The method by which the boundary faces of the mesh are determined in iconHexMesh is quite different to conventional Cartesian and octree mesh generation. Typically, Cartesian or octree mesh generation defines the boundary faces as the exposed faces remaining after removal of the intersected cells and cells outside the flow domain, as illustrated in Figure 2(c). The advantage that the approach described in section 2.1 has over this latter approach is that thin non-manifold surfaces in the geometry can be modelled with a ‘sheet’ of faces in the volume mesh. In general, the approach allows the topology of the initial set of boundary faces to more closely match the topology of the geometry, as shown in Figure 2(b). This enables much more accurate capture of the non-manifold parts of the geometry, such as the open edges of thin surfaces.

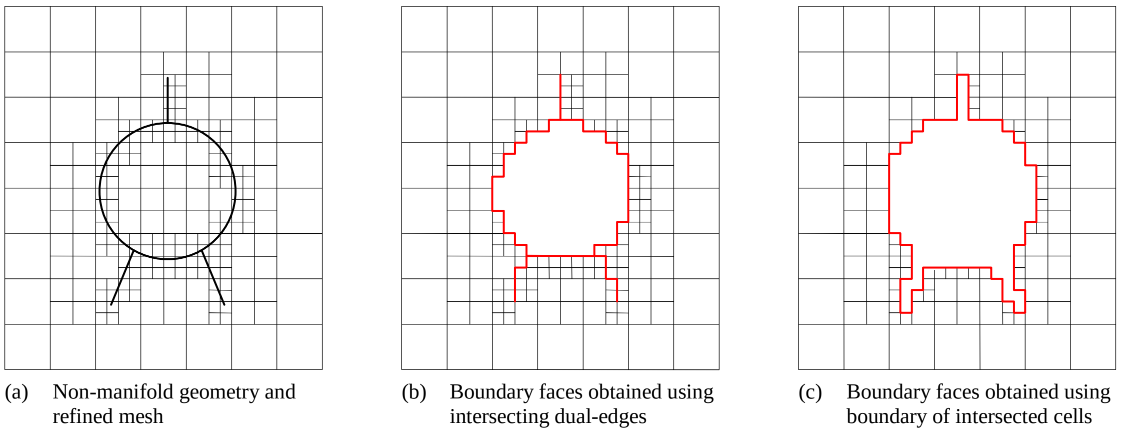
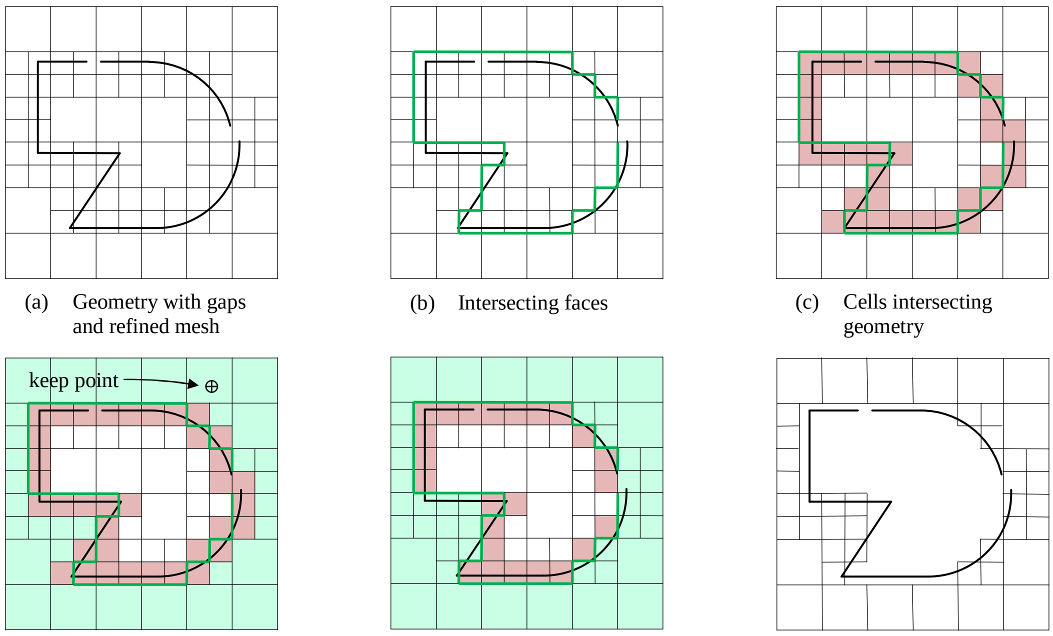

(a) Non-manifold geometry and refined mesh (b) Boundary faces obtained using intersecting dual-edges (c) Boundary faces obtained using boundary of intersected cells

Figure 2: Capture of thin surfaces.

There is a disadvantage to the ‘dual-edge’ intersection method for defining boundary faces however. With this approach, there exists the possibility that a dual-edge passes through a gap in the geometry, thus leaving a hole in the boundary faces which permits the volume mesh to ‘leak’ into parts of the volume which are not intended to be part of the flow domain.



(a) Geometry with gaps and refined mesh
(b) Intersecting faces
(c) Cells intersecting geometry
(d) Marked cells connected to keep point
(e) Re-assignment of cells to keep region
(f) Mesh corresponding to flow domain

Figure 3: Modified process for identification of fluid region.

This limitation is overcome by combining the existing method for detecting boundary faces with the more conventional method. The new method is depicted in Figure 3 and starts, as in the original method, with marking of faces whose dual-edges intersect the geometry. These are defined as ‘intersecting’ faces. The nodes belonging to these intersecting faces are also marked as boundary nodes. The algorithm then marks all cells which intersect the geometry, and paints all the remaining cells connected to the keep point (the keep region). The intersecting cells which share a non-boundary node with the cells in the keep region are added to this region, and finally all cells not in the keep region are discarded. The boundary faces are then defined as the faces on the outer surface of the keep region.

Although the modified process described above improves the tolerance to dirty geometry, it cannot handle the situation of fully resolved gaps, as discussed in [5], where the size of the gap is larger than the local element size. Unfortunately, as computing resources increase and engineers attempt to improve the accuracy of flow simulations by resolving finer and finer detail, the possibility of creating a fully resolved gap becomes more and more likely. In the automotive industry for example, small gaps between body panels or paths through the heating, ventilation and air-conditioning (HVAC) system can connect the region outside the car to the region inside the car cabin. Although tools are available in iconCFD® to detect and visualize leaks in a volume mesh, this still requires labour-intensive visual inspection and manual repair of the model, and prevents the formation of a fully automated analysis process.

3.2 Wrapping at coarser levels

To address the issue of fully resolved gaps in the meshing process, an iterative gap-filling methodology is proposed. The iterative gap-filling methodology starts by applying the modified boundary identification process, described in the previous section, at an intermediate stage of the refinement process, as illustrated in Figure 4(a). This intermediate stage of the refinement process is achieved by refining the mesh until a user-defined refinement level has been reached on all surfaces of the geometry. This is referred to as the wrap level for a surface and is either equal to or less than the maximum refinement level associated with a surface. If the wrap levels have been set correctly, then the set of boundary faces identified in this coarse mesh should form a closed surface enclosing the region to be modelled.

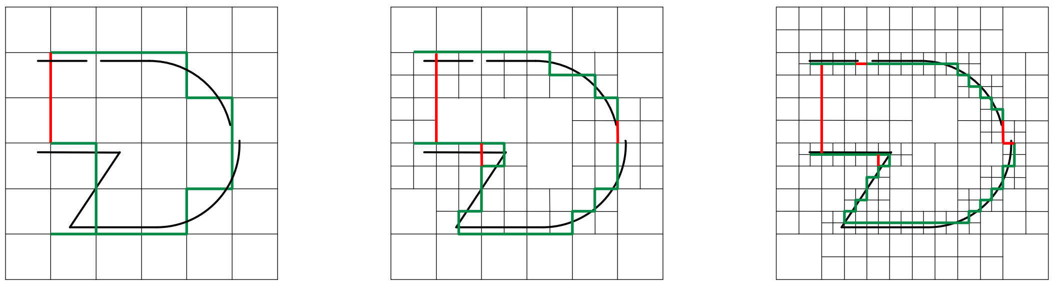

(a) Gap faces (red) identified at level 1 (b) Gap faces (red) identified at level 2 (c) Gap faces (red) identified at level 3

Figure 4: Iterative identification of gap faces during refinement process.

Having created a closed set of boundary faces, a subset of these faces which close the gaps in the geometry can be identified as those whose corresponding dual edge does not intersect the geometry. The refinement process now continues, with the mesh being refined from the wrap level to the maximum level required. The set of gap faces is updated during the mesh refinement to take into account changes to the mesh topology. At each step in the refinement, the process of identifying the boundary faces must be repeated as shown in Figure 4(b) and (c), since the additional refinement may otherwise cause a gap to be created.

The set of gap faces is also filtered at each step of the refinement process, to minimize the area of the faces closing a hole and improve the quality of the final wrapped surface.

3.3 Surface snapping

After the refinement process, and removal of the mesh outside of the flow domain, the boundary nodes of the mesh are snapped to the surface of the geometry as described in section 2.2. However, in the original implementation, it could be assumed that all the boundary nodes were within a relatively small distance from the geometry. With the creation of gap faces spanning potentially large gaps in the geometry, this assumption is no longer valid. However, by maintaining a list of mesh faces corresponding to the gaps in the geometry, it is possible to apply a special treatment to these nodes. Firstly, more aggressive smoothing is applied to the gap faces to improve the quality of the surface mesh in these regions. After this initial smoothing, and snapping to feature lines on the geometry, the dual-edges corresponding to the gap faces are tested for intersection with the geometry. Any intersection of these dual edges indicates that the gap face likely corresponds to a crack in the geometry, rather than a fully resolved gap, and is therefore un-marked as a gap face. During the rest of the snapping process, the displacement of the boundary nodes to conform to the geometry is propagated to the remaining gap faces from the neighboring boundary nodes.

Once the snapping stage has been completed, the gap faces are assigned to neighbouring patches. This allows a consistent layer mesh to be inserted on boundary patches, irrespective of the presence of any cracks or gaps.

3.4 Pure Wrapping Mode

If a wrapped surface is the only output required, then the quality of the volume mesh can be disregarded. This allows for a much faster snapping process, since no checks on the volume mesh quality need to be performed, and there is no need for scaling of the mesh deformation when moving the boundary points onto the surface of the geometry.

Finally, in pure wrapping mode, the surface mesh is triangulated and output to STL format, preserving any patch IDs assigned to the original geometry. The gap faces can either be assigned to a separate patch, or assigned to the nearest adjacent patch.

4 RESULTS

4.1 Flange

To demonstrate the ability to wrap holes larger than the local element size, the wrapping capabilities of iconHexMesh are applied to the Flange geometry from the OpenFOAM® tutorial [18]. This is a simple geometry, as shown in Figure 5(a), which features several holes and recesses of varying diameter. An initial uniform Cartesian mesh of element size 0.02 is created which encloses the geometry, as shown in Figure 5(b). The iconHexMesh mesh generator is then used to apply 4 levels of surface refinement, resulting in the final surface mesh shown in Figure 5(c), which accurately captures the sharp features of the geometry.

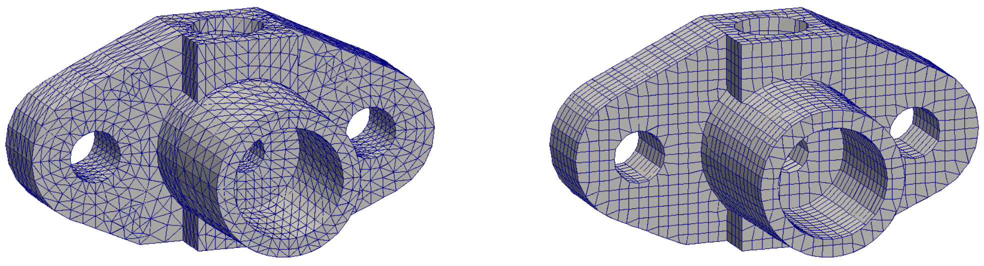

(a) STL geometry (b) Surface mesh

Figure 5: Flange geometry and corresponding surface mesh.

The series of images in Figure 6 show how iconHexMesh is able to close off different size gaps in the geometry using varying wrap levels, but with the same final refinement level in all cases, enabling the geometric fidelity of the model to be maintained as much as possible. In contrast, Figure 7 shows how use of varying refinement level to close off the holes in the mesh results in considerable loss of geometric fidelity.

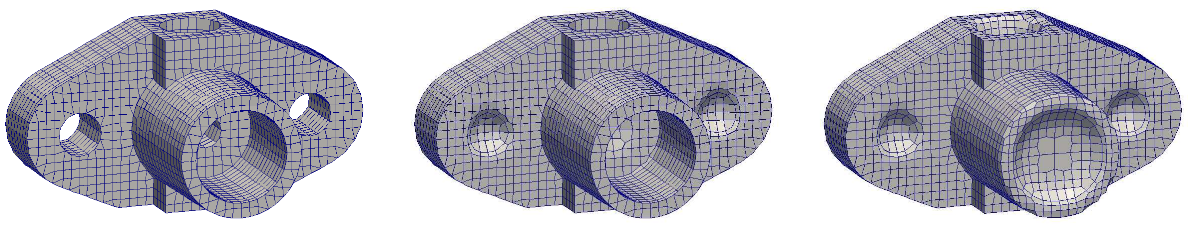
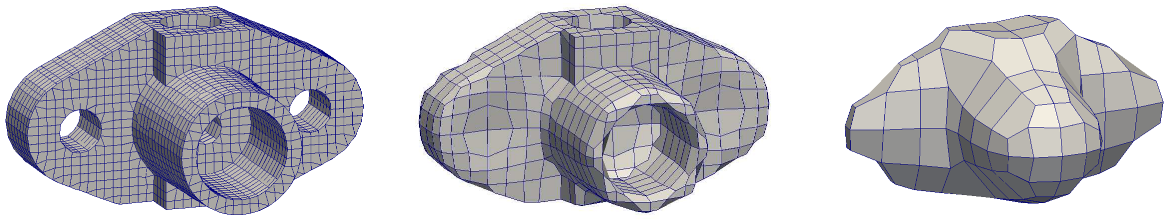

(a) Wrap level = 4 (b) Wrap level = 3 (c) Wrap level = 2

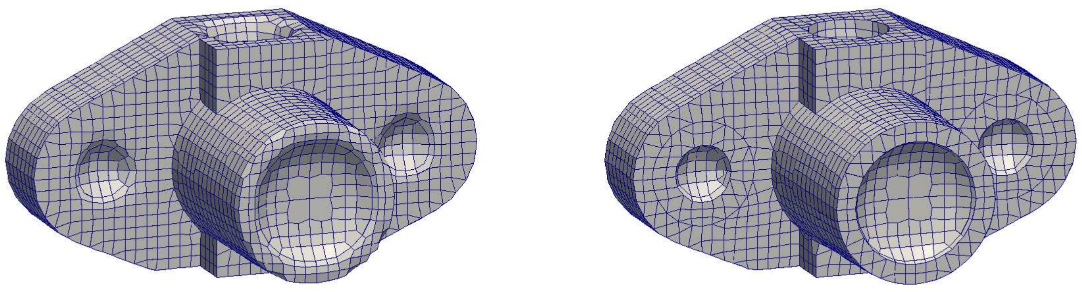
Figure 6: Flange surface mesh at different wrap levels.



(a) Refinement level 4 (b) Refinement level 3 (c) Refinement level 2

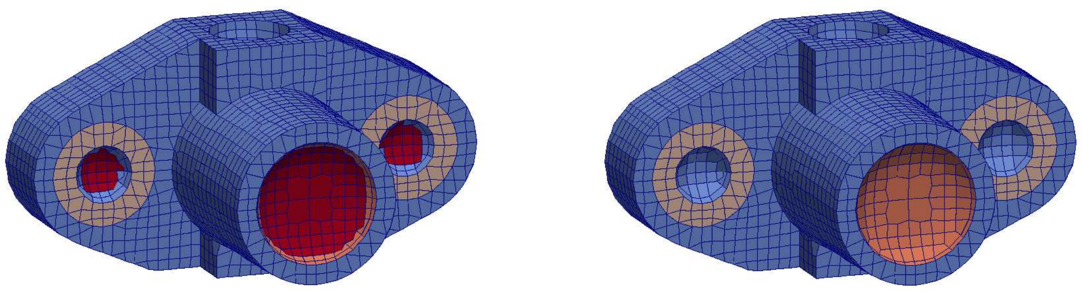
Figure 7: Flange surface mesh at different refinement levels.

The geometric fidelity of the surface mesh can be further improved by enabling the feature line snapping functionality in iconHexMesh, as illustrated in Figure 8. The feature line snapping is able to improve the capture of sharp edges on the geometry, as well as properly resolving the region boundaries on the flat surface of the model. Finally, Figure 9 shows how the gap faces can be kept in a separate patch, or assigned to the same patch as neighbouring faces in the surface mesh.



(a) Surface mesh with basic surface snapping (b) Surface mesh with feature line snapping

Figure 8: Improved geometry capture using feature line snapping.



(a) Gap faces (red) put in separate patch (b) Gap faces assigned to neighbouring patches

Figure 9: Surface mesh shaded by patch ID.

4.2 External Aerodynamics Case

In this section a few holes have been deliberately added to a model of the Koenigsegg Agera hypercar to illustrate how the combined wrapping and meshing capability is able to handle geometry defects commonly encountered in automotive applications. The geometry of the car is shown in Figure 10.

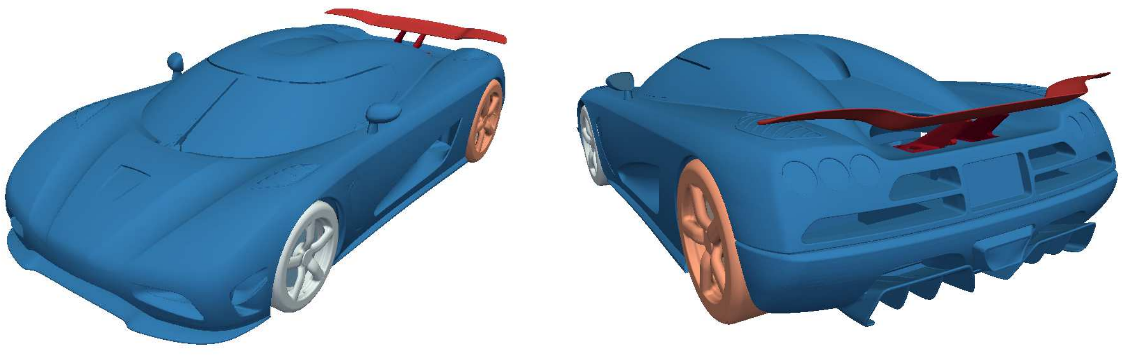


Figure 10: Koenigsegg Agera geometry.

To create a mesh on the Koenigsegg Agera vehicle, an initial Cartesian mesh was generated around the car and wind tunnel with a base size of 3 m. A minimum refinement level of 8 was applied to all of the car surfaces, giving a maximum element size of approximately 12 mm on the car surface. Between one and three levels of curvature refinement were applied to capture the feature edges and highly curved regions of the car, such as the rear wing blade, leading to a minimum element size of 1.5 mm. To capture the viscous boundary layer, 6 layers of cells are inserted at the car surface, with a first layer thickness of 1 mm.

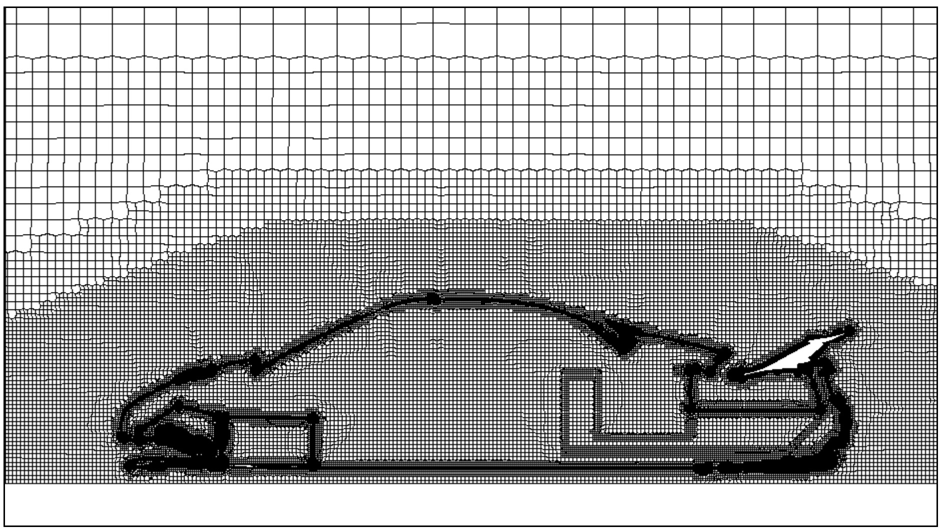


Figure 11: Slice through volume mesh with leakage into cabin region.

A slice through the volume mesh generated without wrapping is shown in Figure 11, clearly indicating that mesh has leaked into the car interior, resulting in a mesh which is not suitable for an external aerodynamic flow simulation. A flow simulation on this mesh will waste valuable computational resources, and may fail to converge due to the stagnant region inside the car.

A leak detector utility is provided within the iconCFD software suite, which can be run in order to help find gaps in geometries in this type of situation. Figure 12 shows a path (in red) of cells detected using this tool which connect a user-specified orphan point inside the car to the keep point. It can easily be seen in this case that the leakage path corresponds to a gap around the wind screen highlighted in blue in Figure 12. However, there may be many leaks in the geometry and the process of finding and closing them all can be slow and labour-intensive on a complex industrial model.

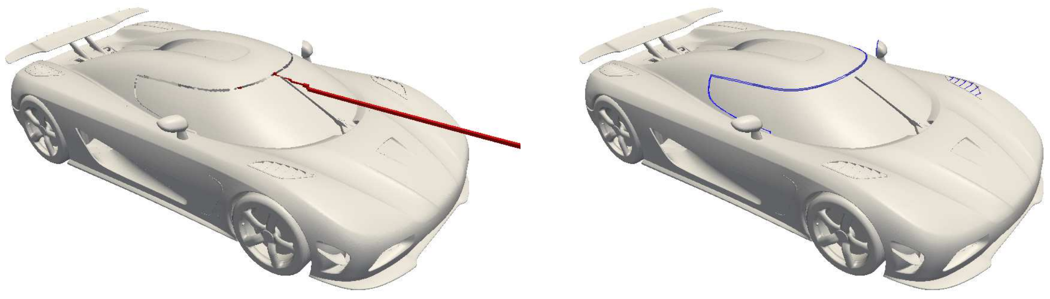


Figure 12: Leakage path detected (red) and corresponding hole in geometry (edges in blue).

In order to close the holes in the geometry without the need for any leak detection and manual repair, a wrap level of 7 was applied to all the car surfaces during the meshing process. This should close any holes of size 23 mm or smaller. All other meshing parameters were kept the same. Several views of the gaps in the geometry and the surface mesh generated in these regions are shown in Figure 13. A slice through the resulting surface volume mesh is shown in Figure 14, indicating that the wrapping has managed to successfully close all the gaps and prevent the mesh leaking into the interior.

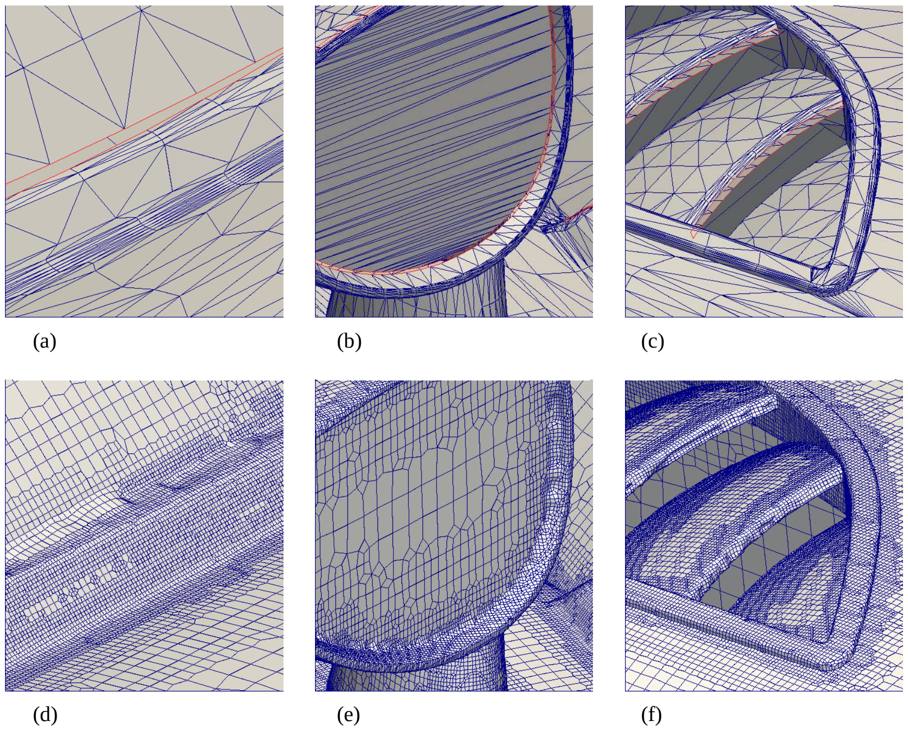


Figure 13: Details of car geometry (a), (b), (c) and corresponding surface mesh (d), (e), (f).

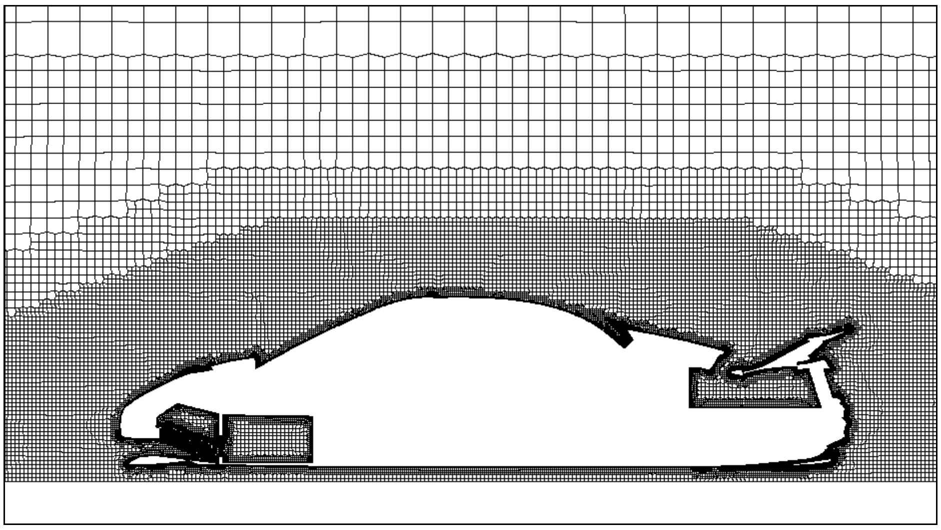


Figure 14: Mesh of car with wrapping activated.

Table 1 gives some statistics of the meshes obtained with and without wrapping activated during the meshing process. These indicate that the mesh leakage into the cabin incurs a 79% increase in element count, which could result in a significant waste of computational resources if not detected early in the analysis process. Both meshes were generated using 32 cores of a Linux cluster with Intel Xeon E5-2670 (2.60GHz) processors.

	Mesh 1 (unwrapped)	Mesh 2 (wrapped)
Time taken (s)	6,593	6,907
Number of cells	66,826,035	37,366,550
Number of faces	215,589,109	120,713,107
Number of points	83,252,580	46,732,723

Table 1: Comparison of mesh statistics.

4.3 Engine Block

In the automotive industry, under-hood thermal management (UHTM) deals with the design and verification of critical components in the engine compartment to ensure proper cooling and acceptable operating temperatures in a wide range of driving conditions. Although accurate capture of the highly detailed geometry in the engine compartment is not important for UHTM simulations, the full car geometry must be included, since the blockage effects of the engine and other components must be modelled properly. The engine geometry is however, highly detailed, and manual cleanup of the CAD geometry to provide a watertight surface can be very complex and time-consuming [19]. For these reasons, surface wrapping tools are often employed in the automotive industry to wrap engine geometries prior to mesh generation.

This test case therefore demonstrates the use of iconHexMesh in pure ‘wrapping’ mode, to create a watertight surface wrap of a complex engine block geometry. The model, shown in Figure 15(a), comprises 887 separate components grouped into 57 solids defined with a total of 4.2 million triangles. An initial Cartesian mesh was created with an element size of 1.25 m. A uniform surface refinement of level 9 was applied to the engine, giving a smallest element size of 2.44 mm. A wrap level of 5 was applied to fill any holes larger than 39 mm, resulting in the surface wrap shown in Figure 15(b). The entire wrapping process, including projection of the wrap surface to the geometry and capture of feature lines, took 278 seconds on a Linux workstation using 2 Intel Xeon X5650 (2.67GHz) processors. The final wrapped surface consists of a single closed manifold surface containing 762,028 triangles.

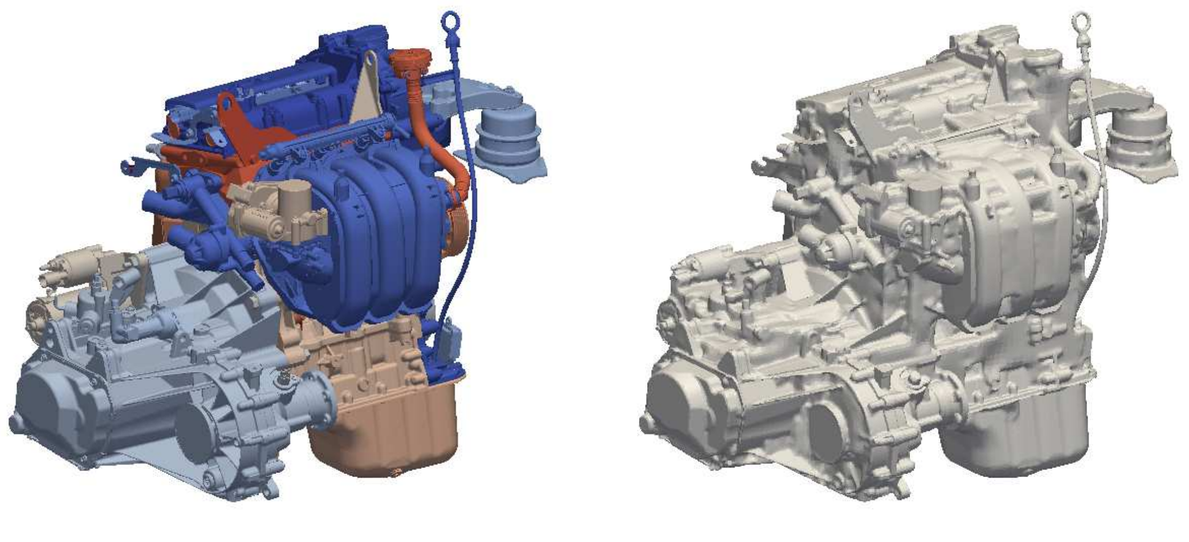

(a) Engine geometry shaded by PID (b) Watertight surface wrap of engine

Figure 15: Engine geometry and wrap.

In the detailed view of the wrap surface shown in Figure 16, the ability of the wrapping functionality to close large holes in the model, whilst still capturing fine details, is clearly demonstrated. A large circular hole in the geometry can be seen in the centre of Figure 16(a), which the wrap surface closes off in Figure 16(b), whilst still capturing the thin plate and narrow rod structures nearby.

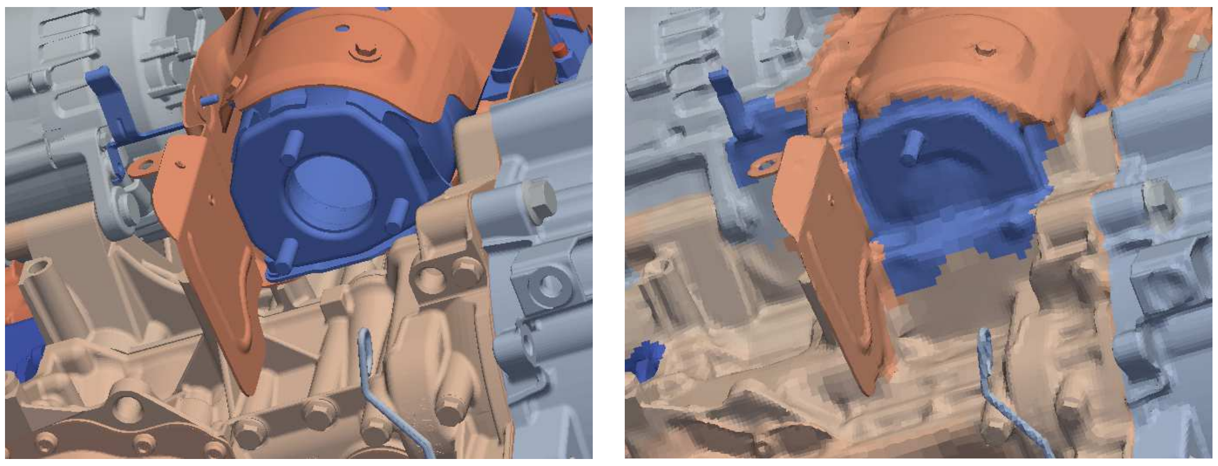

(a) Engine geometry (b) Surface wrap

Figure 16: Detail of engine geometry and wrap, shaded by PID.

4.4 UHTM Case

The final example in this section is used to demonstrate the wrapping capability on a fully complex industrial application. The model is a detailed geometry of the Skoda Fabia II containing the complete geometry of the engine compartment including power-train, suspension, cooling, exhaust system, and electrical components. The geometry consists of 14 STL files, which together comprise 382 patches and 36 million triangles. A clip through the geometry is shown in Figure 17, which illustrates the complex topology of the engine compartment. Manual clean-up of such a geometry and subsequent processing to allow volume mesh generation can involve several man weeks of CAD repair and preparation [17].

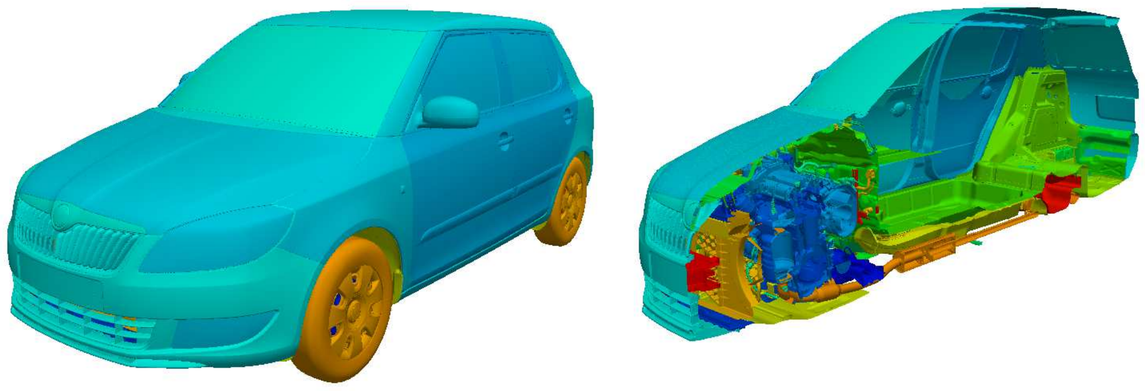


Figure 17: Geometry and clip through geometry of UHTM model

The 'pure wrapping' mode was used to generate a watertight surface wrap of this model suitable for subsequent analysis. Firstly, an initial background mesh of element size 0.625 m was generated enclosing the entire car geometry. A wrap level of 4 was applied to the entire geometry to close any holes smaller than 40 mm in the assembly. The background mesh was refined to level 8 (2.44 mm) on all surfaces, and an additional 2 levels of curvature refinement were applied to provide accurate capture of geometry features. The resulting surface wrap is shown in Figure 18. A slice through the background mesh in Figure 19 shows that the wrapping capability has managed to close any gaps which connect the outside of the car to the cabin.

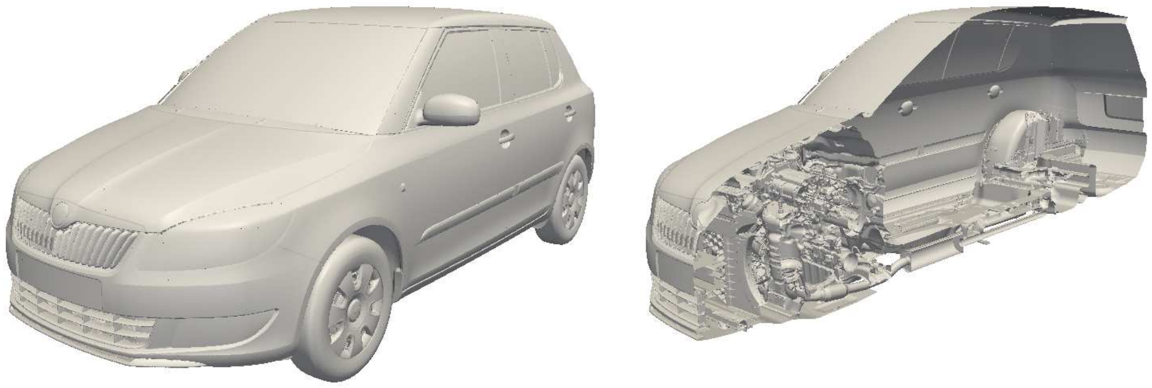


Figure 18: Wrap and clip through wrapped surface of UTHM model

The wrapping process took 2 hours and 42 minutes on 32 cores of a Linux cluster with Intel Xeon E5-2670 (2.60GHz) processors. The final wrapped surface consists of 12.5 million triangles.

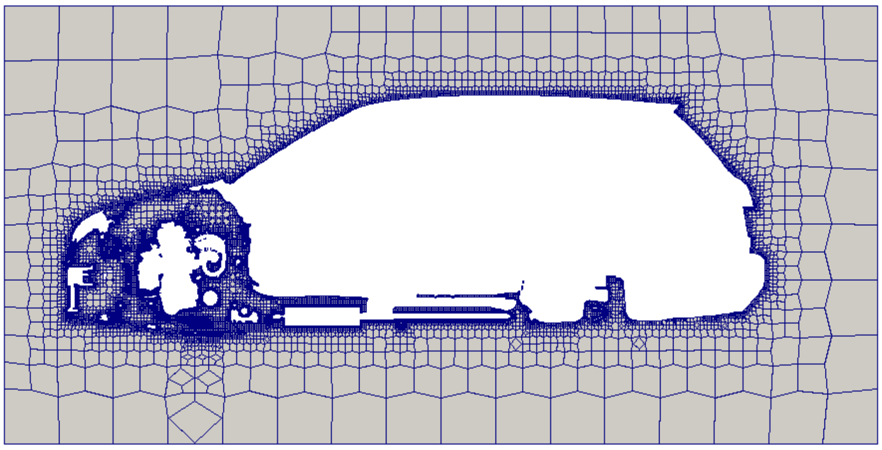


Figure 19: Slice through background mesh on UHTM model

5 FUTURE WORK

Although the combined wrapping and meshing approach presented here offers a significant improvement in automation of the simulation process, the user is still required to prescribe suitable refinement levels for the wrapping based on an upper bound on the size of any gaps which need to be filled in the geometry. Determining the size of gaps in the geometry is not a trivial task, and in future it would be preferable to remove this burden from the user. Instead of specifying wrap levels, it is envisaged that the user specifies one or more orphan points, similar to the keep points, which would indicate volumes of the model which are not to be included in the simulation. As the refinement process progresses, as soon as a path is created connecting an orphan point to a keep point, the refinement level at which wrapping is required can be identified.

In order to help keep selected components separate during the wrapping and meshing process, proximity refinement can be employed. However, this is currently applied between the selected surface and any other surface. In future, this will be extended to allow users to specify proximity refinement between selected pairs of surfaces and therefore enable highly focused contact prevention.

The primary focus of this work has been the combined wrapping and meshing functionality within iconCFD. However, as mentioned previously, it is possible to operate the mesher in a pure 'wrapping' mode, and output the resulting surface mesh as a modified set of STL files. In this scenario, it may be desirable to further improve the quality of the wrapped surface for the upstream applications. Topological operations such as edge swapping, edge splitting and edge collapsing [20], as well as node smoothing [20] are a few ways in which to achieve better surface mesh quality. If the size of the wrapped STL is a concern, then there are various surface mesh decimation techniques [21] which could be employed to reduce the number of triangles. There is also potential to improve the performance of the pure wrapping mode, by avoiding certain operations which are only required when the quality of the resulting volume mesh is of importance.

6 CONCLUSIONS

- A combined wrapping and mesh generation capability has been described which allows the original geometry to be retained to whatever fidelity is required in the simulation. This capability eliminates the de-featuring of geometry associated with conventional surface wrapping techniques.
- The problem of fully-resolved gaps encountered in other interior-to-boundary mesh generation approaches is avoided by wrapping at intermediate refinement levels during the meshing process.
- Implementing wrapping within the framework of an existing parallel mesh generation process means that wrapping can be performed quickly and efficiently with minimal overhead. The wrapping is also able to re-use a substantial amount of software developed in the mesh generator for snapping to geometry lines and surfaces.
- The ability to wrap geometry within the meshing process allows special treatment to be applied to the gap-closure faces, leading to a better quality volume mesh for flow simulation.
- The wrapping capability is designed to handle extremely complex industrial models containing many parts with complex topology. No assumptions are made regarding the quality or orientation of the input geometry, or the shape or nature of any holes present in the assembly. The wrapping is controlled simply by specifying the minimum size of hole which should be closed in the geometry.

7 ACKNOWLEDGEMENTS

The authors would like to thank Koenigsegg Automotive AB for their kind permission to use the Agera vehicle geometry, and Skoda for their kind permission to use the Fabia II vehicle geometry.

REFERENCES

[1] Y. K. Lee, C.K. Lim, H. Ghazialam, H. Vardham, E. Eklund, Surface Mesh Generation for Dirty Geometries by Shrink Wrapping using Cartesian Grid Approach, *Engineering with Computers*, August 2010, Vol 26, Issue 4, pp 377-390.

[2] M.W. Beall, J. Walsh, M. S. Shephard, Accessing CAD Geometry for Mesh Generation, *Proceedings of the 12th International Meshing Roundtable*, pp 33-42, Sept. 2003.

[3] L. Marechal, Advances in Octree-Based All-Hexahedral Mesh Generation: Handling Sharp Features, *Proceedings of the 18th International Meshing Roundtable*, pp 65-84, 2009.

[4] A.A. Demargne, R.O. Evans, P.J. Tiller, W.N. Dawes, Practical and Reliable Mesh Generation for Complex, Real-World Geometries, *Proceedings of the 52nd Aerospace Sciences Meeting*, National Harbour, Maryland, January 2014.

[5] Z. J. Wang, K. Srinivasan, Complex “Dirty” Geometry Handling With An Interior-to-boundary Grid Generation Method, AIAA 2001-2538, June 2001.

[6] Y. Jun, A Piecewise Hole Filling Algorithm In Reverse Engineering, *Computer Aided Design 37*, 263-270, 2005.

[7] W. Zhao, S. Gao, H. Lin, A Robust Hole-Filling Algorithm For Triangular Mesh, *International Journal of Computer Graphics*, Vol. 23, Issue 12, November 2007.

[8] J. Carr, R. Beatson, J. Cherrie, T. Mitchell, W. Fright, B. McCallum, Reconstruction and representation of 3D objects with radial basis functions, *Proceedings of SIGGRAPH*, Los Angeles, CA, 12–17 August 2001, pp. 67–76. ACM Press, New York, 2001.

[9] J. Branch, F. Prieto, P. Boulanger, A Hole-Filling Algorithm for Triangular Meshes using Local Radial Basis Function, *Proceedings of the 15th International Meshing Roundtable*, pp. 411-431. Springer, Heidelburg, 2007.

[10] R.T. Whitaker, A Level-Set Approach to 3D Reconstruction From Range Data, *Int. J. Computer Vision 29*, 203-231, 1998.

[11] H.K. Zhao, S. Osher, R. Fedkiw, Fast Surface Reconstruction Using the Level Set Method, *Proceedings of the IEEE Workshop on Variational and Level Set Methods in Computer Vision (VLSM 2001)*, pp. 194-201, 2001.

[12] F. Juretić, N. Putz, A Surface-Wrapping Algorithm with Hole Detection Based on the Heat Equation, *Proceedings of the 20th International Meshing Roundtable*, pp. 405-418, 2011.

[13] A. Kumar, A.M. Shih, Hybrid Approach for Repair of Geometry with Complex Topology, *Proceedings of the 20th International Meshing Roundtable*, pp. 387-403, 2011.

[14] E.W. Lorenson, E.H. Cline, Marching Cubes: A High Resolution 3D Surface Construction Algorithm, *ACM Computer Graphics 21(4)*, 1987.

[15] D. Martineau, J. Gould, J. Papper, Towards an Efficient Distributed Geometry for Parallel Mesh Generation, Research Note, *22nd International Meshing Roundtable*, 2013.

[16] H.G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, *Computers In Physics, Vol. 12*, No. 6, Nov/Dec 1998.

[17] L.P. Kobbelt, J. Vorsatz, U. Labsik, H. Seidel, A Shrink Wrapping Approach to Remeshing Polygonal Surfaces. *Computer Graphics Forum, Proceedings of Eurographics '99*, 18(3): 119–130, 1999.

[18] Flange case geometry. Retrieved on 15/01/2016 from https://github.com/OpenFOAM/OpenFOAM-3.0.x/tree/master/tutorials/resources/geometry

[19] K. Srinivasan, Z.J. Wang, W. Yuan, R. Sun, Vehicle Thermal Management Simulation using a Rapid Omni-Tree Based Adaptive Cartesian Mesh Generation Methodology, *Proceedings of ASME Heat Transfer/Fluids Engineering Summer Conference*, Charlotte, North Carolina, USA, July 11-15, 2004.

[20] D. Wang, O. Hassan, K. Morgan, N. Weatherill, Enhanced remeshing from STL files with applications to surface grid generation. *Commun. Numer. Meth. Engng.*, 23: 227–239, 2007.

[21] L. Kobbelt, S. Campagna and H. Seidel, A General Framework for Mesh Decimation, *Proceedings of Graphics Interface 1998 Conference*, June 18-20, 1998.