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

AUTOMATIC GENERATION OF 3D UNSTRUCTURED HIGH-ORDER CURVILINEAR MESHES

Michael Turner[1], David Moxey[1], Spencer J. Sherwin[1] and Joaquim Peiró[1]

[1]Department of Aeronautics, Imperial College London
Imperial College London, London, SW7 2AZ, UK
e-mail: m.turner14@imperial.ac.uk, d.moxey@imperial.ac.uk, j.peiro@imperial.ac.uk,
s.sherwin@imperial.ac.uk

Keywords: needs some

Abstract. *The generation of suitable, good quality high-order meshes is a significant obstacle in the academic and industrial uptake of high-order CFD methods. These methods have a number of favourable characteristics such as low dispersion and dissipation and higher levels of numerical accuracy than their low-order counterparts, however the methods are highly susceptible to inaccuracies caused by low quality meshes. These meshes require significant curvature to accuratly describe the geometric surfaces, which presents a number of difficult challenges in their generation. As yet, research into the field has produced a number of interesting technologies that go some way towards achieving this goal, but are yet to provide a complete system that can systematically produce curved high-order meshes for arbitrary geometries for CFD analysis. This paper presents our efforts in that direction and introduces an open-source high-order mesh generator, NekMesh, which has been created to bring high-order meshing technologies into one coherent pipeline which aims to produce 3D high-order curvilinear meshes from CAD geometries in a robust and systematic way.*

1 INTRODUCTION

Unstructured low-order (linear) mesh generation of complex curvilinear domains has reached a level of sophistication and robustness that has seen the computational analysis of fluid flow become common place within numerous industrial applications. However, these linear meshes and their corresponding solvers are often limited to second-order accuracy. Recently, there has been an increased interest in CFD techniques which utilise high-order approximations as an alternative option to increase numerical accuracy within the bounds of reasonable computational cost. Furthermore, these methods can be used on complex domains with unstructured meshes [1] and exhibit higher rates of convergence on properly constructed meshes [2]. As such, high-order finite element analysis of fluid flow has been highlighted as a potential route for advancing CFD towards higher fidelity methods [3].

The development of high-order finite element solvers for unstructured meshes has allowed the development of high-order fluid flow solvers for a range of applications. These solvers have matured into a well-developed set of tools, such as, for example, the *Nektar++* spectral/hp element framework [4]. However, a significant bottleneck in the high-order CFD process is the generation of unstructured high-order meshes [3]. At present, this is one of the key factors preventing the wider uptake of high-order methods for CFD within academia and especially industry [1]. While unstructured high-order mesh generation is not a new area of academic research, the documented examples within academia seldom present the full and robust mesh generation frameworks which are required for non-expert users. The theoretical guarantees of mesh validity that exist for some methods of linear meshes (e.g. Delaunay [5]) do not exist for high-order meshes. This, among other reasons, makes high-order mesh generation a significantly tougher challenge than linear mesh generation.

High-order mesh generators are yet to reach a level of maturity where they can be used by anyone except expert users in a robust and reliable fashion. They have been shown to create valid suitable meshes on complex geometries [6, 7, 8]. However the issue of robustness is yet to be fully addressed. No research into high-order mesh generation has presented a full methodology which demonstrates the ability to systematically generate meshes on arbitrary geometries of interest to industry. In addition to this, the ability for a non-expert user to use the tools, input their own geometries and get a valid mesh without issues is rarely mentioned. Also the applicability of the meshes which can be created, with respect to the solvers and solutions which will be used, is not usually documented in the literature of high-order mesh generation. Before any form of industrial uptake can be truly considered, robustness must be addressed within all high-order mesh generation processes.

Currently, the standard approach adopted by almost all high-order meshing techniques is the *a posteriori* modification of a valid unstructured linear mesh [2, 6, 7, 8, 9, 10, 11, 12, 13]. Under this principle, the faces and edges lying on the surface of the geometry are curved by adding high-order nodes which also lie on the geometry surface. The issues with this approach are well documented [9, 14], as this simple idea presents a key obstacle to high-order mesh generation that must be overcome: in the case of high geometric curvature the curving process can frequently create elements which self-intersect or have near-tangent vertices, making the mesh unsuitable for computational use. The more recent research into high-order mesh generation focuses on the correction of these invalid elements and only for the correction stage. Little consideration is given to the meshing process as a whole. This work can be categorised into two types: those which use PDE solvers to prescribe some deformation onto the mesh nodes, which makes the mesh valid through the use of non-linear elasticity [15], linear elasticity [13],

thermo-linear elasticity [16] or the Winslow equations [7] and those which use numerical optimisation of quality based functionals [8, 6]. While these techniques may well be robust and highly capable, none present a high-order mesh generation framework geared towards end users that are looking to achieve high-order CFD in viscous simulations. The reality of CFD analysis is that, despite how critical meshes are to the resulting computation, mesh generation should ideally be mostly transparent and automatic for the end user. In linear mesh generation, efforts have been made to ensure the process as painless and quick as possible, but in high-order meshing the experience and technical ability of the user has never addressed. In this current situation, industrial uptake of high-order methods will unlikely be achieved in the near future because these users cannot produce the meshes easily and therefore find high-order methods quite inaccessible.

The purpose of this paper is to present our initial efforts to address the challenge of automatic mesh generation. We describe the philosophy behind the methods and the work carried out thus far on a new open-source program for generating 3D unstructured high-order meshes. Our goal is to use them for the CFD simulation of compressible, incompressible, viscous and inviscid flows, and to produce a tool with as much automation and robustness as possible, that will make it accessible to non-expert users. By automatic, we mean that the meshing process should be able to go from a CAD definition of the domain to the final valid high-order mesh, without requiring anything but the most basic of interactions with the user. This means that the program, as well as having the ability to generate high-order meshes needs the ability to handle CAD, generate coarse linear meshes and perform invalid element correction as part of one process. By robust, we mean that the system should work for as wide a range of geometries as possible and be able to produce valid high-order meshes with a good rate of success. This program is called *NekMesh*, which exists within the open-source high-order framework *Nektar++* and has entered a stage of beta testing before more development is conducted and more robust and capable technologies are added in the near future.

This paper begins by describing the program and outlines the core ideas behind the processes used currently at each stage of the pipeline. It also presents a number meshes and shows examples of the CFD results obtained on them and finishes by discussing our plans for future developments.

2 *NEKMESH*

NekMesh is a high-order mesh generation and modification program which can: convert meshes from one format to another; edit meshes whilst retaining high-order information; and generate high-order meshes from a CAD definition. It is set up in pipeline style; a series of modules are constructed and executed in order to arrive at a high-order mesh. A typical execution will involve an input stage, some processing (mesh editing) stages if required and an output. This arrangement means that the program can easily be used for a wide number of applications and can, with relative ease, be set up to read or write the high-order mesh in a number of different formats[1]. The focus of this paper is the generation of high-order meshes from a CAD description. We describe each stage of the pipeline in the following sections.

2.1 CAD INTERACTION

To interact with CAD geometries, functions including loading, modifying and querying the CAD are all required. *NekMesh* and *Nektar++* in general, have been interfaced with *OpenCas-*

[1]These currently include the *Nektar++*.xml, the *Gmsh* .msh and the *PyFR* .pyfrm formats.

cade [17] to provide this facility. *OpenCascade* is a large open-source library encapsulating a vast range of CAD operations and manipulations. Due to the large size of *OpenCascade*, and also to provide a facility to interact with alternative CAD engines, *NekMesh* uses a small wrapper layer between *Nektar++* and *OpenCascade*, thereby reducing the thousands of possible routines down to a few that are needed for mesh generation. Adopting this approach means that the *Nektar++* CAD engine shields developers from the complexity of *OpenCascade*. Consequently, the CAD handling within the program becomes more effective, robust and efficient.

2.2 AUTOMATIC MESH SPECIFICATION

One of the more fundamental challenges of high-order mesh generation is the creation of valid, suitably coarse linear meshes. It can be difficult for a user to define a set of mesh spacing which remains sufficiently coarse, sufficiently smooth, approximate the surface of the geometry well and still have the necessary resolution in the correct regions of the domain. This becomes extremely challenging in highly complex geometries and would usually require the user to go through many time-consuming iterations of generation to arrive at a suitable linear mesh. In *NekMesh*, this problem is alleviated by using a system to automatically obtain a full set of mesh spacings for the whole domain with very little input from the user.

NekMesh uses an octree structure for the spatial decomposition of the domain. Each subdomain, or octant, has a unique mesh specification parameter associated with it, which is queried during the meshing processes. To construct this representation of the domain the radius of curvature R is first sampled over the geometric surface of the domain and it is taken as

$$R = \min\left\{\frac{1}{k_1}, \frac{1}{k_2}\right\}, \tag{1}$$

where k_1 and k_2 are the principal curvatures at the point on a surface parametrised as $\boldsymbol{r}(u, v)$. A routine to ensure optimum sampling is used, such that any octant in the final octree which encompasses any region of geometric surface is guaranteed to contain more than one curvature sampling. The radius R is then related to a mesh sizing parameter δ as

$$\delta = 2R\sqrt{\varepsilon(2-\varepsilon)}. \tag{2}$$

The parameter ε can be viewed as a control of the sensitivity to curvature: decreasing this value increases the number of elements that will be created for a given curvature. Since R has a range from infinity, so too does δ, therefore sensible bounds must be placed on the value of δ such that $\delta_{\min} \leq \delta \leq \delta_{\max}$. In this setting then, $\delta_{\min}$, $\delta_{\max}$ and ε form the only set of user parameters required to obtain a set of mesh spacings.

Clearly, this curvature sampling only has information of the boundary surfaces. The octree is created by first encompassing the whole domain in a octant and then recursively and equally subdividing the octants until a stopping criteria is met. A octant will be subdivided if:

- The curvature samples within the octant differ in the value of δ by more than 10%; or
- The subdivision will not result in the new octants being smaller in size than $\delta_{\min}$; or
- If any of the neighbouring octants to a given octant are due to subdivide according to the two previous criteria, and their subdivision will cause a difference in size between neighbouring octants of more than one subdivision.

The first two points relate only to the curvature samples. Under these two conditions, only octants which contain a portion of the geometric surface will be split. The last point addresses this and ensures that the octree is smooth and therefore suitable to contain a smooth set of mesh spacings. Any octants which contain a number of curvature sampling points are then assigned a value of δ, which is taken as the minimum in the sample set. This only provides mesh spacings in octants which encompass some region of the surface. To complete the set of mesh spacings in the octree, this surface octant information is propagated outward using a mesh gradation criteria. To do this, each octant i is examined in turn. If the octant has any neighbours which have a value for δ, j, the relation

$$\delta_i = \delta_j + 0.1 r_{ij} \tag{3}$$

is used, where r_{ij} is the distance between the centroids of the two octants. This is repeated until all the octants in the domain have a value of δ. To ensure a smooth mesh specification, the values of the δ are then adjusted so that

$$\frac{|\delta_i - \delta_j|}{r_{ij}} \leq 0.1 \tag{4}$$

between an octant i and any of its neighbours j.

This system brings a very important feature to the mesh generation pipeline: automation. It permits a non-expert user to obtain a linear mesh which has suitable resolution in high curvature regions and that can be used for high-order meshing relatively quickly and without having to spend a lot of time setting up a large number of parameters. In regions of high curvature, any mesh generation pipeline is likely to create invalid elements if the linear mesh resolution is too low, but coarseness is the goal for linear mesh generation in the high-order meshing context. This fine balancing act is a challenge for the user. However within this system, very little effort is required and the number of invalid elements will be greatly reduced because the system will add higher resolution in regions of high curvature.

An additional advantage of this approach is that it is easy and computationally cheap to estimate the number of elements to be generated. With coarse meshes being the ideal and the computational cost of high-order methods being relatively high, the ability to predict the number of elements and alter the mesh before wasting any time generating is a powerful tool for the user. Figure 1 shows the accuracy of the element count prediction for three geometries up to 250 thousand elements. Furthermore, it is possible for the user to define refinement regions, with a smaller mesh sizing than would otherwise be specified, in the domain which will work in conjunction with the octree system to define a overall smooth mesh.

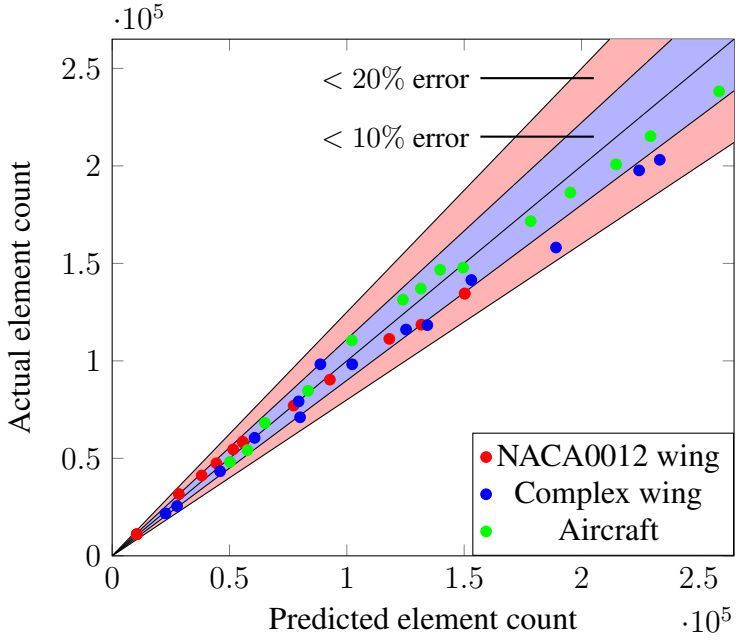


Figure 1: Predicted versus actual element count for various meshes using the automatic octree system

The use of a automatic mesh specification driven by surface topology using an octree structure is not completely new [18]. However, this reference noted that the computational cost and controllability of the method was poor, due to the use of the octree background grid, to the extent that the authors created a new system for automatic mesh definition which did not require a background grid [19]. In the context of generating coarse linear grids for high-order meshing, which inevitably contain far fewer elements, these conclusions are somewhat different. Indeed, we propose an efficient, robust and quality-controlled procedure. This is accomplished by actively keeping the algorithms in the construction of the octree as simple as possible.

Figure 2 shows an example high-order surface mesh of the DLR F6 geometry overlaid with a sliced view of the octree in the symmetry plane [20]. This figure shows that in regions of relatively low and constant curvature the octants are quite large; however in regions such as the tail and nose sections, the octree has smoothly subdivided numerous times to accommodate the very high curvature.

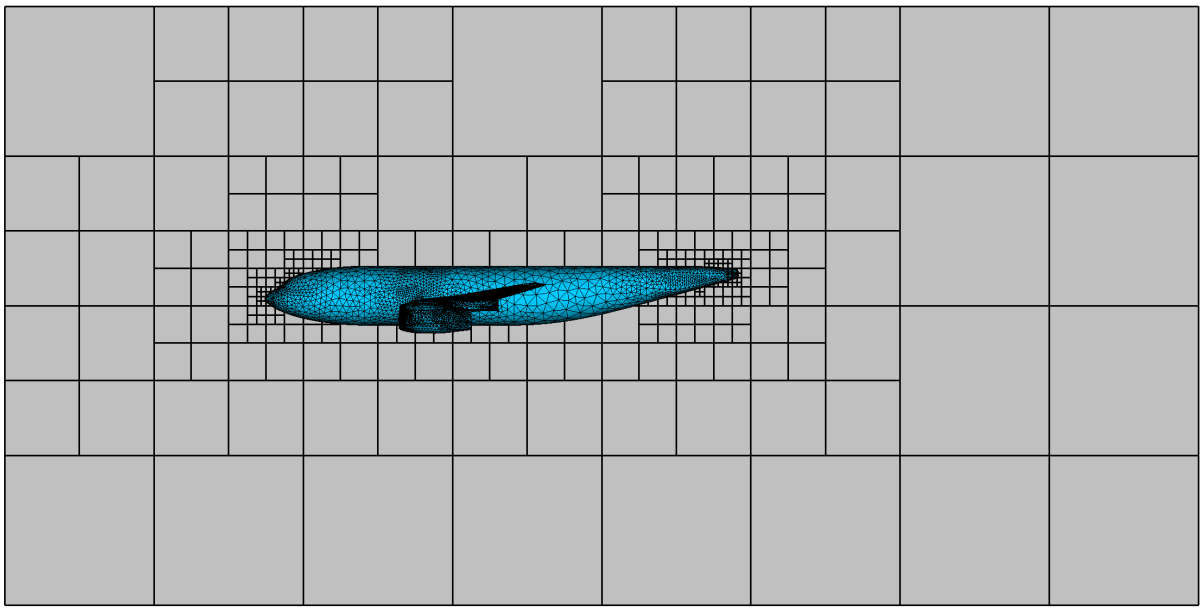


Figure 2: High-order mesh of the DLR F6 geometry with an overlay of the octree.

2.3 LINEAR MESH GENERATION

To automate high-order mesh generation, linear mesh generation must be part of the CAD to CFD pipeline. Using alternative linear mesh generators, and then importing the result, lowers robustness and quality because the routines are not typically designed for high-order use, and CAD information may not be preserved in the process. Some 3D linear mesh generators, such as those included in commercial packages will use a system for the mesh generation which first eliminates all the curvature information and reduces the domain to a linear approximation. This offers high levels of robustness but in linear meshes and the corresponding numerical methods the levels of inaccuracy in the geometric surface caused by this approach are, in most cases, too small to impact the final results. However in high-order methods this level of error is highly significant. Further to this, this approach can result in poor geometric representation in coarse meshes, especially in regions where surfaces are discontinuous. It also means that the surface mesh vertices cannot be located in the parameter spaces of the surfaces without using some form of reconstruction of this information, which can introduce errors and robustness issues in the generation of the high-order surface mesh.

To address these issues, *NekMesh* uses a bespoke approach to the generation of the linear mesh. Using the *NekMesh* CAD engine as the underlying source of information, the linear mesh is built in a bottom-up approach, thus ensuring geometric accuracy is maintained. In this linear mesh generation process, firstly, mesh vertices are fixed to the vertices of the CAD (0D). Curves are then meshed in their 1D parameter space using the end vertices as boundaries and the 3D real distances to decide the distribution in compliance with the mesh spacing within the octree. The 2D parameter spaces of the surfaces are then meshed, using the curve meshes as boundaries, and finally the interior volume is meshed using 3D constrained meshing. This bottom-up approach ensures that the CAD information associated with all the vertices, edges and faces in the surface mesh is kept and can be utilised easily in the high-order stages of the meshing pipeline.

To perform some of the more complex parts of the linear mesh generation, *NekMesh* uses third-party libraries, namely *Triangle* [21] to perform the 2D meshing in the parameter plane

of the surfaces and *TetGen* [22] to perform the interior tetrahedral meshing of the 3D volume. As might logically be expected, the meshing of the surfaces in the parameter plane using a Delaunay based method can produce highly irregular and low quality meshes once the surface mesh is projected into its 3D form, depending on how distorted the parametrisation of the CAD surface is. To overcome this, the linear surface mesh generator in *NekMesh* uses a optimisation step after the 3D surface mesh has been created. In this optimisation, both edge swapping and point relaxation are used to maximise the quality of the surface mesh. Clearly this approach does not preserve the Delaunay characteristic of the initial parameter plane mesh, but this is not our goal so long as the resulting mesh is of good quality. We find that this approach to linear mesh generation is robust and provides a solid foundation for high-order meshing, but because of the dependencies of the CAD, its ability to perform well is tightly linked to the distortion introduced by the CAD parametrisation.

2.4 HIGH-ORDER SURFACE MESHING

The key stage in the generation of a high-order mesh is the creation of the high-order surface. We know that small inaccuracies in the representation of the geometric boundary have a large impact on the flow solution. These inaccuracies include: highly distorted surface elements, i.e. very high curvature in the mesh entities; mesh nodes being a significant distance from the true CAD surface; and under-representation of the geometric curvature due to using a too low polynomial order with too large a element. If the vertex locations of the linear surface mesh are taken to be fixed, producing a high-order surface can be accomplished simply by using an affine mapping of the triangle in the 2D parameter plane to the reference triangle of a high-order element. This can then be used to locate the new high-order nodes in the parameter space, which are then projected into 3D using the CAD engine. However, this means that the high-order triangles will inherit the distortion of the CAD surface, lowering the quality of the mesh and in some cases causing invalid elements. To address this issue, *NekMesh* is equipped with a method to take the high-order surface mesh made using the affine mapping approach and optimise the location of the high-order nodes to reduce distortion. This is done by modelling the mesh entities as spring networks and minimising the spring energy, in a similar approach to the work of [12]. In mathematical notation, this can be expressed as

$$\min f, \tag{5}$$

$$f = \sum_{s} \frac{||\boldsymbol{r_1} - \boldsymbol{r_2}||^2}{w_s}, \tag{6}$$

which states that f, the spring energy, is the sum over all the springs in the system, where $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$ are the 3D locations of the nodes at the ends of the springs and w_s is the inverse of the spring stiffness, which is calculated as a function of the nodal point distribution being targeted. Because the linear mesh vertices are held fixed during this procedure, the problem can be reduced to an entity-by-entity approach. First we optimise mesh edges that lie on curves; then edges that lie on surfaces; and finally interior triangle faces that lie on CAD surfaces. In the first case (edges on CAD curves), the problem is a 1D optimisation of spring system in the curve's parameter space t, so that

$$f = \sum_{i=1}^{P} \frac{||\boldsymbol{r}(t_{i+1}) - \boldsymbol{r}(t_i)||^2}{w_i}, \tag{7}$$

where i are the $P+1$ nodes along the high-order edge. Here, P is the polynomial order of the mesh being created and $w_i = z_{i+1} - z_i$, where z_i is the i-th entry in the distribution of nodal triangular points in the reference space of an edge, where $-1 \leq z \leq 1$. The initial values of t used come from the linear 1D mapping

$$t_i = t_1 \left(\frac{1 - z_i}{2} \right) + t_{P+1} \left(\frac{1 + z_i}{2} \right) \quad i = 1, ..., P+1. \tag{8}$$

where t_1 and t_{P+1} are the parametric coordinates of the end nodes of the edge, which are the vertices in the linear mesh and are considered to be fixed.

Performing the optimisation of the edges which lie on the CAD surfaces follows a very similar procedure but is formulated in the 2D parameter plane, i.e

$$f = \sum_{i=1}^{P} \frac{||\boldsymbol{r}(u_{i+1}, v_{i+1}) - \boldsymbol{r}(u_i, v_i)||^2}{w_i}. \tag{9}$$

This procedure reduces the distortion found in the high-order edges by minimising the length of the edge; that is, the optimised high-order edge will lie approximately on the geodesic between the two end points on the surface.

The procedure for optimising the location of face interior nodes requires a slightly alternative approach. The system is considered as a set of freely movable nodes, consisting of those nodes lying on the interior of the triangle, and a set of fixed nodes which lie on the edges. Each of the free nodes is connected to a system of six surrounding nodes by springs, and this is the system which is minimised. In a triangle of order P, there are $(P-2)(P-1)/2$ interior nodes. The function f is then written as

$$f = \sum_{i=1}^{(P-2)(P-1)/2} \sum_{s}^{6} \frac{||\boldsymbol{r}(u_i, v_i) - \boldsymbol{r}(u_s, v_s)||^2}{w_s}, \tag{10}$$

where w_s is calculated as the distance between the two nodes in a reference equilateral triangle, shown in the figure below along with the connectivity of the springs. The choice of a six spring system means that the method is applicable to any point distribution at any order. For example, Figure 3 shows a $P = 4$ triangle with a Gauss-Lobatto-Legendre distribution along the edges and a triangular Fekete distribution for the face interior points. To optimise the energy of the system a bounded version of the BFGS algorithm is used. This bounding is necessary due to the limits of the parameter space in the CAD entities [23]. One of the primary advantages of this approach is the availability and low cost of computing analytical gradients.

Figure 4 shows the effectiveness of this optimisation procedure. The left-hand figure shows the surface mesh before optimisation, and the right-hand figure after optimisation of the spring networks. In this case, the highly distorted CAD surface of the rounded leading edge of a wingtip causes suboptimal surface mesh generation. The figure clearly shows that the high-order triangles are deformed under the linear mapping. However, when this optimisation procedure is performed, the mesh edges approximate geodesic lines better and the resulting surface mesh is smoother.

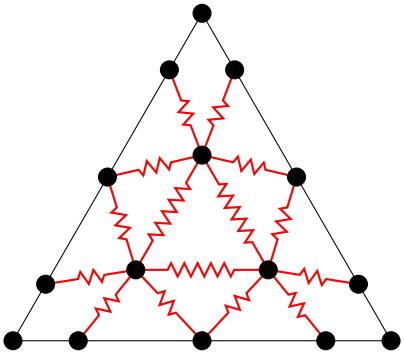
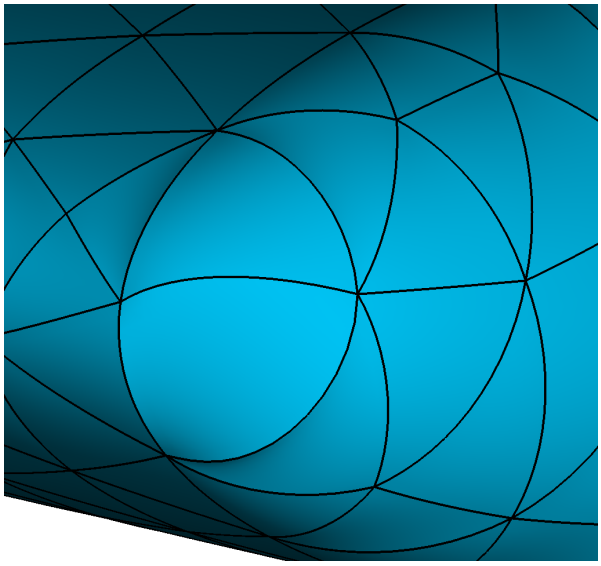
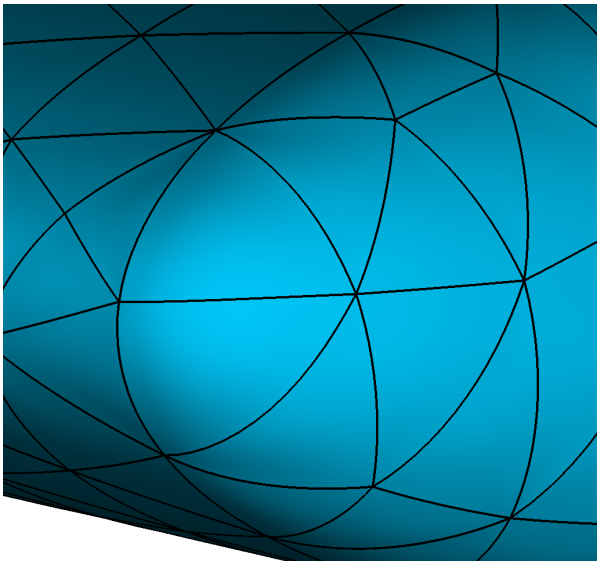


Figure 3: Reference Fekete distribution of points in a fourth-order triangle and the six spring system linking the free nodes.



(a) Unoptimised high-order surface.



(b) Optimised high-order surface.

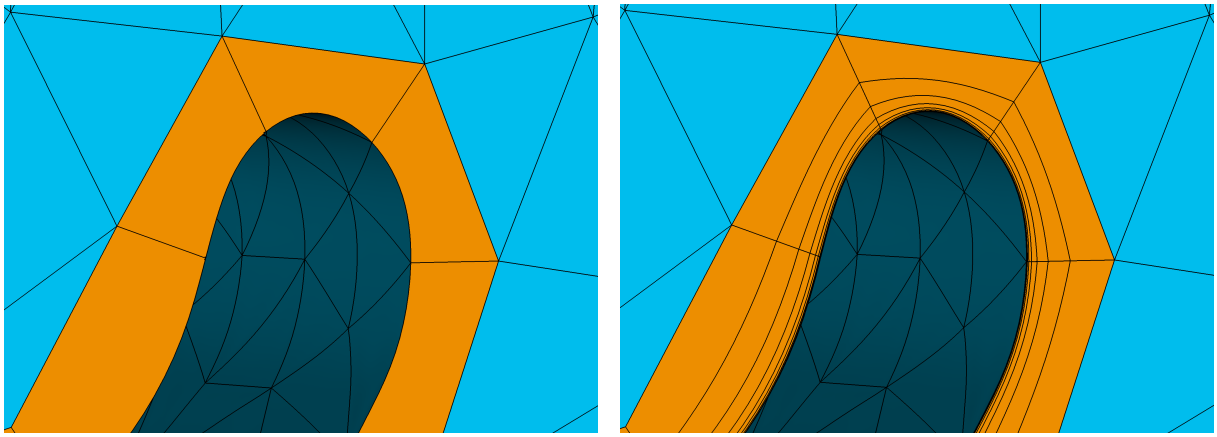
Figure 4: The high-order surface created using the affine mapping with (b) and without (a) optimisation in a region of high distortion in the CAD surface.

2.5 BOUNDARY LAYER GENERATION

The generation of anisotropic boundary layers to achieve low y^+ values has gone somewhat unnoticed in the recent research associated with creating robust high-order mesh generators, but is vital for the analysis of viscous flows. The sharp gradients in flow that are present in viscous simulations require highly stretched elements of a high aspect ratio of 100:1 and above in the wall-normal direction. This therefore poses a significant challenge for high-order mesh generation, since imposing surface curvature on thin elements will almost certainly yield self-intersecting elements in regions of high curvature.

To generate high-order boundary layers, *NekMesh* generates prismatic elements on the geometric boundary and tetrahedra in the rest of the mesh. The system uses a process whereby it inserts a linear 'macro' prismatic layer into the mesh of the desired thickness. This occurs after surface generation but before volume generation. The rest of the volume generation continues from that point using the top surface of the prisms as a pseudo-surface to mesh from. Once the high-order surface has been generated, which will curve one of the triangular faces

of the prism, the macro-prism is then split using an isoparametric approach [24]. This allows for the creation of very thin boundary layer elements without self intersection, since the height of the macro-layer can readily accommodate the curvature of the surface. The generation of the linear macro-prism layer represents one of the newest features of the program. Figure 5 shows a boundary layer region of macro-prisms split using the isoparametric approach to produce highly curved valid boundary-layer elements with very high aspect ratios, despite this they remain valid.



(a) (b)

Figure 5: Shows the splitting of a macro prism layer into 8 layers resulting in curved elements with high aspect ratios.

2.6 CORRECTION OF INVALID ELEMENTS

Up to this point, we have imposed the curvature of the CAD surfaces onto the surface triangulation and generated a volume mesh consisting of either tetrahedra (for inviscid meshes) or prisms and tetrahedra (for viscous boundary-layer meshes). However, this approach will invariably lead to the generation of volume elements where the curved faces impinge on another face, leading to a self-intersecting elements that are not suitable for computation. As such, a correction method is required in order to propagate the curvature into the interior of the domain, so as to correct invalid elements and generate a valid mesh.

As discussed in the introduction, there are a range of methods that can be used to fix these invalid elements. We have found that with the curvature targeting resolution from the automated mesh specification, high-order aware surface refinement and the use of prismatic boundary layers which accommodate more curvature than tetrahedra, the number of invalid elements generated is usually quite small. Complex cases with over 50,000 surface elements, providing the user parameters are sensible, tend to produce only a few invalid elements and with prismatic layers in a number of cases we find they can be prevented altogether. This could be alleviated further, or perhaps even eliminated, with high-order aware tetrahedral generation and this is a point for future work.

However, where invalid elements arise at the CAD surfaces, *NekMesh* can incorporate the use of a linear elastic solver, where the mesh is modelled as a solid body and the displacement at the CAD surface can be imposed to push curvature into the domain, as described in [16]. Since this operation is expensive, as it involves the solution of an elliptic PDE, we aim to

reduce the domain on which the deformation is imposed. We identify the invalid elements and their immediate neighbours by considering adjacent elements up to 10 levels deep, where one level is defined as connectivity through an element face. The linear elasticity equations are then solved on the subdomains, and the resulting displacements are imposed on the original mesh. Additionally, if there are still a large number of elements, the elastic solver can be run in parallel. Depending on the initial conditions of the linear mesh it may not be able to reach a valid configuration in all cases; however, we have found this approach to be quite effective in alleviating invalid elements.

3 EXAMPLES

This section describes two examples of complex three-dimensional geometries for which high-order meshes have been created using the *NekMesh* system. In each case we also show a flow solution to demonstrate the applicability and validity of the mesh. The first example is a high-angle of attack NACA0012 wing with a rounded wingtip, which is designed to run at high Reynolds number. Due to the convex nature of the geometry, the addition of a prismatic layer adjacent to the geometry meant that the high-order mesh was valid without needing to resort to the elasticity solver to correct any elements. The second is significantly more complex: a DLR F6 aeroplane geometry [20], an all tetrahedral mesh which did require the use of the linear elastic solver to correct invalid elements. In each case, the high-order mesh was created from the CAD definition using four user parameters: $\delta_{\min}$, $\delta_{\max}$, ε and P, where P is the desired order of the mesh.

3.1 NACA WING

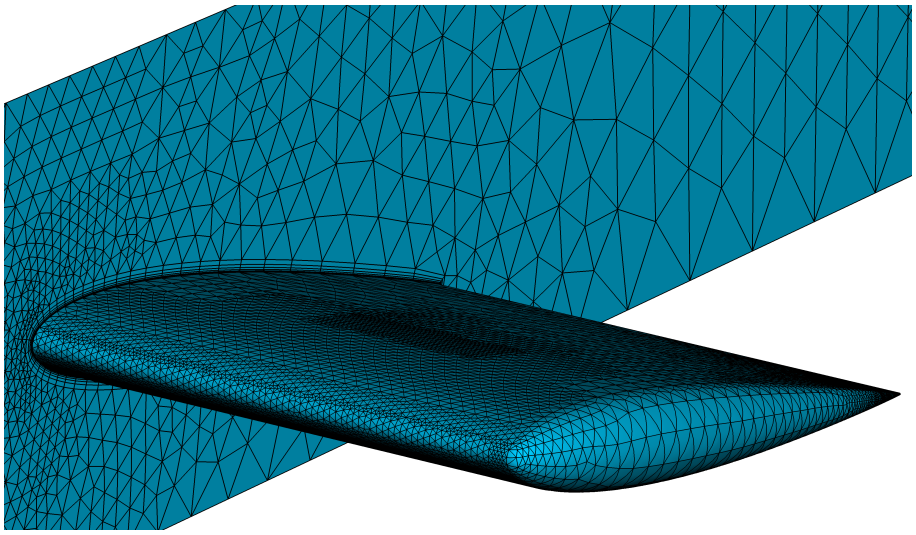
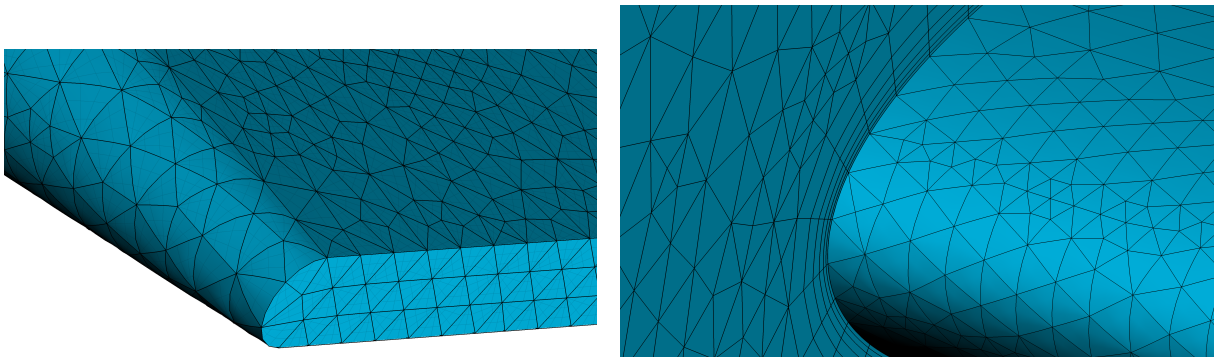


Figure 6: High-order mesh of the NACA wing.

Figure 6 shows a image of the surface of the NACA wing geometry. This mesh has a anisotropic boundary layer, as shown in figure 7. Regions of high curvature such as the leading edge of the wing and the curved wing tip, have increased resolutions compared to other parts of the mesh, as would be expected with the automated specification system. On the suction surface the resolution of the mesh has been manually increased to capture a separation region in the flow solution. Despite this modification, the octree system has ensured that the mesh

remained smooth, without large changes in element volume.



(a) Trailing edge of the mesh near the curved wing tip.

(b) High-order curved boundary in the symmetry plane.

Figure 7: Enlargements of areas of the NACA wing mesh.

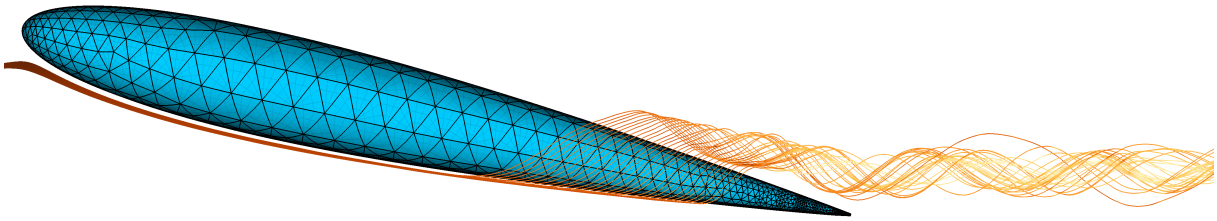


Figure 8: Particle trajectories following the tip vortex in the flow simulation at $Re = 10^5$ computed on a NACA wing mesh wiht $P = 4$.

To demonstrate the validity of the mesh, an incompressible flow simulation at $Re = 10^5$ was performed using *Nektar++*. Of particular interest for this case is the nature of the strong wingtip vortex produced [25]. Figure 8 shows that this vortex lies in the expected position, with the flow separating along approximately the correct location along the wing chord. We have found that in previous experiments, meshes which did not have suitable resolution or boundary layers were not able to properly capture these features.

3.2 DLR F6

The second example we consider is a significantly more complex DLR F6 jet geometry. A fully tetrahedral Euler mesh was created. Which required the use of the elasticity solver to correct invalid elements, but this number was small due to the resolution used near the surface and use of curvature-aware meshing. Figure 9 shows the mesh, with an inset highlighting the engine intake high curvature region. It can be observed that the automatic system for mesh specification has achieved its goal and added higher resolution to regions of the geometry which have high curvature, such as the engine intake and aircraft nose. Figure 10 shows a solution produced with this mesh for a low Reynolds number incompressible flow. While these results are not necessarily physically meaningful, they do demonstrate the suitability of the mesh. With a viscous boundary layer and significantly more computational time, this mesh could be capable of achieving much more realistic and relevant results, but this is outside the scope of this paper.

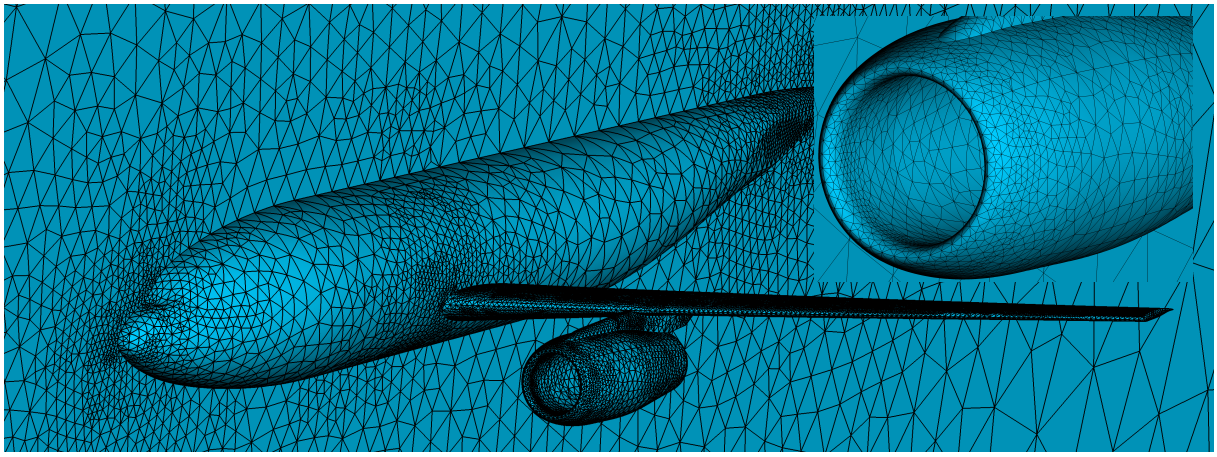


Figure 9: High-order mesh of the DLR F6 geometry. The inset shows an enlargement of the mesh around the front of the engine.

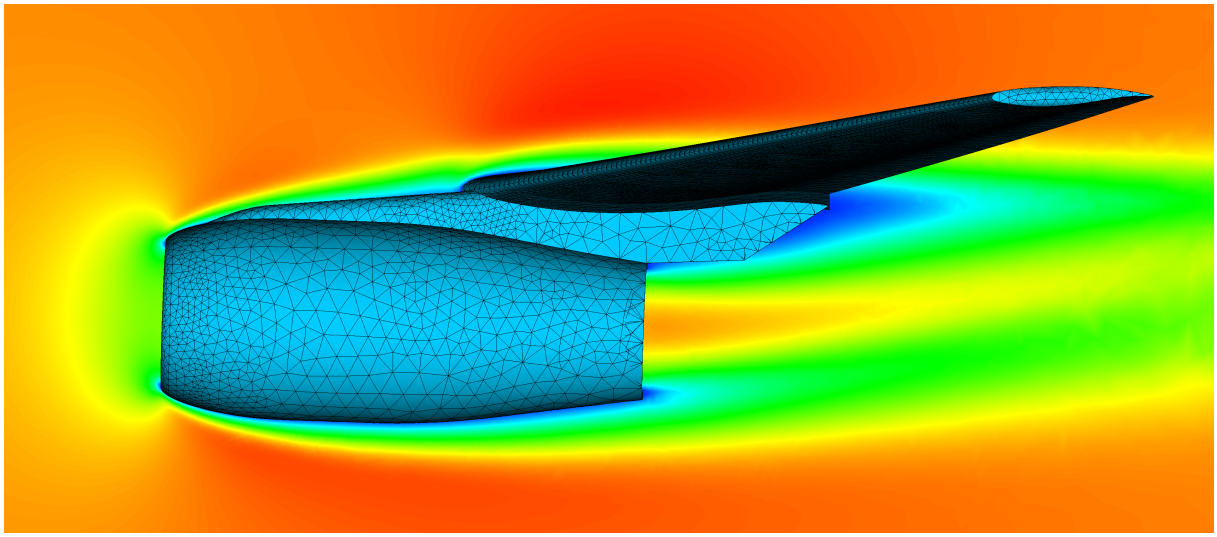


Figure 10: Low Reynolds number flow solution through the engine section.

4 CONCLUSIONS

We have introduced *NekMesh*, a system specifically designed to tackle robustness and complexity issues in the generation of high-order curvilinear 3D meshes for complex geometric cases. The methodology for each stage of the mesh generation pipeline has been outlined and two examples have demonstrated the capabilities of the system. *NekMesh* is currently under development; this article outlines the capabilities of the system as it is today, and marks the beta testing of the preliminary framework. We expect that in the future the structure and methodology of the program will advance and change becoming a much more capable system. The work in the near future will focus around more complex boundary layer generation techniques and more advanced robust methods for *a posteriori* invalid element corrections.

REFERENCES

[1] PE Vincent and A Jameson. Facilitating the adoption of unstructured high-order methods amongst a wider community of fluid dynamicists. *Mathematical Modelling of Natural*

Phenomena, 6(3):97–140, 2011.

[2] O Sahni, XJ Luo, KE Jansen, and MS Shephard. Curved boundary layer meshing for adaptive viscous flow simulations. *Finite Elements in Analysis and Design*, 46(1-2):132–139, 2010.

[3] J Slotnick, A Khodadoust, and J Alonso. CFD Vision 2030 study: A path to revolutionary computational aerosciences. Technical report, NASA, 2014.

[4] C Cantwell, D Moxey, A Comerford, A Bolis, G Rocco, G Mengaldo, D Grazia, S Yakovlev, J Lombard, D Ekelschot, B Jordi, H Xu, Y Mohamied, C Eskilsson, B Nelson, P Vos, C Biotto, M Kirby, and SJ Sherwin. Nektar++: An open-source spectral/hp element framework. *Computer Physics Communications*, 192:205–219, 2015.

[5] D. T. Lee and B. J. Schachter. Two algorithms for constructing a Delaunay triangulation. *International Journal of Computer and Information Sciences*, 9(3):219–242, 1980.

[6] A Gargallo-Peiró, X Roca, and J Sarrate. Optimization of a regularized distortion measure to generate curved high-order unstructured tetrahedral meshes. *International Journal for Numerical Methods in Engineering*, 103:342–363, 2015.

[7] M Fortunato and P-O Persson. High-order unstructured curved mesh generation using the Winslow equations. *Journal of Computational Physics*, 307:1–14, 2016.

[8] T Toulorge, C Geuzaine, JF Remacle, and J Lambrechts. Robust untangling of curvilinear meshes. *Journal of Computational Physics*, 254:8–26, 2013.

[9] S Dey, RM O'Bara, and MS Shephard. Towards curvilinear meshing in 3D: the case of quadratic simplices. *Computer-Aided Design*, 33(3):199–209, 2001.

[10] XJ Luo, MS Shephard, RM O'Bara, R Nastasia, and MW Beall. Automatic p-version mesh generation for curved domains. *Engineering with Computers*, 20(3):273–285, 2004.

[11] MS Shephard, JE Flaherty, KE Jansen, X Li, XJ Luo, N Chevaugeon, JF Remacle, MW Beall, and RM O'Bara. Adaptive mesh generation for curved domains. *Applied Numerical Mathematics*, 52:251–271, 2005.

[12] SJ Sherwin and J Peiró. Mesh generation in curvilinear domains using high-order elements. *International Journal for Numerical Methods in Engineering*, 53(1):207–223, 2001.

[13] ZQ Xie, R Sevilla, O Hassan, and K Morgan. The generation of arbitrary order curved meshes for 3D finite element analysis. *Computational Mechanics*, 51(3):361–374, 2012.

[14] SJ Sherwin and J Peiró. High-order automatic mesh generation: optimal p-type surface generation. In *16th IMACS World Congress*, Lausanne, 2000.

[15] P-O Persson and J Peraire. Curved mesh generation and mesh refinement using Lagrangian solid mechanics. In *47th AIAA Aerospace Sciences Meeting*, pages 1–11, Hampton, Virgina, 2009.

[16] D Moxey, D Ekelschot, U Keskin, SJ Sherwin, and J Peiró. A thermo-elastic analogy for high-order curvilinear meshing with control of mesh validity and quality. *Computer-Aided Design*, 72:130–139, 2016.

[17] OpenCascade, 2016.

[18] L.A. Kania and S.Z. Pirzadeh. A geometrically-derived background function for automated unstructured mesh generation. In *17th AIAA Computational Fluid Dynamics Conference*, page 5240, Toronto, 2005.

[19] S Z. Pirzadeh. Advanced unstructured grid generation for complex aerodynamic applications. *AIAA Journal*, 48(5):904–915, 2010.

[20] O. Brodersen and A. Stuermer. Drag prediction of engine-airframe interference effects using unstructured Navier-Stokes calculations. In *19th AIAA Applied Aerodynamics Conference*, number 2001-2414, Anaheim, 2001.

[21] J Shewchuk. Triangle: Engineering a 2D quality mesh generator and delaunay triangulator. *Applied Computational Geometry: Towards Geometric Engineering*, 1148:203–222, 1996.

[22] H Si. TetGen, a Delaunay-based quality tetrahedral mesh generator. *ACM Transactions on Mathematical Software*, 41(2):11:1–11:36, 2015.

[23] RH Byrd, P Lu, J Nocedal, and C Zhu. A limited memory algorithm for bound constrained optimisation. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[24] D Moxey, M Green, SJ Sherwin, and J Peiró. An isoparametric approach to high-order curvilinear boundary-layer meshing. *Computer Methods in Applied Mechanics and Engineering*, 283:636–650, 2015.

[25] J-E. W. Lombard, D. Moxey, S. J. Sherwin, J. F. A. Hoessler, S. Dhandapani, and M. J. Taylor. Implicit large-eddy simulation of a wingtip vortex. *AIAA Journal*, pages 1–13, 2015.