2011-01-0523

Automated Meshing for Aero-Thermal Analysis of Complex Automotive Geometries

W.N.Dawes
Cambridge University Engineering Department, CFD Laboratory, Cambridge UK

W.P.Kellar, N.C.Eccles & S.A.Harvey
Cambridge Flow Solutions, Compass House, Vision Park, Histon, Cambridge UK

ABSTRACT

Successful product development, especially in motorsport, increasingly depends not just on the ability to simulate aero-thermal behavior of complex geometrical configurations, but also the ability to automate these simulations within a workflow and perform as many simulations as possible within constrained time frames. The core of these aero-thermal simulations - and usually the main bottleneck - is generating the computational mesh.

This paper describes recent work aimed at developing a mesh generator which can reliably produce meshes for geometries of essentially arbitrary complexity in an automated manner and fast enough to keep up with the pace of an engineering development program. Our goal is to be able to script the mesh generation within an automated workflow - and forget it.

INTRODUCTION

Many approaches to mesh generation have been put forward - and we have tried most of them. The key requirements are the ability to deal with dirty geometry emerging from CAD, to produce acceptable mesh quality no matter how complex the geometry, to scale straightforwardly no matter how large the geometry - and to never fail.

The two main novelties of our approach are as follows. First we use a bottom-up octree derived from a space filling curve as a background mesh - with variable depth refinement to match variable geometry curvature. This background octree captures the geometry reliably even if the geometry is dirty. The bottom-up approach allows efficient dynamic load balancing of the mesh onto a multi-core compute cluster. Second, we construct a body-conformal mesh, with viscous layer cells, via shape insertion and mesh morphing based on continuous control of mesh quality via a set of mesh metrics (warpage, skew, smoothness) and optimization algorithms. Finally, hanging nodes are removed and the mesh exported in standard hybrid mesh formats. The mesh can also be post-processed into polyhedral form to reduce cell count. With our approach, complex geometries exported from CAD via STL/VRML can be reliably converted into meshes containing many millions of cells in a matter of tens of minutes on a modest CPU cluster.

The organization of the paper is as follows: first the basis of the methodology will be described and illustrated. Then, application to some geometries of automotive relevance will be presented. Finally, potential future work will be described.

METHODOLOGY

Our methodology has been described in detail in a series of papers - Dawes et al [2005-2010]. Our intention here is to summarize the key elements of our approach and attempt to demonstrate its advantages.

The starting point is the geometry represented in any tesselated format, STL or VRML, and preferably exported as a set of parts from a CAD solid modeler (to increase simulation productivity by allowing automated boundary associativity as described for example by Haimes et al [1998]). This geometry can be "dirty" in the sense that there can be small gaps in the STL or the STL may be locally folded or distorted - our approach simply meshes over the top of these imperfections. Geometrical features are resolved down to the requested smallest mesh scales, sub-scale geometrical features are automatically smoothed. The imported geometry is captured as a distance field and managed as a Level Set using a methodology developed from the work of Adalsteinsson et al [1995]. This implicit solid model is available to both guide the mesh generation itself and to support downstream geometry editing and morphing operations (which will not be discussed further in this paper but Baerentzen [2001] and Dawes [2005] are relevant references).

The first step in the mesh generation is to develop a bottom-up octree derived from a space filling curve (see Samet [1990] or Sagan [1994]) as a background mesh. Variable depth refinement is employed to match local mesh scale to variable geometry curvature. Figure 1 illustrates this: the zero Level Set - the geometry - is cut into the mesh using standard computer graphics constructs, see for example Glassner [1990+].This "cutting" is very efficient and scales very well even to arbitrarily sized geometries. This bottom-up approach is already known, see for example Tu et al [2006], but has not been commonly adopted in other meshing systems (see for example Aftosmis et al [1998] or Bussoletti et al [1985]). We selected it because it allows efficient dynamic load balancing of the mesh onto a multi-core PC cluster. Figure 2 shows a simple mesh and its associated Space Filling Curve, BFJKLMHIDE (which is a Morton Z-curve sequence derived by interleaving each cell's locational code) followed by division into two domains, "yellow" (BFJKL) and "green" (MHIDE) thus achieving good load balancing.

The second step in the mesh generation process is the construction of a body-conformal mesh, with viscous layer cells. This is based on a combination of "shape insertion" (developed from the ideas of Yerry et al [1983]) and mesh morphing based on continuous control of mesh quality measured via a set of mesh metrics (warpage, skew, smoothness) and managed by formal optimization algorithms. Figure 3 illustrates this twin approach. After this, as a post-processing filter, hanging nodes are removed and the mesh exported in standard hybrid mesh formats.

Several particular aspects of the mesh generation deserve special mention and are illustrated in Figure 4. First is the need for automatic feature detection to manage and resolve convex corners; the illustration shows the surface mesh on a ribbed structure with and without this feature turned on. The other two illustrations show layer meshing (on a turbine blade) and polyhedralised export which usefully reduces cell count.

With our approach, complex geometries can be reliably converted into meshes containing many millions of cells in a matter of tens of minutes on a modest CPU cluster. Although the meshing can be driven by a GUI we have specifically designed the system to be fully automatable and scriptable to be embedded within a workflow.

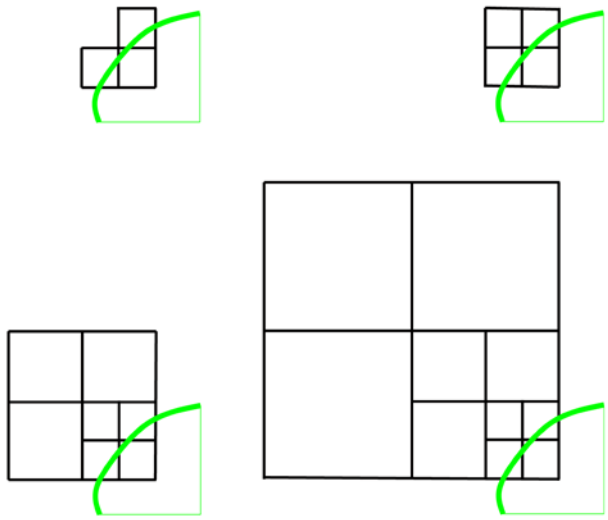

Figure 1: Bottom-up Octree; the Level Set zero, "green" cuts the emerging octree using standard computer graphics constructs.

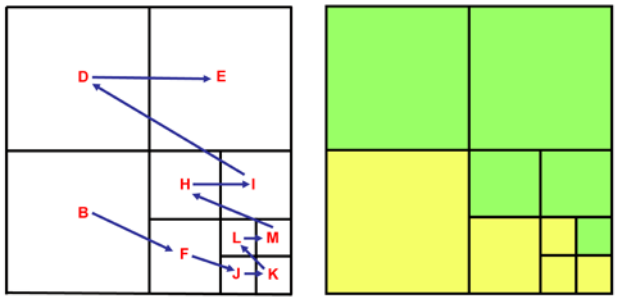

Figure 2: Parallel load balancing via a Space Filling Curve: BFJKLMHIDE is a Morton Z-curve sequence derived by interleaving each cell's locational code and division here into two domains, "yellow" (BFJKL) and "green" (MHIDE) achieves good load balancing.

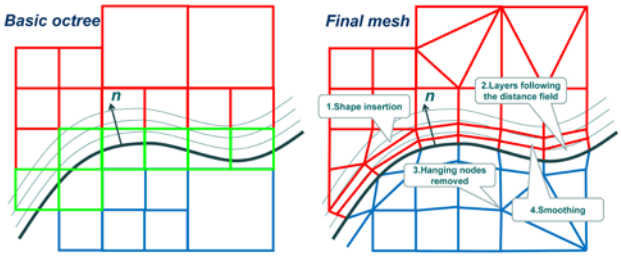

Figure 3: Body-conformal mesh generation via shape insertion and mesh morphing.

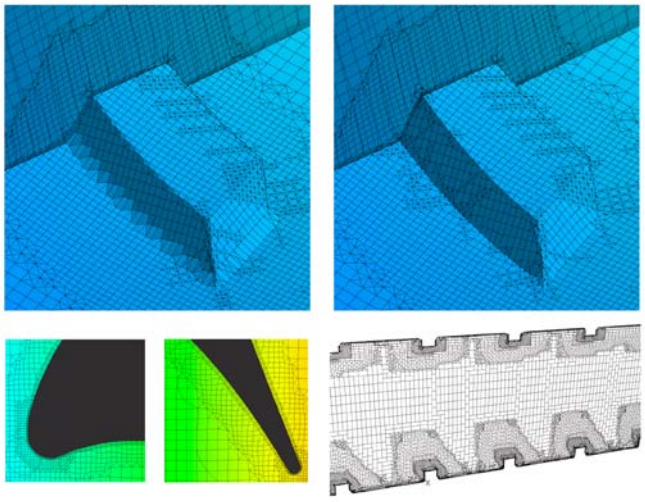

Figure 4: Particular aspects of the mesh generation: top row, convex corners without (left) and with (right) automatic feature detection; bottom left, layer meshing; bottom right, polyhedralised export

RESULTS

To illustrate applications of our approach we show three representative sets of meshes. These are all generated from generic geometries available from public domain websites.

The first, shown in Figure 5, is a generic brake assembly showing a conjugate mesh for CHT (conjugate heat transfer) analysis; the air-side is colored red and the metal-side, green. The mesh contains around 10M cells and was generated on about 15 minutes from STL import to hybrid mesh export.

The second example, Figure 6 represents a generic motor bike: top left is the imported STL geometry file; the other pictures illustrate the surface mesh and associated slices through the volume mesh. This mesh contains around 30M cells and was generated on about 20 minutes from STL import to hybrid mesh export.

The final example, shown in Figure 7, consists of a generic car and under-hood. The mesh contains around 100M cells and was generated on about 45 minutes from STL import to hybrid mesh export. One shot shows the layer mesh on the external air-side of the hood; one shows the well resolved, very complex under-hood domain.

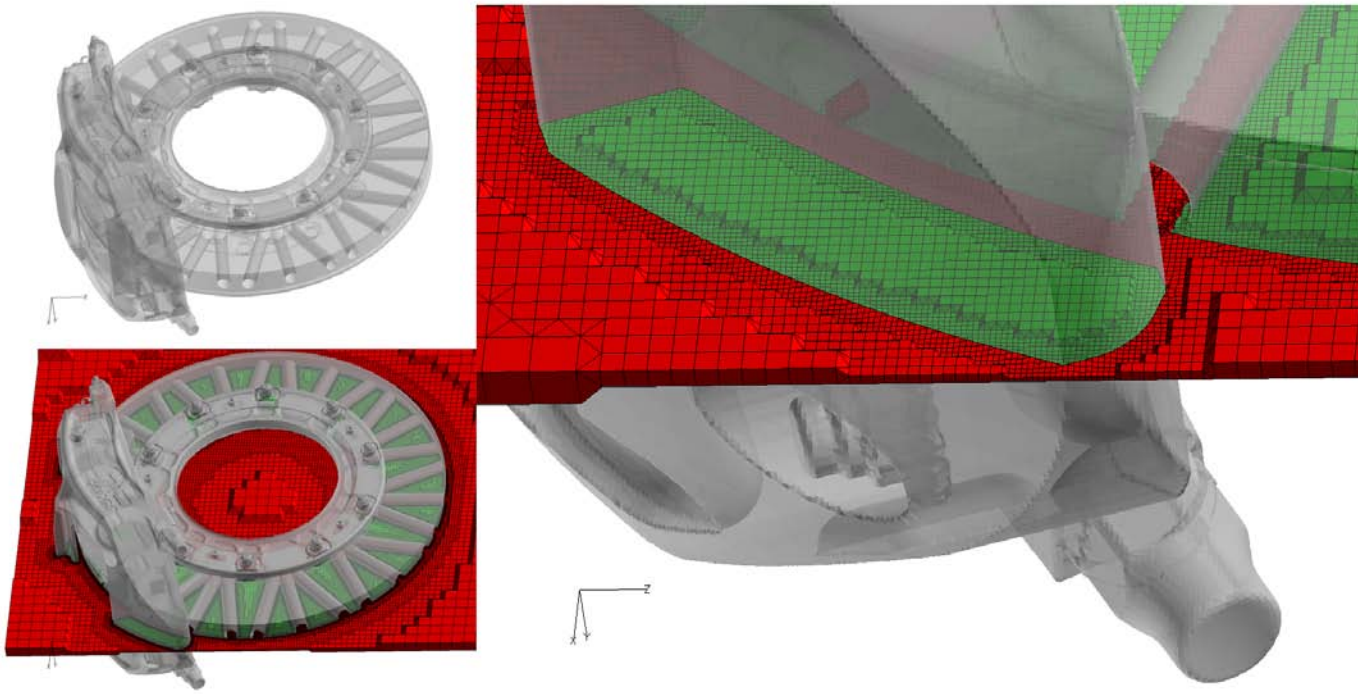

Figure 5: A generic brake assembly showing a conjugate mesh for CHT analysis; air=red, metal=green.

SUMMARY/CONCLUSIONS

This paper has described recent work aimed at developing a mesh generator which can reliably produce meshes for geometries of essentially arbitrary complexity in an automated manner and fast enough to keep up with the pace of an engineering development program.

We have tried to illustrate the potential of our approach on a range of representative geometries.

The mesh generation can be fully automated and scripted within an integrated workflow.

In future work we will exploit the geometry editing abilities of our underpinning solid model to support design optimisation.

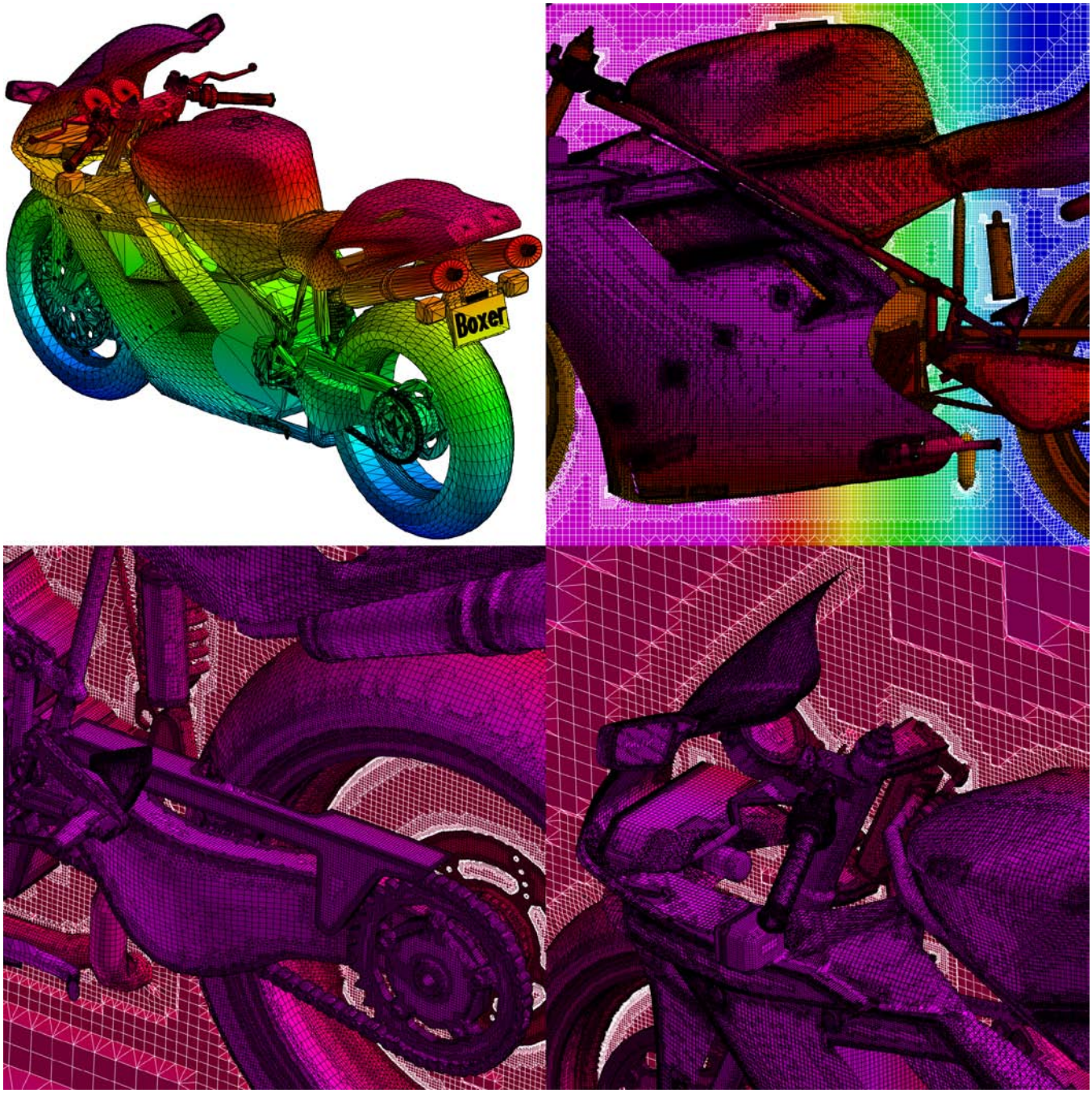

Figure 6: A generic motor bike: top left is the imported STL; the other pictures illustrate the surface mesh and associated slices through the volume mesh.

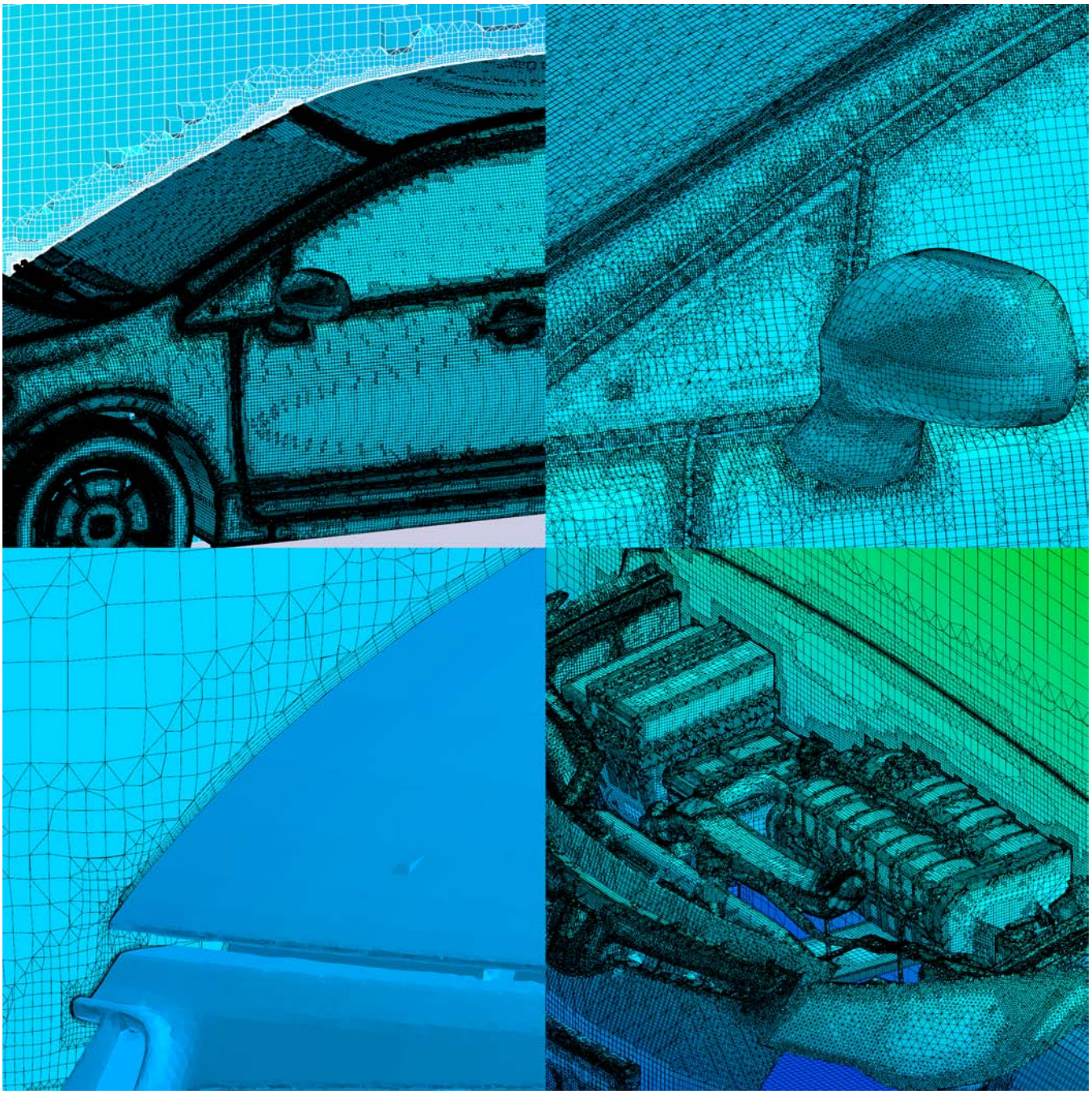

Figure 7: A generic car and under-hood

REFERENCES

1. Dawes WN “Building Blocks Towards VR-Based Flow Sculpting” AIAA-2005-1156

2. Dawes WN, Kellar WP, Harvey SA “Towards a fully integrated parallel geometry kernel, mesh generator, flow solver & post-processor” AIAA-2006-45023

3. Dawes WN, Kellar WP, Harvey SA “Viscous Layer Meshes from Level Sets on Cartesian Meshes” AIAA-2007-0555

4. Dawes WN, Kellar WP, Harvey SA “Towards topology-free optimisation: an application to turbine internal cooling geometries” AIAA-2008-925

5. Dawes WN, Kellar WP, Harvey SA ”A practical demonstration of scalable parallel mesh generation” AIAA-2009-0981

6. Dawes WN, Kellar WP, Harvey SA “Generation of conjugate meshes for complex geometries for coupled multi-physics simulations” AIAA-2010-062

7. Haimes R & Follen GL “Computational Analysis Programming Interface” Proc. 6th Int. Conf. On Numerical Grid Generation in Computational Field Simulations, Eds. Cross, Eiseman, Hauser, Soni & Thompson, July 1998.

8. Adalsteinsson D & Sethian JA “A level set approach to a unified model for etching, deposition & lithography II: three dimensional simulations” J.Comput.Phys, 122, 348-366, 1995

9. Baerentzen A “Volume sculpting: intuitive, interactive 3D shape modelling” IMM, May 2001

10. Samet H “The design and analysis of spatial data structures” Addison-Wesley Series on Computer Science and Information Processing, Addison-Wesley 1990

11. Sagan H “Space Filling Curves” Springer-Verlag 1994

12. Glassner AS “Graphics Gems” Academic Press, 1990+

13. Tu T, Yi H, Ramirez-Guzman L, Bielak J, Ghattas O, Ma K-L & O’Hallaron DR “From mesh generation to scientific visualization: an end-to-end approach to parallel super-computing” SC2006, Tampa FL, 2006

14. Aftosmis MJ, Berger MJ & Melton JE, “Robust and efficient Cartesian mesh generation for component-based geometry” AIAA J., 36, 6, 952-, 1998

15. Bussoletti JE, Johnson FT, Bieterman MB, Hilmes CL, Melvin RG, Young DP & Drela M “TRANAIR: solution-adaptive CFD modelling for complex 3D configurations” AIAA Paper 1995-xxxx, 1985

16. Yerry MA & Sheppard MS “Automatic three-dimensional mesh generation by the modified octree technique” Int.J.for Num.Methods in Engineering, vol.20, 1965-1990,1983

CONTACT INFORMATION

will.kellar@cambridgeflowsolutions.com

ACKNOWLEDGMENTS

We are very pleased to acknowledge the technical & financial contributions of our Development Partners.