

**AIAA-2009-0372**

Invited Paper for a Special Session on Level Sets

## **Using Level Sets as the basis for a scalable, parallel geometry engine and mesh generation system**

WN Dawes,\*

CFD Laboratory, Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, UK,  
and

WP Kellar & SA Harvey,

Cambridge Flow Solutions Ltd, Compass House, Vision Park, Cambridge, CB4 9AD, UK

**The background to this review paper is research we have performed over recent years aimed at developing a simulation system capable of handling large scale, real world applications implemented in an end-to-end parallel, scalable manner. The particular focus of this paper is the use of a Level Set solid modeling geometry kernel within this parallel framework to enable automated design optimization without topological restrictions and on geometries of arbitrary complexity. Also described is another interesting application of Level Sets: their use in guiding the export of a body-conformal mesh from our basic cut-Cartesian background octree – mesh - this permits third party flow solvers to be deployed. As a practical demonstrations meshes of guaranteed quality are generated and flow-solved for a B747 in full landing configuration and an automated optimization is performed on a cooled turbine tip geometry.**

### **I. Introduction**

There are a number of current trends in simulation. The first is for ever larger model sizes. For example, in an aero-engine an individual turbine blade-passage can be meshed with say 250k nodes; a whole annulus of these blades needs 15M cells say; the stage 30M; adding in the cooling system would take the total past 100M; adding conjugate heat transfer leads to 150M; and so on. The second trend is to link more disciplines together. For example as soon as an aero-thermal simulation can be performed for a cooled turbine blade the urge becomes irresistible to couple in FE analysis on the metal side to predict hot-cold running geometries. The third trend is to gather all this together in “process chains” for routine industrial exploitation – leading towards automated design optimization. Apart from size, the common element linking all this together is the geometry.

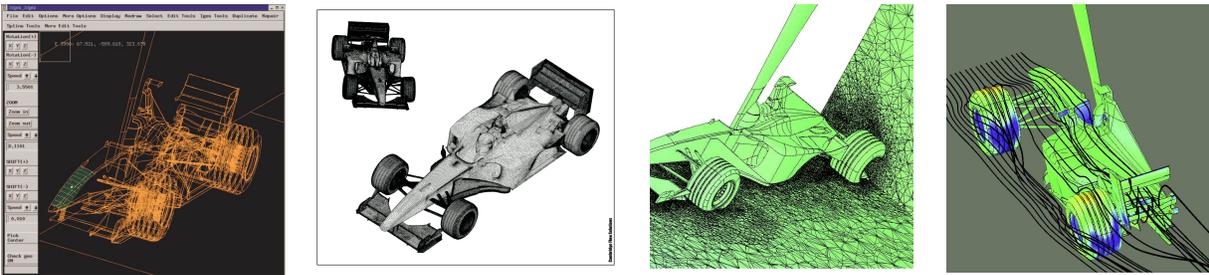
These simulations are characterized both by real, complex geometries and also by scale – not just physical scales (which may be widely disparate) but also by scale of mesh resolution needed to support realistic modeling like LES. It is clear that in the future, simulation must employ end-to-end parallelism – from the geometry kernel through the mesh generation and onto the solver/post-processor. This parallelism must be scalable and built on data structures and software architecture paradigms capable of dynamic load balancing. This paper will focus on the use of Level Sets as a suitable solid modeling kernel to support this geometry-centric vision.

To establish some context, Figure 1 shows some work performed nearly ten years ago simulating the flow around an entire F1 racing car (summary in Kellar [2003]). The unstructured tetrahedral mesh was a very modest 7.5Mcells

---

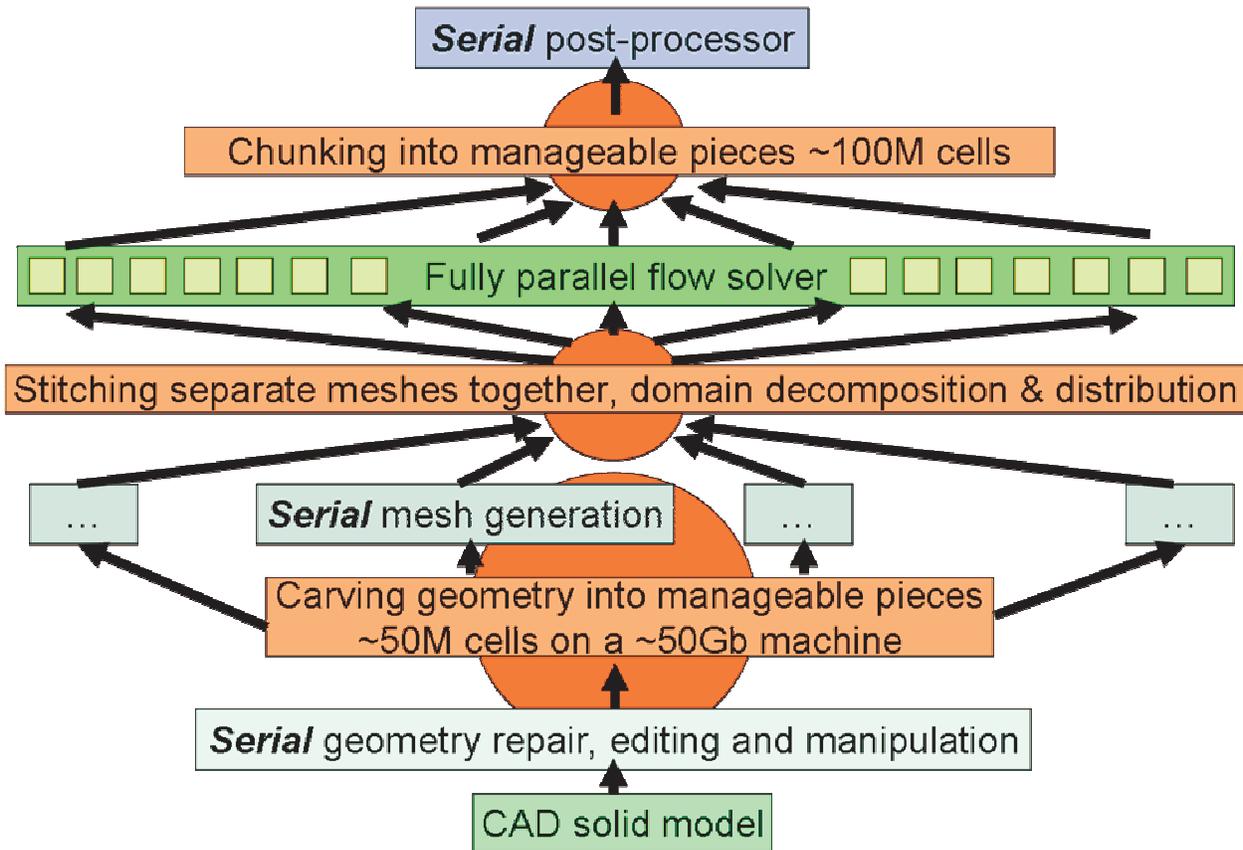
\* wnd@eng.cam.ac.uk

and used a simple CAD-to-mesh-to-solver simulation system (Dawes et al [2000]). The CAD model exported from UniGraphics in IGES form contained many hundreds of entities and the repair of these into a watertight solid followed by mesh generation took many weeks. By contrast the flow solution itself took a matter of days.

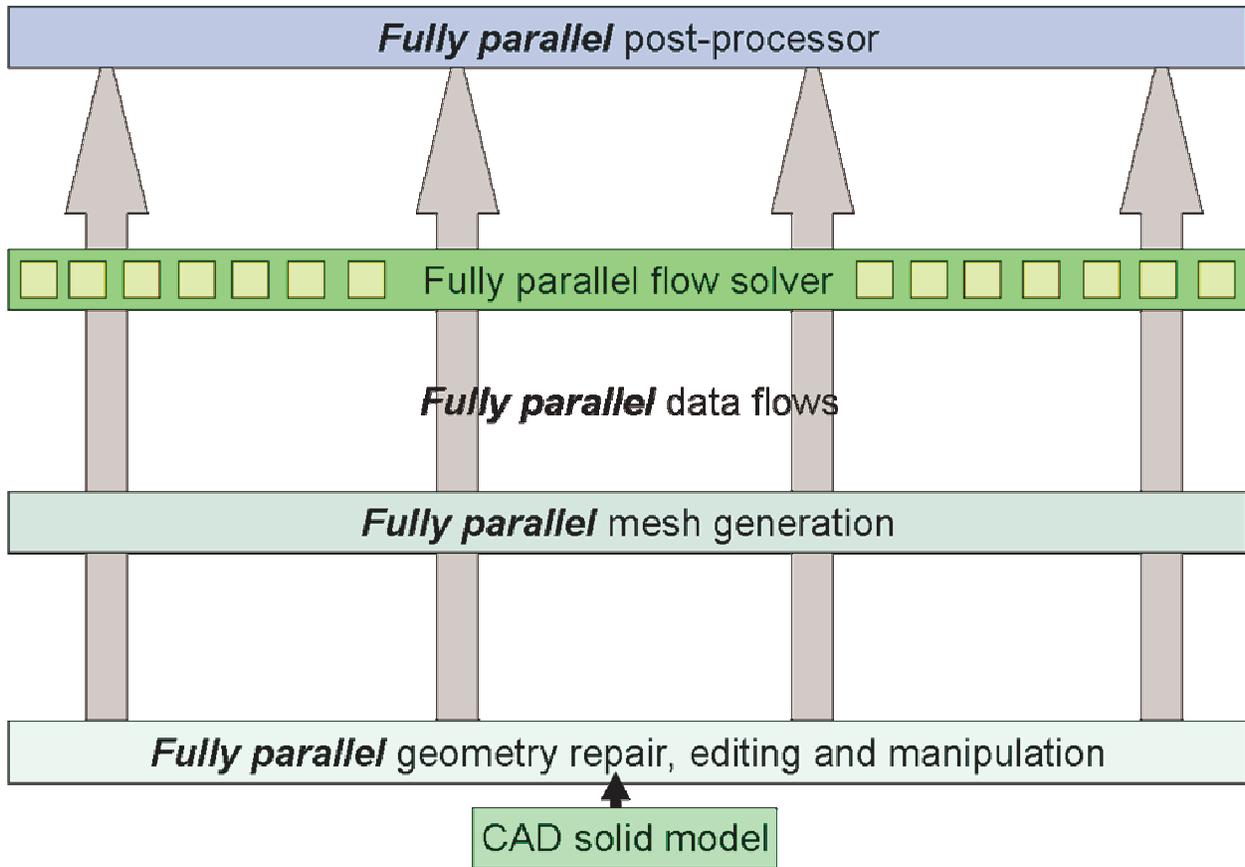


**Figure 1: Simulation of the flow around an entire F1 racing car – Kellar [2003]; from left to right: CAD export as IGES, surface mesh, volume mesh, flow solution.**

Nowadays simulations like this are routinely performed with ~500M cells to get useful predictive accuracy and we have successfully used a series of COTS products (for example: UniGraphics, ANSA, ICEM, FLUENT and FieldView) linked together as shown in Figure 2. As the Figure makes clear, and even though the final quality of the predicted fluid dynamics is satisfactory, this process chain is far from ideal. There are a number of critical, serial bottlenecks impeding data flow; we have had to produce “glueware” to perform critical tasks like geometry “carving” and mesh “stitching” & “chunking”. The flow solver performs well in parallel and occupies only a day or so of overall wall-clock time; everything else is serial & slow and needs perhaps two weeks to accommodate major topological geometry changes. And this is the crunch – *changing geometry is the key activity of a design engineer.*



**Figure 2: A typical process chain for flow simulation of model with size ~500M cells**



**Figure 3: An end-to-end fully parallel process chain.**

A far preferable process chain is sketched in Figure 3: an end-to-end fully parallel process chain.

Our research over recent years has been devoted to this vision. We have explored one possible way forward and reported our experiences in a sequence of papers. Our chosen methodology was deliberately different from the current, orthodox CFD process chain – and is determinedly geometry-centric. The essence of this new approach was the integration of a geometry kernel based on a Level Set approach with an octree-based cut-Cartesian mesh generator, RANS flow solver and post-processing all within a single piece of software.

The basic building-block work was reported by Dawes [2005]; the potential to parallelize the entire system was reported in Dawes [2006] and illustrated with prototype versions of our software, *BOXER*. Replacing the cut-cells with body-conformal meshes was described in Dawes et al [2007]; in this work the underpinning Level Set was used with optimization algorithms based on mesh quality metrics to enable the export of meshes with guaranteed quality to drive third party solvers like FLUENT®. Figure 4 illustrates the *BOXER* paradigm. This year, Dawes et al [2009] reported two additional novelties: first, a generalization of the earlier work to permit variable depth octree refinement to enable variable surface refinement; second, a radical rework of the earlier parallel mesh generation from a simple top-down octree to a bottom-up octree based on Morton coding and Space Filling Curves. Also reported are the associated extensions to permit smooth surface reconstruction from underlying Level Set to allow body-conformal mesh export – with no hanging nodes. All of this is implemented in a scalable manner within a robust C++ architecture.

The remainder of this review paper will focus on applications we make of Level Sets as a geometry-centric kernel.

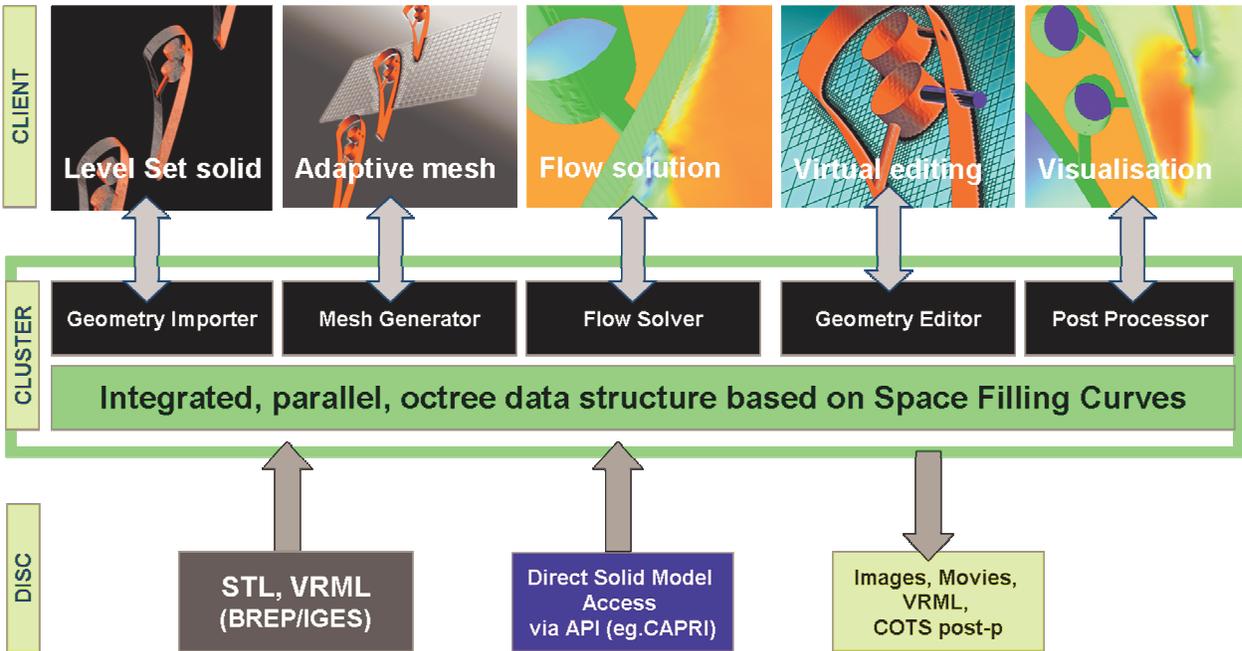


Figure 4: The *BOXER* paradigm.

## II. What are Level Sets and how can they help?

First it is helpful to summarize the three basic types of solid model: Constructive Solid Geometry, Boundary Representation and Spatial Occupancy.

*Constructive Solid Geometry* (CSG) is based on the definition & manipulation of simple analytical bodies (cube, sphere, ellipse,...) which can be scaled, translated and rotated. These body primitives are then combined via Boolean logic together with inside/outside conventions to enable quite complex solid models to be produced. The advantages are simplicity and compactness of storage; the disadvantages are difficulty in representing the more free-form sorts of shapes encountered in aerodynamics.

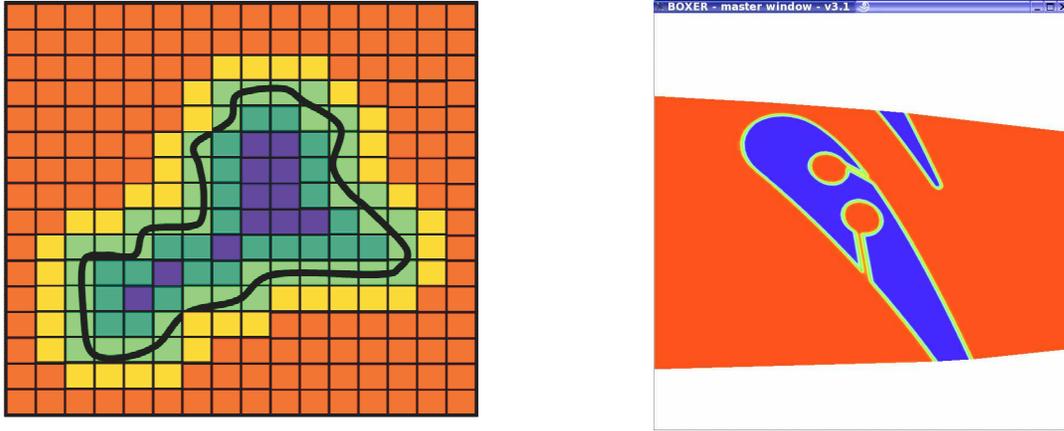
*Boundary Representation* (BREP) solid modelling is based on combining patches, edges & topology bindings into a “watertight” solid. Typically NURBS patches are used and complex, multiply curved surfaces can be stored very economically (the geometry needs only to be re-constructed for viewing or manipulation). The big disadvantages are in producing – and maintaining under manipulation – the watertight bindings and supporting topology (genus) change.

*Spatial Occupancy* solid models consist of Cartesian hexahedral cells (perhaps stored in an octree data structure for efficiency) with cells occupied, or vacant or cut (with perhaps some local surface shape data stored). The big advantages are simplicity and generality – topology change is trivially supported – but the disadvantage is the elevated storage overhead compared to BREP & CSG.

Commercial CAD uses mostly BREP with CSG constructs where possible coupled with octree-like data structures if needed, to make searching operations efficient. The kernel vendors license access (for example UniGraphics sits on Parasolid) and direct read of the kernel can interrogate the solid model to return data like curvature, inside/outside etc... Of particular interest is access to the tessellation of the surface of the solid model which is used to render it. We use CAPRI (Haimes et al [1998]) which combines kernel queries with visualisation & is implemented for UG (Parasolid), Pro/E (Pro/Toolkit) & CATIA. However, the tessellations themselves are not directly suitable as a flow mesh and thus must be used as a driver for a more flow oriented mesh generator. In a more general sense, tessellated

surfaces – especially those held in STL or VRML formats – have become the de facto standard geometry representation.

For our work on *BOXER* our choice of solid modeling required a method which could be parallelized – with no scalability issues – and which could support topological change for geometries of arbitrary complexity. Accordingly we were drawn to the Spatial Occupancy approach. When a solid model is represented using Spatial Occupancy each cell – or *voxel* – can have associated with it the signed distance to the nearest point on the body (or bodies). This is known as a “distance field”. This is illustrated in Figure 5 which shows (on the left) a body represented simply by a distance field extracted using the “city block” method (Perng et al [2001]); much smoother, “exact” body representations can be extracted using the Level Set approach (right hand plot). Boundaries are represented as the zero isosurface of the distance field.



**Figure 5: Bodies represented by distance fields: on the left derived from a city block methods; on the right from the Level Set approach; both fields are “narrow band”.**

The Level Set method itself (see for example Sethian [1987], Osher et al [1998] and Adalsteinsson et al [1995]) has as its key idea the representation of a propagating interface as the zero value of a signed distance function

$$\phi(x, t > 0) = \pm d$$

It is easy to show that  $\phi$  has an associated evolution equation

$$\phi_t + F |\nabla\phi| = 0$$

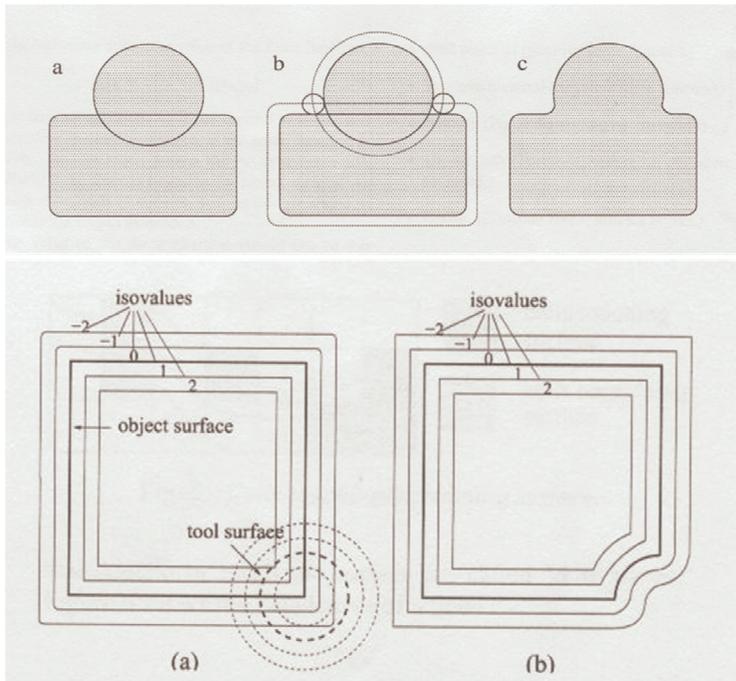
where  $F$  is the speed function in the normal direction (a function of curvature, deposition/etch rate etc.). At any time the distance function can be re-initialised by solving  $|\nabla\phi| = 1$  using simple iterative techniques that mimic time marching evolution.

It should be immediately clear from the above that the transformation of an *explicit* geometry representation (CSG, BREP) into an *implicit* one based on a simple, scalar field variable, the distance field, offers tremendous advantages from the point of view of enabling a parallelizable solid modeling geometry kernel to support not only mesh generation but also design and automated design optimization. There are also some benefits in guiding generation of viscous layer meshes for complex geometries which will be described later. However, next, the editing of geometry itself – the key activity for a designer – will be described. This has been called “sculpting”.

A very interesting recent development in the field of 3D computer graphics has been real-time sculpting in virtual reality – first proposed by Galyean et al [1991]. The basic principles are very simple: space is divided up into a 3D mesh of cells – “voxels”; a work-piece and a tool are defined and manipulated as basic spatial occupancy solid models; interactions between the tool and the work-piece result in modified geometry; these interactions can be via

adding or removing of material (basic CSG type operations) or spraying of new material onto an existing solid or anything else which can be devised. The whole procedure is similar in concept to sculpting with clay – and quite complex and sophisticated results can be produced (see for example Bremer et al [2001]). Of course, the surface of the emerging solid body must be managed carefully. There are two main approaches: one is based on the classic graphics rendering algorithm called “marching cubes” which simply aims to remove aliasing (see Perng et al [2001]; a more sophisticated approach is based on Level Set techniques (for example Baerentzen [2001]) – this is capable of faithfully representing multiply curved surfaces.

*Sculpting*, simply means editing the distance field as illustrated in Figure 6. Topological changes are trivially supported since the underlying solid model is based on Spatial Occupancy rather than conventional BREP. Typical solid modelling operations from CSG take the form of Boolean sums which are replicated in distance fields via simple inexpensive and *parallelizable* voxel-wise logic.



CSG (Computational Solid Geometry)  
Boolean sums

$$C = A \cup B$$

Replicated in distance fields via simple voxel-wise logic

$$d_{\text{new}} = \min(d_{\text{object}}, d_{\text{tool}})$$

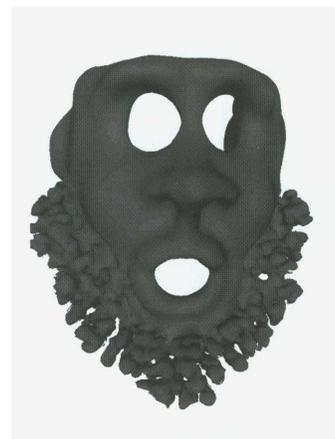
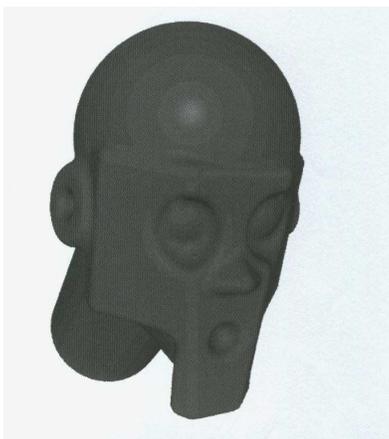


Figure 6: Sculpting using Boolean summation via CSG and translated into voxel-wise distance field operations (top); basic head (bottom left) sculpted simply via CSG Boolean operations enacted via a spatial occupancy solid model of tool & work-piece; the head on the right results from much more complex tool/work-piece interactions (from Baerentzen [2001]).

In the remainder of this review paper we will describe the use we have made of a Level Set solid modeling kernel within *BOXER* to first support fully parallel mesh generation and second to the use of Level Sets to guide viscous layer meshing. Then the opportunity presented by geometry sculpting to design will be described in the context of cooled gas turbine blades. Finally, an automated, topological design optimization aimed at reducing turbine tip heat load will be outlined.

### III. Parallel, bottom-up octree mesh generation

In earlier work, reported in Dawes et al [2006] mesh generation based on a very simple *top-down* octree was described. This starts with a single master cell which then divides in response to the geometry it contains until the final mesh is produced. This is easy to code and was parallelized using simple coordinate-axis based load balancing; promising early results were obtained & reported. However, as this work developed in application to arbitrary geometries it proved difficult to ensure satisfactory load-balancing – which must be done on-the-fly - as the mesh is generated. A better way is to invert the process and generate the mesh from the *bottom-up* – from the finest cells up the tree to the coarser ones. This sort of approach is less common and typically is based on Space Filling Curves and Morton coding (see for example Tu et al [2007]). However, it is much easier to dynamically load balance this approach and hence achieve parallel scalability.

The approach we adopted is described in full in Dawes et al [2009]; the ability to distribute a parallel representation of the geometry across the PC cluster was very useful and efficient in supporting this.

### IV. Generation of viscous layer meshes

The generation of viscous layer meshing for complex geometries remains a huge challenge. We have found the Level Set distance field very useful as a guide to near-wall mesh construction (see Dawes et al [2007]).

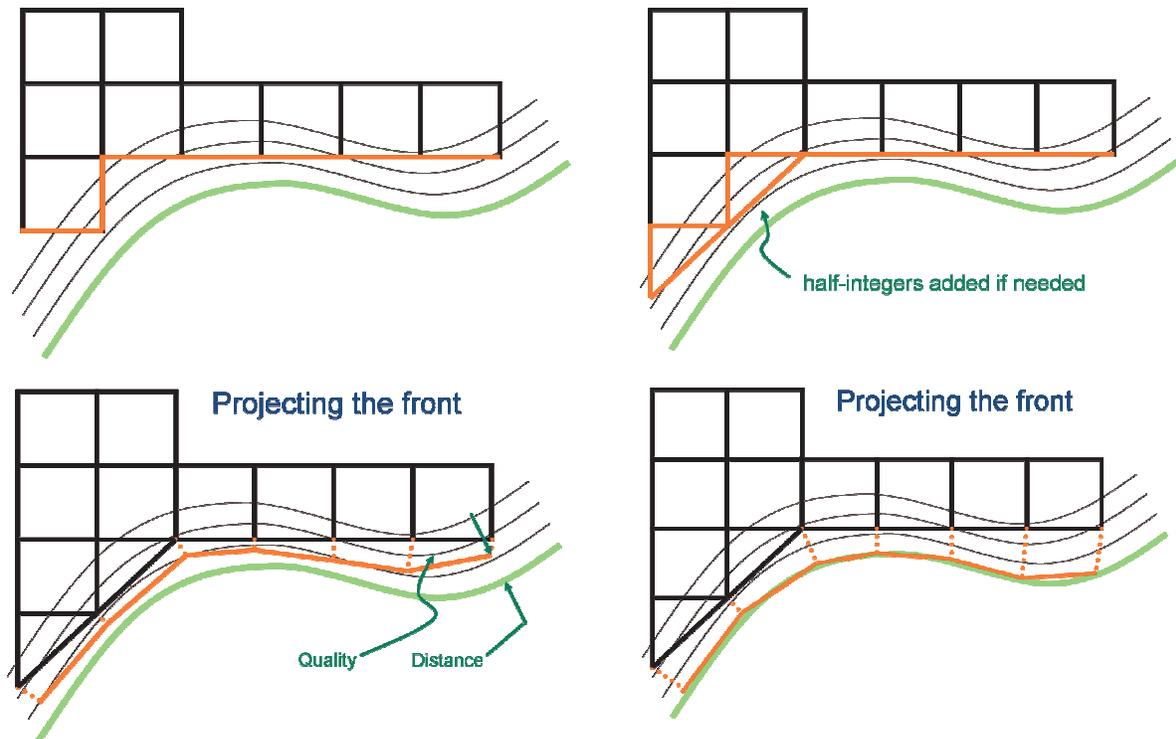
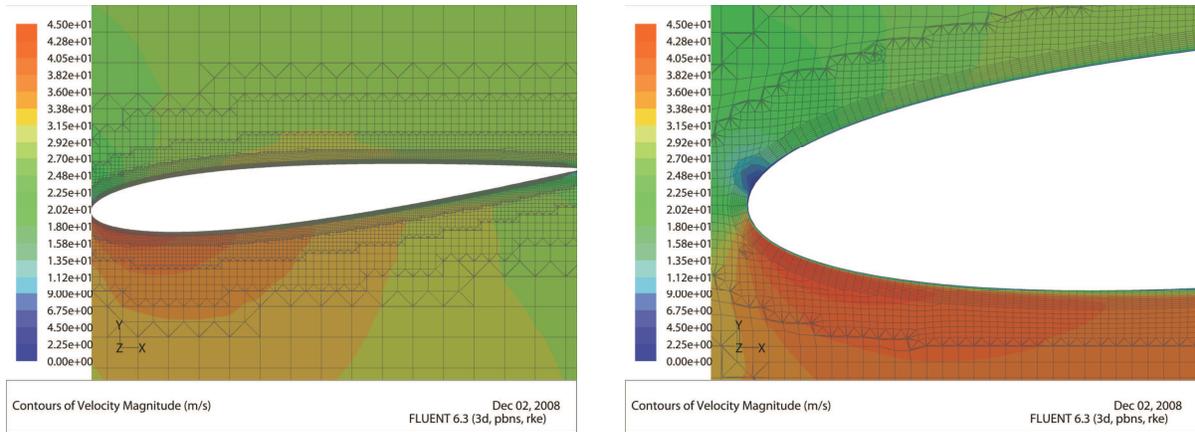
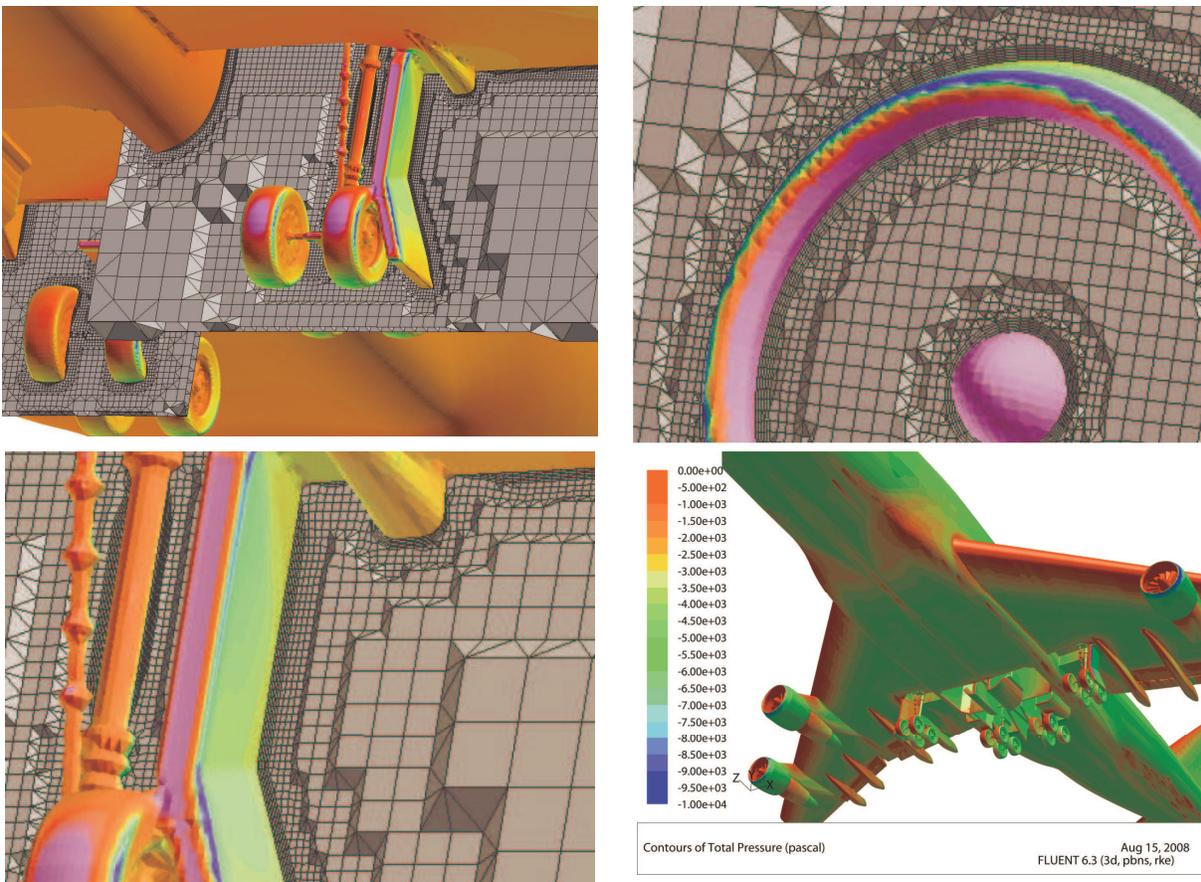


Figure 7: Generation of near-wall layer meshes guided by the distance field.

Figure 7 sketches the generation of a body-conformal mesh with near-wall layers. Removing the cut cells exposes a front of quadrilateral faces. These are then projected towards the body guided by the gradient of the distance field – which corresponds to the local body normal. (The Laplacian of the distance field corresponds to the local body curvature and can be used to guide local mesh refinement to permit feature recognition).



**Figure 8: Classic airfoil boundary layer mesh exported from *BOXER* to FLUENT® (Kellar [2008]).**



**Figure 9: A B747 in full landing configuration; various mesh views showing both the overall mesh generated by *BOXER* with viscous layers together with a flow solution obtained using FLUENT® (Kellar [2008]).**

An optimizer is used to minimize the distance of the front to the body whilst managing the quality of the mesh via various metrics (skew, warpage, etc...) so that the mesh is always of guaranteed quality. This is a key enabling step for any sort of automated design optimization.

This means that in practice clean, layer meshes can easily be constructed on relatively smooth areas of geometry (where there are likely to be boundary layers which can be effectively resolved on layer meshes) whereas near corners the mesh returns to isotropic (which is also what fluid dynamical resolution requires).

Current capability is illustrated in Figure 8 via the classic airfoil boundary layer mesh (Kellar [2008]) and in Figure 9 which shows a (rather under-resolved) flow solution of the flow around a B747 in full landing configuration – with all slats, flaps & wheels deployed. Figure 9 shows various views of the body-conformal mesh generated by *BOXER* from imported STL then exported into FLUENT® and solved at approximately approach speed and angle of attack. The mesh contains around 18M cells and each wetted surface has up to seven viscous layers. Probably a factor of ten more mesh resolution would be needed for a meaningful fluid dynamic simulation but the best test of – and proof of – mesh quality is a flow solution and it is in that spirit that this is presented here.

## V. Parallelizable geometry editing for topological optimization

An important application for us of the Level Set geometry kernel is enabling automatable geometry editing with full topological freedom. To illustrate this we will describe an example based on the important practical problem of turbine cooling. This represents a real, engineering application of the “sculpting” described in an earlier section.

Blades in the high pressure sections of high-performance aero-engines and land-based gas turbines are very commonly cooled using air from compressor exit delivered via an internal air system to within the blade – in complex serpentine cooling passages – and to the blade surface via film cooling holes. Much research, development and testing goes into the design of this critical technology. Inadequate cooling is responsible for a significant fraction of in-service problems and rectifying poorly performing design can be very expensive.

The essential difficulty of applying modern analysis tools like CFD to the design process in the complexity of the geometry combined with the vulnerability of the CFD Process to rapid and a priori unknown changes in geometry. Even with obvious short cuts like parameterising the CAD model & templating the CFD process it takes an unacceptable time to turn around simulations of new geometries – sometimes over a week per new geometry. This is exacerbated by the need to make frequent *topological* changes to the cooling concept – a blade always looks like a blade but the internal cooling system could have one or three or five passages with or without interconnection with an a priori unknown number of film cooling passages in a priori unknown locations. This means that in practice there is little scope within realistic design timescales to try innovative design.

The development of *BOXER* was started with exactly this sort of problem area in mind with the ultimate ambition of providing a rapid prototyping tool with sufficient speed, generality and accuracy to allow such designs to be optimized.

Figure 10 illustrates the basic idea. A cooled blade represented by the green isosurface of the zero distance from its associated Level Set is edited by a virtual tool – represented also by a distance field and rendered in blue. The virtual tool is moved with 6 degrees of freedom, either by the mouse or via scripting, and then edits a new film cooling hole into the blade. The underlying solid model is rebuilt and the new geometry re-meshed – all in a matter of seconds. Figure 11 shows “before” and “after” cut planes through the distance field (which is really a “narrow-band” field since it is only computed and stored over  $\pm 5\Delta$ , where  $\Delta$  is the smallest mesh scale). This simple but very powerful procedure can be extended with more generalized tools – even tools of the user’s choice imported via STL files – and other processes like Free Form Deformation.

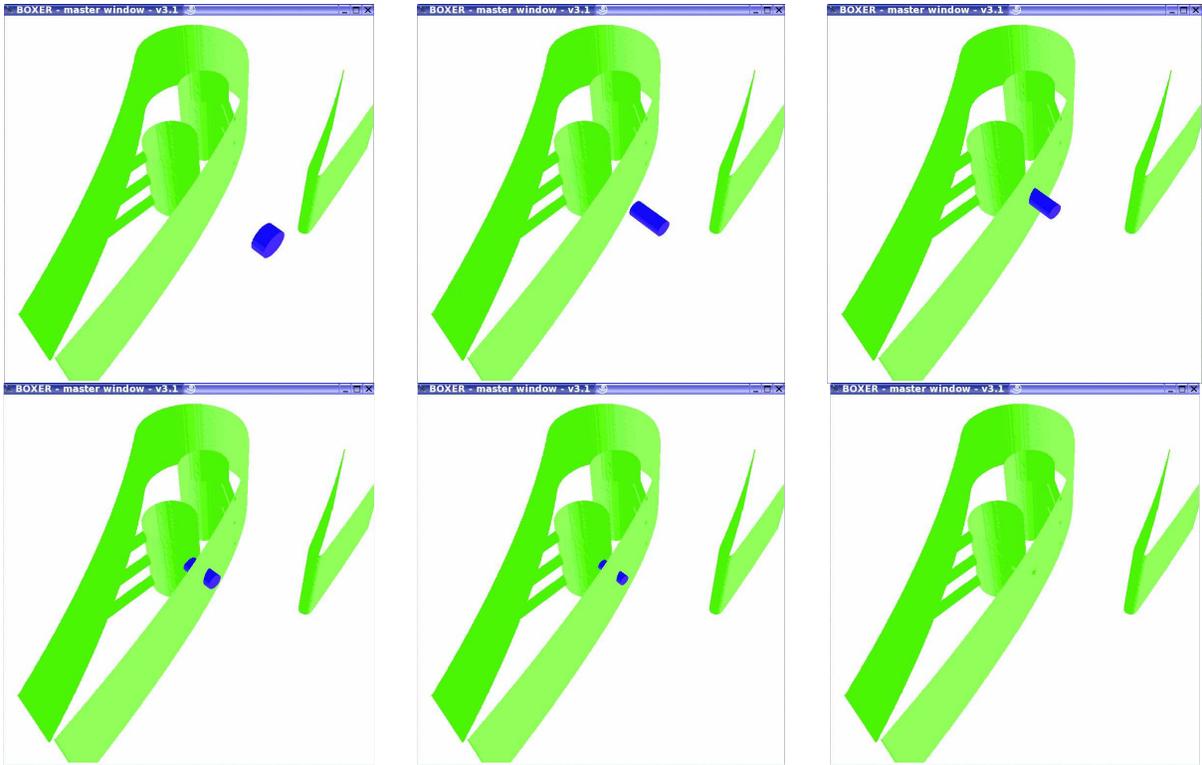


Figure 10: Sculpting a new film cooling passage through the distance field of a cooled turbine blade (the zero isosurface, the blade, is rendered green) with a cutting tool (blue).

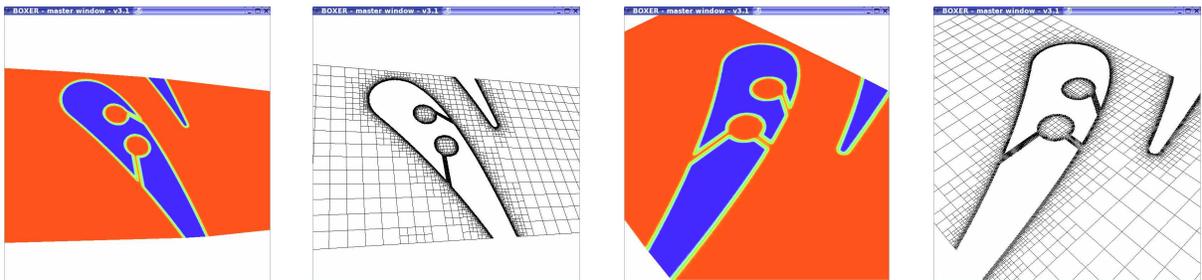


Figure 11: Cut planes through the blade distance field before (left pair) and after (right pair) sculpting.

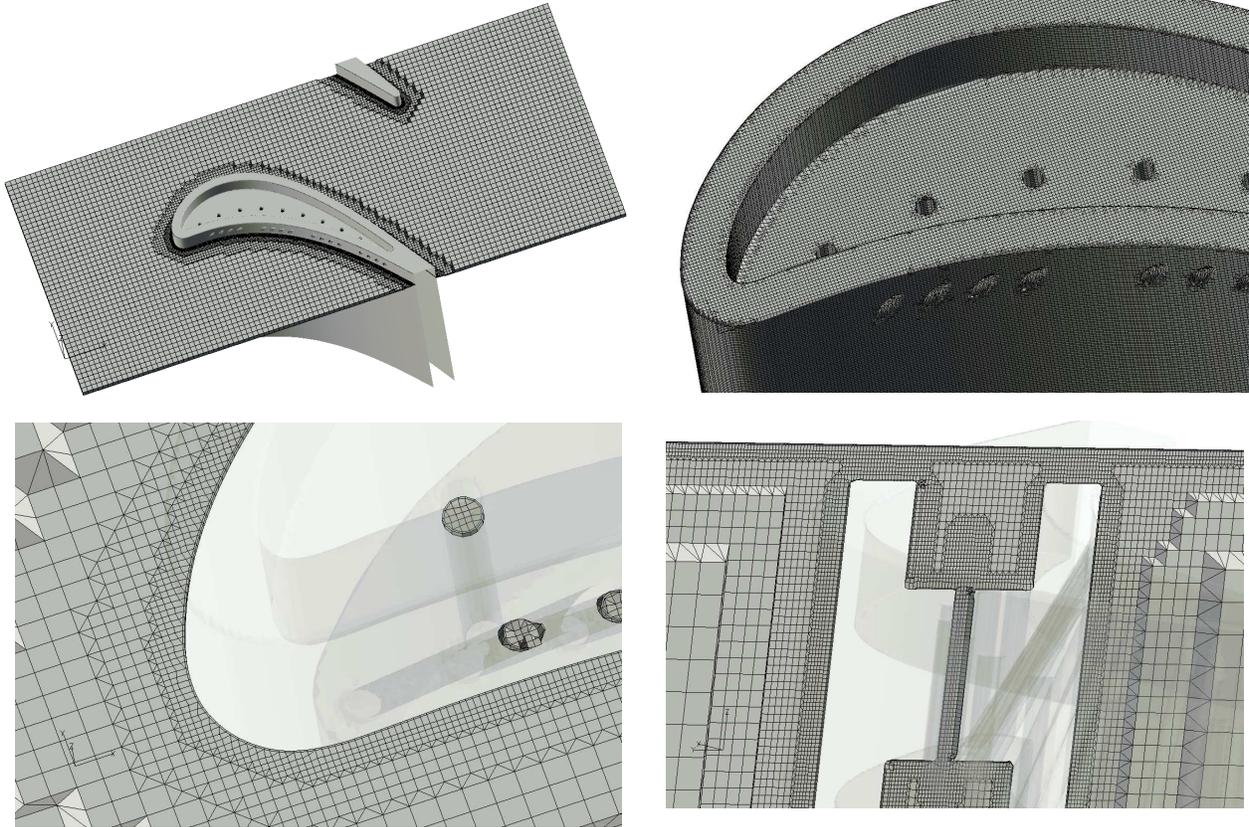
## VI. Automated optimisation

Finally, we will conclude by illustrating the application of all these ideas described above to the automated optimization of the tip heat load of a cooled turbine blade. (A full account is given in Dawes [2009] - a simple summary is presented here).

The candidate geometry is a simplified cooled, transonic turbine blade with a trenched tip and designated TG1; this has been designed, tested and analysed as part of an EU FP6 project on turbine cooling – AITEB-2 (see Janke et al [2005]). A mesh containing about 7.8M cells was generated with *BOXER* direct from the STL representation of the geometry exported from UniGraphics. This mesh was then exported from *BOXER* – after the addition of a viscous layer – in body-conformal, hybrid form and in FLUENT format. Figure 12 shows general and detailed views of the

mesh; in the actual optimization simulation the mesh was also adapted to follow the blade wakes. This mesh generation (including editing the cooling holes into the geometry) was fully scripted and took about 18 minutes to accomplish. The exported mesh is generated to conform to the published FLUENT mesh quality metrics. In particular the blade surface  $Y^+$  values were in the range 10-15 over the whole blade (and obviously lower in the trench itself).

For flow simulations FLUENT 6.3 was used – run on a cluster of four PC’s. The flow is subsonic so the pressure based solver was selected, with second order flux terms and the Spalart-Allamaras turbulence model. Fully converged simulations took around 36 wall-clock hours. The mid-span flowfield is perfectly classical – mid-loaded and with thin, well behaved-boundary layers.



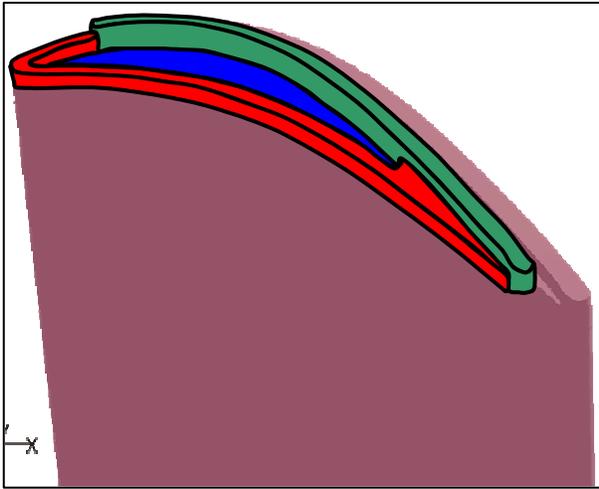
**Figure 12a: Basic blade-blade-mesh**

**Figure 12b: Detail view of the trenched tip**

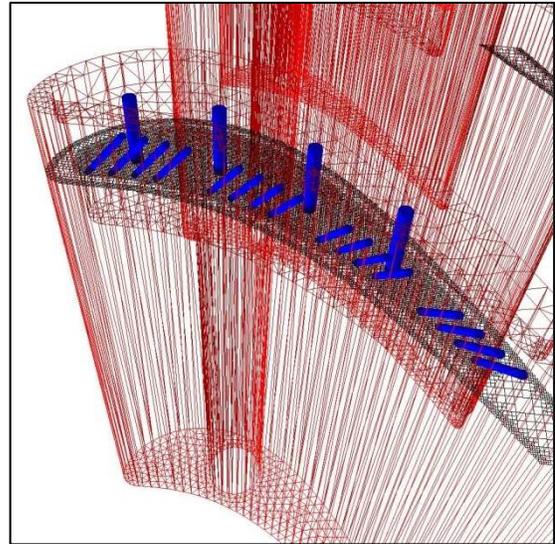
Two objective functions were considered – both individually and in weighted combination. The first was the “cooled row efficiency” which is like the classical mass-weighted total pressure loss ratio with corrections to make allowance for the (presumed dissipated) kinetic energy of the cooling jets. The second objective function was the classical “film-cooling effectiveness” which is really a measure of coolant delivery. The second objective function was evaluated over the tip region of the blade as defined in Figure 13a – our main interest is tip heat load.

Two aspects of the design of the turbine tip were parameterized for automated use within the optimization: the number of dust holes within the trenched tip and the depth of the trench itself. Other parameters could have selected but the aim here is to illustrate the potential of the current *BOXER*-based methodology.

To manage the dust holes a polynomial function was fitted based to the original location of the dust holes and leading/trailing edge location of the bottom squealer surface. Then variable numbers of new dust holes were equally spaced along this polynomial under the control of the optimizer. The total area of the dust holes was kept constant and all dust holes had equal diameters – this keeps their total mass flow approximately constant.



**Figure 13a: The definition of the surface area used to evaluate tip heat load**



**Figure 13b: The virtual tools used to edit the cooling and dust holes into the basic blade geometry**

The dust holes were cut into the solid model of the blade by *BOXER* using virtual tools as illustrated in Figure 13b – the optimizer controls the location of the tools via a script and then *BOXER* uses the tools to edit the underpinning Level Set solid model for the blade and then rebuilds the mesh – re-exporting it in body-conformal FLUENT format. The depth of the trench is varied over a fixed range via *BOXER*'s Free Form Deformation tool; again, after editing the Level Set solid model for the blade, *BOXER* rebuilds the mesh. This geometry editing/re-meshing phase takes about 4 minutes overall and is guaranteed to produce a mesh with solvable mesh quality metrics.

The optimizer selected for this study was a simple Design of Experiment – DoE - which produced a range of solutions for a range of parameters from which a simple Response Surface could then be constructed. Table 1 below shows the 3x3 parameter range selected. All other aspects of the blade geometry – including the tip clearance – and all primary and secondary boundary conditions were kept the same. The entire DoE ran completely automatically with no human intervention – topological geometry editing, mesh regeneration, flow solution & extraction of the objective functions – and on a cluster of 4 PC's took wall-clock 14 days.

<b>4/3.2mm</b>	<b>4/4.1mm</b>	<b>4/5.0mm</b>
<b>7/3.2mm</b>	<b>7/4.1mm</b>	<b>7/5.0mm</b>
<b>10/3.2mm</b>	<b>10/4.1mm</b>	<b>10/5.0mm</b>

**Table 1 The parameter ranges selected for a simple Design of Experiment study: format is number of dust holes/squealer trench depth**

Each simulation deserves attention in its own right from the point of view of better appreciating the revealed flow physics but there is not enough space here; instead some indicative results will summarize the results, then the optimization DoE itself will be discussed. Figure 14 shows predicted blade surface stagnation temperature distributions viewed from the suction side. Clear differences can be seen between the cases in terms of the three-dimensional flow structures in the tip region associated with different number of dust holes and differing over-tip leakage vertical structures.

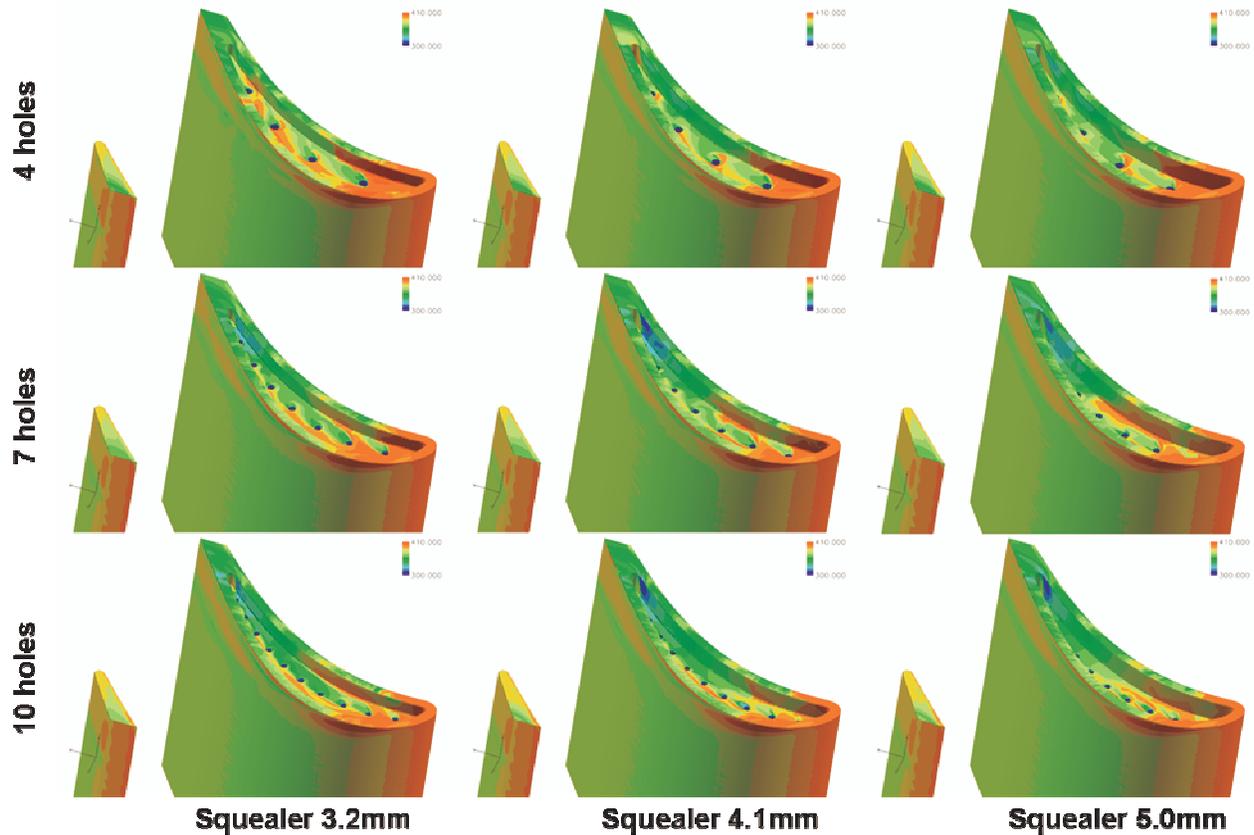


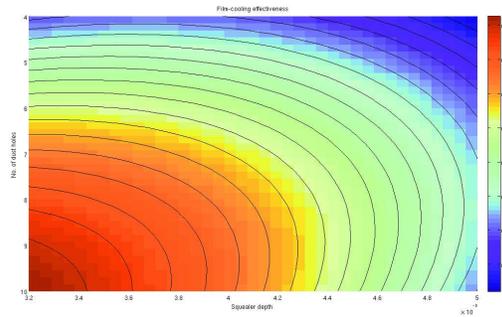
Figure 14: DoE results – surface stagnation temperature – suction side view (range 300-410K)

In more quantitative terms, Figure 15 show contours of a series of least squares quadratic Response Surfaces fitted to two of the various objective functions used in the DoE: film cooling effectiveness & cooled row efficiency. The Response Surface plots are oriented in the same way as the flowfield plots in Figure 14 and the parameterization data in Table 1.

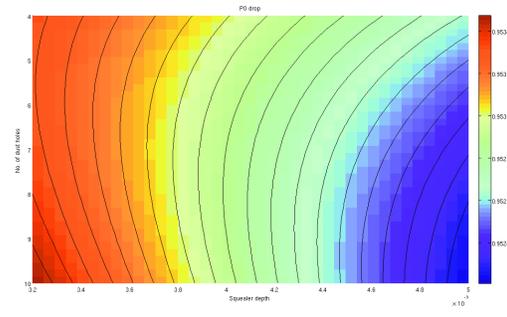
A number of interesting observations arise. In terms of blade row loss coefficient, the loss corrected for the cooling (Figure 15b) is around 11.5% - very high but as expected as coolant mixing is known to be a significant loss mechanism – varying little over the geometric parameter range – mainly because the parameterization tried to keep the coolant mass flow approximately constant by varying dust hole diameter with number of holes.

By contrast, the film cooling effectiveness (Figure 15a) displays a much larger variation – 33% to 37% - with geometric parameterization. For small squealer trench depths (left side of the plots) there is clearly a benefit to having a fixed coolant flow discharge in smaller individual amounts through a much larger number of dust holes; indeed in Figure 14 the delivery of coolant to the trench is visibly better in the case of 10 holes/3.2mm than 4 holes/3.2mm. For deeper trenches, the optimum number of dust holes becomes rather fewer approaching around 8 for a 5mm trench.

Overall, this rather rudimentary automated optimization represents strong evidence for the efficacy of the Level Set based geometry model – every parameterization produced a run-able mesh, every result was valid – no human intervention was required at all. This would not have been possible with more conventional, scripted simulation system.



**Figure 15a: film cooling effectiveness – range 0.335 (blue)-0.370 (red)**



**Figure 15b: cooled row efficiency – range 0.881 (blue)-0.884 (red)**

## VII. Concluding Remarks

This review paper has described the application of the Level Set methodology as a Spatial Occupancy solid modeling kernel to support a scalable, parallel simulation system. Incorporated as the heart of our *BOXER* software system we have demonstrated a range of real-world engineering applications including automated design optimization of complex geometries like cooled gas turbine blades.

Our near-term plans include extending the *BOXER* paradigm to multi-physics applications like structural and thermal analysis and rapid prototyping in general.

## VIII. References

- 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
- Aftosis MJ, Berger MJ & Melton JE, “Robust and efficient Cartesian mesh generation for component-based geometry” *AIAA J.*, 36, 6, 952-, 1998
- Aftosis MJ, Berger MJ & Murman SM, “Application of space filling curves to Cartesian methods for CFD” *AIAA Paper* 2004-1232, Reno January 2004
- Baerentzen A “Volume sculpting: intuitive, interactive 3D shape modelling” *IMM*, May 2001
- Bremer P-T, Porumbescu SD, Kuester F, Hamann B, Joy KI & Ma K-L “Virtual clay modelling using adaptive distance fields” ,xxx, 2001
- Dawes WN, Dhanasekaran PC, Demargne AAJ, Kellar WP & Savill AM, “Reducing bottlenecks in the CAD-to-mesh-to-solution cycle time to allow CFD to participate in design” *ASME Journal of Turbomachinery*, 2002
- Dawes WN, Kellar WP, Harvey SA, Dhanasekaran PC, Savill AM & Cant RS “Managing the geometry is limiting the ability of CFD to manage the flow” *AIAA Paper* 2003-3732, 33rd *AIAA Fluid Dynamics Conference*, 23-26 June 2003, Hilton Walt Disney World, Orlando
- Dawes WN “Building Blocks Towards VR-Based Flow Sculpting” 43rd *AIAA Aerospace Sciences Meeting & Exhibit*, 10-13 January 2005, Reno, NV, *AIAA-2005-1156*
- Dawes WN “Towards a fully integrated parallel geometry kernel, mesh generator, flow solver & post-processor”, 44<sup>th</sup> *AIAA Aerospace Sciences Meeting & Exhibit*, 9-12 January 2006, Reno, NV, *AIAA-2006-45023*
- Dawes WN, Harvey SA, Fellows S, Favaretto CF & Vellivelli A “Viscous Layer Meshes from Level Sets on Cartesian Meshes”, 45<sup>th</sup> *AIAA Aerospace Sciences Meeting & Exhibit*, 8-11 January 2007, Reno, NV, *AIAA-2007-0555*
- Dawes WN, Harvey SA, Fellows S, Eccles N, Jaeggi D & Kellar WP “A practical demonstration of scalable, parallel mesh generation” 47<sup>th</sup> *AIAA Aerospace Sciences Meeting & Exhibit*, 5-8 January 2009, Orlando, FL, *AIAA-2009-0981*
- Dawes WN...*ASME Paper*, Orlando FL, 2009 (under review)
- Galyean TA & Hughes JF “Sculpting: an interactive volumetric modelling technique” *ACM Trans., Computer Graphics*, vol.25, no.4, pp 267-274, 1991
- Glassner AS “*Graphics Gems*” Academic Press, 1990+
- Haines R & Follen GL “Computational Analysis Programming Interface” *Proc. 6<sup>th</sup> Int. Conf. On Numerical Grid Generation in Computational Field Simulations*, Eds. Cross, Eiseman, Hauser, Soni & Thompson, July 1998.
- Jones MW & Satherley R “Voxelisation: modelling for volume graphics” , xxx, 2000

Kellar WP, Savill AM & Dawes WN "Integrated CAD/CFD Visualisation of a Generic F1 Car Front Wheel Flowfield" Lecture Notes in Computer Science, Vol 1593, 1999

Kellar WP "Geometry modelling in CFD and design optimisation" PhD Dissertation, Cambridge 2003

Kellar WP Private communication, 2008

Janke et al...AITEB2...2005

Osher S & Sethian JA "Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations" J.Comput.Phys, 79, 12-, 1988

Perng K-L, Wang W-T, Flanagan M & Ouhyoung M "A real-time 3D virtual sculpting tool based on modified marching cubes", SIGGRAPH, 2001

Samareh, JA "Status & Future of Geometry Modelling and Grid Generation of Design and Optimisation" J.of Aircraft, Vol 36, no1, Feb 1999

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

Sagan H "Space Filling Curves" Springer-Verlag 1994

Sussman M, Smereka P & Osher S "A level set approach for computing solutions to incompressible two-phase flow" J.Comput.Phys, 114, pp 146-159, 1994

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

Viecelli JA "A computing method for incompressible flows bounded by moving walls" J.Comput.Phys, 8, 119-, 1971

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