University of Exeter coursework header sheet

ECMM102 Group Project (Meng) (A, TRM1+2 2016/7)

009412

1027442

630022828

Coursework:	Individual contribution to the group achievement
Submission Deadline:	Thu 4th May 2017 12:00
Personal tutor:	Dr Hao Qin
Marker name:	G_Tabor
Word count:	13913

First marker's comments	Indicative mark
Second marker's comments	Second mark
Moderator's comments	Agreed mark

I2 Report

Automatic structured and unstructured mesh generation using Pointwise

Joseph Gowans

2016

4th year MEng Group Project

Signed.. [illegible]

College of Engineering, Mathematics, and Physical Sciences
University of Exeter

I2 Report
ECMM102

Title: Automatic structured and unstructured mesh generation using Pointwise

Word count: 13913
Number of pages: 40

Date of submission: Thursday, 04 May 2017

Student Name: Joseph Gowans

Programme: Meng Mechanical Engineering

Student number: 630022828

Candidate number: 009412

Supervisor: Dr Gavin Tabor

Abstract

This individual report represents part of group project that aimed to optimise the shape of a draft tube using computational fluid dynamics and machine learning. A Bayesian optimiser, integrated with the numerical solver OpenFOAM optimised the diffuser section of a draft tube using a Catmull-Clark spline representation. This report presents the work done by this author to contribute to the group objectives.

Within any CFD process, a mesh must first be created in order to discretise the geometry and therefore solve the Navier-Stokes equations to provide a solution. This meshing process often requires large amounts of user input to produce a mesh of good quality which will then go on to produce accurate CFD results. The aim within this project was to automate this process to allow use within an optimiser. An automated structured meshing tool was created using the recent developments in scripting within the commercial meshing software Pointwise. It was shown to reliably produce good quality meshes for up to 4 control point splines. Processes were put in place to inhibit inaccurate solutions as a result of poor quality meshes influencing the optimiser. In addition, a novel framework to allow use of a combination of meshing tools within an optimiser was implemented. This however, still requires further work to reduce the sensitivity of the solutions to the meshing tool, before this can reliably be used.

Keywords: Design Space, Draft Tube, Optimisation, CFD, Mesh

Acknowledgements

Many thanks to Dr Steven Daniels and Dr Alma Rahat for their continued help and guidance throughout this project, as well as to Prof Gavin Tabor for his help and supervision.

Table of contents

1 Introduction and background

As part of the system of a reaction turbine, a draft tube is installed at the runner outflow in order to diffuse the flow leaving the turbine. It is shaped such that it creates a pressure difference between inlet and outlet to increase the work done over the turbine. This can help to recover some of the available pressure head that is lost at the exit of the turbine and can appreciably improve the overall efficiency of the system. Since these types of reaction turbines are used within the hydroelectric industry, large amounts of fluid pass through the system over long periods of time. Therefore, even small improvements in draft tube efficiency can have large impacts on the amount of energy produced from a hydroelectric power plant.

Using machine learning algorithms, the design of the draft tube can be altered and then evaluated using computational fluid dynamics. The shape can then be optimised with the goal of reducing the pressure drop across the draft tube. This was the main objective for the group project.

Once the geometry has been perturbed by the optimiser, the flowfield must be discretised into a mesh before it can be solved. Within the CFD process, meshing remains one of the most challenging and influential aspects [1]. Since the meshing process requires a large amount of manual input and is very specific to each individual geometry, automating this procedure is challenging and a certain amount of control over the specifics of the mesh and its quality are lost within an optimiser. User intervention must be eliminated from the meshing process; this is one of the most challenging aspects of producing an automated meshing tool. Creating meshes reliably and with good quality over many geometries is of primary importance within a CFD optimiser, as without it solutions to any degree of accuracy cannot be produced.

This report presents the use of Catmull-Clark sub-division splines to represent the geometry and the use of Pointwise Glyph scripting as a bottom up meshing tool used to re-mesh the altered geometry for each evaluation.

1.1 *Specific Aims*

I. Develop a robust structured meshing tool.
II. Define methods to inhibit poor quality meshes.
III. Automate meshing procedures using Python.

2 Literature review

2.1 *Shape representation*

Much of the work done on shape representation within an optimiser comes from the aeronautics industry on the design of aerofoil sections. The necessity for regular redesign of aerofoils combined with the relative computational inexpense of evaluating the two-dimensional flow, make optimisation a very powerful tool. There are a range of techniques available in order to represent geometry with varying degrees of detail. Vassburg et al [2]present the use of a single Bezier curve to represent aerofoil cross section, in a drag reduction optimisation, allowing interior control points to vary, whilst other boundary points remain fixed. Alternatively, Carrier et al [3], use a B-spline method to represent their aerofoil geometry, using two curves to model the upper and lower surfaces with varying numbers of control points. Initially, it would seem that the two methods are very similar in nature, curves represented by a finite number of control points, and little sensitivity of an optimiser to the representation would be present. However, Vassburg and Johnson [4], who further the work done by [2] and [3], analyse this sensitivity, comparing the use of Bezier curves to that of B-splines under similar conditions. The results of the study conclude that for 6 control points the disparity between the reported optima is '*quite astonishing*' [4], with the optimal drag for a Bezier represented aerofoil converging at 346 counts, compared to 126 counts for the B-spline representation. The authors concede that their understanding of the resulting discrepancy is not yet complete, however, they postulate that the root cause may lie in the global vs local control of each representation.

Some of the earlier work performed on parameterising draft tube geometry is done by Marjavaara and Lundstrom [5] in which, the radius only of a sharp heel draft tube, was optimised. Due to the simplicity of the parameterisation, the design space is very limited in size and thus a purely local optimum is found. In order to fully parameterize a draft tube McNabb et al [6] suggest that 100+ design variables are required to fully explore the design space and return a global optimum. This is however infeasible, given the computational expense of the problem, and as a result the authors experiment with a Swedish parameterisation. This method abstracts the width profile and hence the general shape, interestingly, using a parametrically derived diffusion curve along with a defined section 'type'. Breaking down the entire geometry into 26 parameters, of which 12 were offered as design variables.

A more common parameterisation technique for interior ducts includes the work of Eisinger and Ruprecht [7] who abstract the geometry into a finite number of symmetrical cross sections. Parameterising each rectangular section into height, width and radius of each corner fillet, with straight connections between each section.

2.2 *Automatic Meshing Techniques*

Rhoads [8] comments that most, if not all of the open source tools for mesh generation, use a cut cell method to facilitate automation in a stable and robust manner. The author highlights that whilst they are able to produce meshes for almost any geometry there is a well-known disadvantage; the sacrifice of control and sometimes, for more complex geometries, the presence of a refined surface mesh and boundary layer cells. Daniels et al [9], employ the use of the OpenFOAM tool snappyHexMesh for the purpose of mesh regeneration within the optimisation of an interior duct. The resulting performance of the meshing tool echoes the sentiments of Rhoads [8] , whilst also adding that limitations of the tool for mesh regeneration within an optimiser being that it can only remove sections from the original volume mesh, potentially limiting the exploration of the design space.

Whilst there are advantages of cut-cell methods in terms of grid control, Rhoads [8] points out that a bottom up approach, in which the topology of the mesh is much more explicitly defined, is often preferred in order to reduce the loss of mesh element quality and therefore the sensitivity of solutions to the grid. These methods often require more time investment and more skilled user input than an automated cut-cell approach, although as alluded to by Fabritius and Tabor [10] and also [9] the time required to achieve a satisfactory mesh from these automated tools, is not insignificant.

However, up until recently [11], bottom-up meshing tools such as Pointwise were not able to be automated within an optimisation procedure. The introduction of the ability to automate this software, through their scripting language has provided an alternative and powerful tool for automated meshing. A higher order structured mesher has been developed for finite element analysis by Grigoriev et al [12], using the Pointwise scripting language. The authors highlight the tool's quality by its ability to produce higher order, fully structured meshes of high quality, with accurate geometric representation whilst being '*both general and robust*' [12]. Whilst this is used for a finite element analysis, it highlights the benefits and power that this tool can provide for automated procedures.

Guibault et al [1] also present their developments on the automation of a robust structured meshing tool for use with various turbomachinery components. Much of their work is devoted to developing feature detection algorithms that allow for the domain to be automatically subdivided into blocks based on the geometric characteristics of the domain. The authors comment that the success of their meshing software is measured by its ability to create meshes of good quality across a range of geometry modifications and components. Mcnabb et al [6] utilise this meshing tool within their draft tube optimisation, however, the specific framework and procedures of their 'in-house' meshing tool are not detailed in the report.

Another solution to the problem of meshing within a CFD shape optimiser is to move or distort an initial and refined volume mesh; this method was originally formulated for problems involving a moving boundary within the solver. Whilst not specifically designed for this process, a shape conformed mesh is produced once the boundary has been moved and the process is therefore applicable and has been utilised within optimisers [9].

Eriksson [13] uses the transfinite interpolation method (TFI), which is an algebraic method wherein each mesh point is generated, using univariate coordinate interpolation, by marching in turn between the inner and outer boundaries; this is known to be the simplest way to generate structured grids automatically. There are limited measures in place to control the cross-over of grid elements and as such, this method is attractive, mainly for simple domains with predominantly convex boundaries and has been extensively used for the meshing of aerofoils [14]. Allen [15] also comments that this interpolation method allows for limited control over mesh quality, since each line of mesh elements is independent of its neighbours.

Furthering this work, hyperbolic methods have been proposed by [16], in which all grid points within each plane are solved simultaneously, marching outwards from the inner boundary. This method, however, does not consider any downstream information and the distribution of outer boundaries cannot be prescribed. This makes it rather limiting in terms of a multi-block topology. Hybrid methods have been generated that involve parabolic approaches at the boundaries that account for this lack of downstream information [17]. However, these methods have been shown to be somewhat sensitive to the curvature and smoothness of the boundaries [15] [17], and in this sense the robustness of this method is limited.

Allen [15] comments that the elliptic equation schemes produced by [18] [19] generally produce the meshes of the highest quality of all the grid distortion approaches, with recent work allowing for continuity across multi-block domains [20]. The author also comments that some slight trade-offs, as a result of the higher final mesh quality, are a slight increase in computational expense and a marginal limit to the control of the exact spacing at boundaries.

Tez-duyar et al [21] make use of the Arbitrary Lagrangian-Eularian (ALE) grid motion approach, which is reported to allow for high quality grids to be maintained at the boundary of the mesh. The boundary layer formation is achieved through the motion of the grid vertices in order to reshape the surrounding cells, with the motion of the vertices governed by a Laplacian smoothing scheme. This high quality of the distorted mesh is highly advantageous for maintaining accurate modelling of the flow physics within the boundary layer. However, Daniels et al [9], describe degradation of the cell quality close to the perturbed boundary when the mesh is deformed from a simple structured block mesh.

Another technique for mesh movement is implemented by Stein et al [22]; in this method the motion of the nodes is guided by the equations of elasticity and can be used in conjunction with the stabilised space-time formulation [23] as well as the ALE method [21]. Stein et al, use a method they previously developed to solve the equations of elasticity for thin boundary layer elements, this is done separately to those in the rest of the mesh. The boundary layer cells are attributed a higher stiffness in order to limit the distortion of these cells and maintain high quality elements in this region. The authors' comment that when the stiffness relationship between the two regions was optimal, the boundary layer elements acted in a similar fashion to that of the solid boundary and the resulting mesh elements, even for large displacements, were of high quality.

Jasak [23] offers alternative mesh adjustment strategies for cases in which the boundary deformation is extreme enough to render mesh motion alone insufficient to maintain grid quality and robustness. For these such extreme cases, Jasak highlights the use of more fundamental mesh topology changes to counter the breakdown or poor distribution of fixed connectivity meshes in these scenarios. The paper states that single primitive operations on a mesh are inadequate to alter the validity and must be performed in batches such that the mesh is almost completely rebuilt and hence, this method is impractical.

Opposed to the techniques mentioned above, Jasak et al [24] employ the use of an immersed boundary method (IBM) which requires no adjustment or re-evaluation of a body-conformal mesh. Instead, boundary conditions are imposed upon the faces of a background volume mesh at the location where the specified geometric boundary is situated, thus eliminating the computational effort and possible failure of a re-meshing process. Daniels et al [9] put to use Jasak et al's [24] recent implementation in Foam-extend by applying it to the Pitzdaily OpenFoam tutorial case within an optimisation procedure. The authors of [9] remark that although the simplicity, robustness and flexibility of the IBM method are highly advantageous, it is limited in that, currently, there is no control over the grid resolution within the boundary layer and the resulting solutions do not yet mimic those produced by models with body-conformal meshes.

2.3 Mesh Quality

As Vassberg and Jameson [4] allude to, the near optimal solution of a CFD optimiser can be highly sensitive to the mesh quality produced at each evaluation of the objective function. The authors explain that an inaccurate solution can mislead the optimiser, causing it to exploit regions of the design space that in reality are non-optimal, which can lead false optima. To help inhibit this process, Guibault et al [1], impose mesh quality thresholds, discarding geometries that produce mesh elements of bad quality that would have a large impact on the solution.

To provide clarity to the degree to which mesh quality effects solution accuracy, Rhoads [8] , conducts a set of tests on a 2D pure advective transport case, in which the mesh skewness, non-orthogonality and flow alignment are varied. Rhoads shows that the numerical mixing process, as a result of the misalignment of cells, can increase the solution error by one order of magnitude when compared to the baseline interpolations. Non-orthogonality of the mesh cells above 70 degrees is reported to introduce significant error due to an approximation of the gradient, the author also comments that in practice this value would be higher due to the lack of a Laplacian term within the test case. Within OpenFOAM, some levels of non-orthogonality can be corrected by increasing a setting that accounts for this grid error in the solver, which has shown to reduce the impact. The paper suggests that OpenFOAM is highly sensitive to inaccuracies due to skewness because of the way in which the fluxes between adjacent cells are calculated. Rhoads [8] compares the solutions of the test case between meshes formed by the advancing front and Delaunay unstructured meshing algorithms, showing the considerably larger error in the more skewed Delaunay mesh. It is also noted that, although the Delaunay method produces higher skew, the errors induced due to flow alignment are reduced since the mesh is somewhat randomly orientated. It is concluded by Rhoads that, when using OpenFOAM, hex-dominant meshes have far fewer restrictions in terms of the numerics of the solution.

A project similar to that discussed in this report is being undertaken currently, using Pointwise glyph scripting, in which the aerodynamic shape of a big wave surfboard is being optimised with the use of splines as abstraction method [25]. In this project, the authors investigate the use of glyph scripting to generate hybrid meshes using the Pointwise meshing algorithm T-

Rex.

3 Theoretical background and Methodology

3.1 Theoretical Background

3.1.1 Bayesian Algorithm

Bayesian optimisation is a data-driven, surrogate model assisted optimiser that can either be implemented with or without gradient information. As well as the objective function, a Bayesian optimiser uses an acquisition function that can be maximised to determine the next most optimal point at which to query the objective function. Before a Bayesian optimiser can begin, it requires a prior distribution of data points which captures the beliefs of the user to the behaviour of the unknown objective function. Since they are stochastic algorithms they are known to be ideal for *black box* functions such as CFD.

It can be said that CFD within an optimiser can be classed as an approximate fitness function, as the CFD itself is a model - of the real-life scenario - from which a surrogate model is created. If this approximate model produces inaccurate results it can cause the algorithm to converge incorrectly or even stall convergence, thus producing non-optimal results [26] [27]. These potentially inaccurate models can lead to falsely high performance of a particular geometry, potentially misleading the optimiser to exploit a part of the design space which is in actual fact, non-optimal. This can be especially seen when the acquisition function is more exploitative, as is the case here, the expected improvement is used by Ng [28].

3.1.2 Structured Grid advantages

If a good quality structured mesh can be achieved for a particular geometry, it is generally considered to be superior to an unstructured mesh [29]. One reason for this is that structured grids allow for far better control over the interior node locations and cell sizes, as these are generally defined from exterior nodes, controlled by the user. Intrinsically, structured hexahedra contain face pairs which can be aligned with the oncoming flow, whereas, due to their random nature this is not possible for unstructured grids. This alignment allows for the majority of the flux to move through one of the face pairs, of which the benefit is explained later on. Structured meshes are also implicitly larger computational ‘molecules’, as one cell has six neighbours as opposed to four for a tetrahedral cell. This is an advantage for cell centre based codes such as OpenFOAM, as each element has more information about the surrounding fluxes and gradients. Inherently, unstructured tetrahedral meshes use six times as many elements as structured hexahedra to discretise the same space, making unstructured meshes far more computationally expensive.

3.1.3 Mesh Quality

As described in the literature review, the quality of a mesh can act as a proxy for accuracy in a numerical solution. The following section outlines some of the metrics that can be used to measure grid quality and how they are calculated.

3.1.3.1 Skewness

As mentioned in the literature review, high skewness can introduce a significant error into a solution. Skew can be said to be a measure of the difference in position of the face centre vector and the cell centre vector in a given cell and essentially represents the level of distortion of a cell. It is displayed as a ratio ξ, between 0 and 1 with 0 being the ideal value and 1 being the worst possible level of distortion. It is calculated by equation (1), where, as shown in the figure 1, δ is the distance between the actual face centre and the interpolated face centres and D, is the distance between the adjacent cell centres.

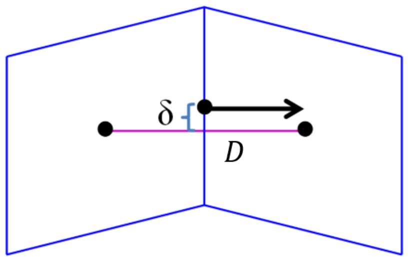


Figure 1 'Diagram showing calculation of skewness between faces'

$$\xi = \frac{\delta}{D} \tag{1}$$

By virtue of the way in which the face-centred pressure gradients are calculated using pressure values taken from the cell centres, a high skewness value, meaning a *large* discrepancy between the positions of cell and face centres can result in a numerical diffusion process. OpenFOAM is known to be slightly more sensitive to the effect of skewness, because of the way in which, for finite volume schemes, the fluxes between cells are calculated from the face normal vectors. Therefore, skewness is a value of high importance; it can greatly impact the convergence of a solution, slowing it down, preventing it and in some cases of high skew, causing the solution to diverge completely [10]. Along with convergence, it also has a large impact on the accuracy of the solution, as shown by Rhoads [8]. Zang et al [30] also mention that the redcution in accuracy due to highly skewed cells may be as significant as the reduction from a second to a first order central-differencing scheme.

Fast convergence of solution is a bigger advantage in the case of CFD-optimisation than in a regular CFD simulation, as computational expense is a much larger issue due to the number of simulations to be run. This heightens the importance of the skewness as a performance metric.

3.1.3.2 Aspect Ratio

The aspect ratio of a cell is the ratio of the longest side to its shortest side. The range of the aspect ratio is obviously from 1, a perfect cube or equilateral tetrahedral, which, purely considering the numerics of the solution, is the optimal value, with an almost unlimited upper bound. For unstructured tetrahedral cells the aspect ratio is a massively important factor as it infers other aspects of the mesh such as the skewness. However, for structured hexahedral cells the impact of the aspect ratio is less obvious and is specific to each individual case.

The aspect ratio has a much larger influence in multi-dimensional flow due to differences in cell size across large gradients. Since, in this case the flow is certainly multi-dimensional, the impact of the aspect ratio could be large. If the flow is directly aligned with the longest side of a high aspect ratio cell, such as is often the case within a boundary layer, the difference in length could not have much influence on the accuracy. However, when there is a component of the flow and a gradient in the direction normal to the cell, this difference in cell resolution in the two directions can have an impact on the accuracy of the solution.

In a simulation in which the boundary layer is to be resolved, large aspect ratios can be tolerated, if they are in an acceptable region of the boundary layer. A threshold value for this is often determined through much experience of the effects that they have with respect to the regions of flow they are present in. Since setting a threshold for individual cases is not achievable within an optimisation procedure, also with the potential for high curvature, the threshold value is set on the safe side.

3.1.3.3 Included angle / non-orthogonality

Non-orthogonality is a similar metric to skewness in that, in a way, it is a measure of how distorted a cell is. It is calculated using the difference in angle between the vector that connects the two adjacent cell centres and the vector that is normal to the common faces of the adjacent cells, as seen in figure 2. The ideal angle, is therefore, 0° for a fully orthogonal cell, ranging up to 180° for a highly distorted cell. The vector connecting the adjacent cell centres represents the approximation of the gradient across the shared face; this

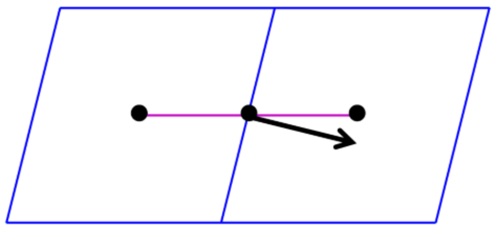


Figure 2 'Diagram showing the vectors from which the orthogonality is calculated'

can introduce significant errors when the dot product of the gradient with the face area is required, as the vectors are misaligned within adjacent non-orthogonal cells. Since the face normal vector is used within OpenFOAM, this makes it more sensitive to the non-orthogonality of cells and is an important quality metric.

This non-orthogonality can have a large impact on the numerics of the solution [10] [23], with values above $70°$ likely to significantly inhibit the convergence of a solution [31].

Unfortunately, Pointwise does not record the non-orthogonality of adjacent cells, however, one can report the minimum and maximum included angle of all the faces within a cell. Whilst the included angle does not directly represent the values that the gradients are approximated over, it can be said that a cell with an extreme minimum or maximum included angle will, in its nature, have a high non-orthogonality. Therefore, it is inferred *and is common practice* that the two metrics would have a similar effect on the numerics of the solution. The ideal value of included angle for a quadrilateral element would be, $\theta = 90°$, i.e. a perfectly orthogonal cell, with an acceptable range being $45° < \theta < 135°$ [32].

3.1.3.4 Area Ratio/smoothness

The area ratio of a two-dimensional domain is the ratio of the larger cell area to the smaller area. It is often quoted as the smoothness of the mesh, i.e. a smooth transition between the cell sizes. The volume ratio is the equivalent metric for three dimensional meshes. General rules of thumb for boundary layer expansion can be quoted as no higher than 2 for structured hexahedral meshes [33]. A large change in volume between cells is often associated with the degradation of convergence, however, the literature detailing the true impact is scarce. Similar to the aspect ratio metric, the range of ideal values is often based on the specifics of the flow and on the judgement of an experienced analyst. Often, the true impact is only determined when a sensitivity study is performed on that specific flow-field.

3.1.3.5 Flow alignment

Whilst not a direct measurement of the shape of the cells within a mesh, the alignment of the cells, specifically the alignment of face pairs of a hexahedral mesh with the predominant flow direction, can have a noticeable impact on the accuracy of the solution [8]. For a mesh with orthogonal cells that are perfectly aligned with a one-dimensional flow, only interpolation errors will be present, as the quantities at the cell faces can be accurately calculated from the cell centre values. The misalignment of the flow can then induce a non-negligible numerical diffusion error because of the discrepancies between the face and cell centre values.

3.1.4 Pointwise

Pointwise is a commercial mesh generation software in which all aspects of mesh pre-processing can be performed. It is advantageous as it is highly flexible, allowing many different CAD file formats to be imported directly into the software whilst also allowing the user to directly export a completed mesh into a variety of formats for different solvers, including OpenFOAM. It is known as one of the best, readily available meshing tools for structured hexahedral meshing [34], by utilising a *bottom-up* meshing procedure in which the distributions are first defined and then two-dimensional domains from which the volume grid is created. This allows for the greatest control over the mesh that is produced. It also has a range of the most reputable meshing algorithms available to the user, some of which are outlined in the literature review.

Glyph scripting is a recent development within Pointwise and is based on the programming language Tcl. It allows full scripting of all the processes available in the GUI, and allows the user to develop further capabilities than have been originally implemented, as shown by Grigoriev et al's [12] implementation of higher order meshes. It provides many commands specific to processes within Pointwise as well as the full capabilities of the Tcl language to perform other operations. This scripting process combined with one of the most capable meshing softwares available, makes it a very powerful tool, especially within an optimisation process.

3.1.4.1 Grid Generation Algorithms

Structured

Pointwise uses the transfinite interpolation method [35] in order to generate structured domains. As mentioned in the literature review, this method can be somewhat limited in terms of the mesh quality. However, Pointwise allows the use of elliptic partial differential equations to resolve these structured meshes; this method is shown to produce some of the best quality meshes of all the widely available algorithms [15].

Unstructured

The default setting for unstructured domains is the Delaunay algorithm, which populates a domain with somewhat irregularly shaped triangular cells. This algorithm has been shown to have slightly inferior results to the Advancing Front algorithm when considering skewness, as shown by Rhoads [8]. There is also the possibility to select the desired cell types; a mesh purely populated with triangular cells or with an unstructured combination of triangles and quads.

3.1.4.2 Structured Grid Smoothing Algorithms

The smoothing and improvement of the grid quality is done by calculating the solution to Poisson's elliptic partial differential equation (PDE); it does this with respect to the various control functions that can be imposed.

These original governing equations were proposed by Thompson et al [36] and forcing terms have then been added to these equations by various authors [37] [38] to control certain aspects of the mesh. Each implementation uses a slightly different method of determining these forcing terms, thus having varying effects on the final mesh distribution.

It contains separate control functions for the interior and boundary of the grid, that determine the treatment of the grid nodes, in order to provide complete control. It also provides control for the constraints of the surfaces on which grid points are able to move, as well as functions controlling the blending of edge into interior functions.

Solution Algorithm

The solution of the grid point positions is found using a successive over-relaxation (SOR) numerical algorithm with or without a multi grid acceleration. It is an iterative process that will converge upon the mesh point locations that satisfy the elliptic PDEs.

Successive over-relaxation: this is a method of solving any linear system of equations computationally. It uses an explicit method and is a variant of the Gauss-Seidel method, resulting in faster convergence of a typically slowly converged solution.

MultiGrid: This method is used in conjunction with SOR and creates successively coarser representations of the original domain. These representations are then solved using the PDEs and the changes to coarser grids are propagated through to the finer meshes, allowing faster convergence of the solution.

Interior Control Functions

These functions control the interior distribution of the grid points within each iteration of the elliptic PDE solver. There are three separate methods available within the Pointwise framework.

Laplacian: For each vertex of the mesh, its new position is based upon the location of its neighbours in order to try and produce the smoothest possible mesh. For this reason, the

Laplacian smoothing scheme mainly focuses on providing the smoothest possible distribution of grid points, this is at the expense of the orthogonality of the mesh and the previously defined clustering or spacing of grid points.

Thomas-Middlecoff: conversely this method focuses on smoothing the interior points with respect to the previously defined distribution of the mesh points at the boundary. The resulting interior distribution is controlled entirely by the boundary distribution, with the free parameters of the elliptic equations being calculated from the Dirichlet boundary values [39]. It is known to be very stable and robust and is the default setting [35].

Fixed Grid: This method is used only to eliminate slope discontinuities in the mesh whilst maintaining the distribution of points in the rest of the grid as best as possible. It therefore, does not smooth the mesh as such and has little influence on the overall mesh quality.

Boundary Control Functions

These forcing functions influence the distribution of the grid points near the boundary and adjust the grid points local to the boundary in order to specify constraints on the first layer of the mesh by adding forcing term coefficients to the elliptic equations. Within the Pointwise framework the desired angle and spacing constraints can be set, these provide the guidelines from which the PDEs can then be solved. The points are then propagated smoothly into the interior mesh. These forcing terms can often be quite large which can cause instability, as some of the terms are solved by central differencing rather than SOR [37]

Von Lavante-Hilgenstock-White: This function formulated by, uses strict adherence to the specified constraints, specifically the angle and spacing controls [38], this is at the slight detriment to the smoothness of the grid points.

Steger-Sorenson: As opposed to the above formulation, this algorithm uses a formulation of the forcing terms that are more approximate of the constraints and can therefore put more of an emphasis on the smoothness and orthogonality of the mesh. [37]

None: There is the option to set no conditions for control of the boundary mesh, allowing all grid points to be formulated from the interior function algorithm.

Distribution boundary conditions

These boundary conditions control the ability of the above algorithms to alter the distribution of the boundary points, and can have a large impact on the quality of the mesh.

Fixed: This setting maintains the previously set distribution of the points along the boundary, not allowing the smoothing algorithms to move the points at all.

Orthogonal: The distribution of the boundary points is allowed to move freely along the fixed exterior boundary in order to maintain orthogonality with other interior cells.

Floating: This allows the algorithms to have full control over the movement of the boundary points, treating them as if they were points on the interior grid.

There are also settings available to allow control of the blending of the boundary to the interior meshing functions, as well as the spacing control of the boundary points with respect to adjacent grids or otherwise.

3.1.5 Python

Python is an open source, object-oriented programming language and has a range of libraries available to the user. One such library is an OpenFOAM wrapper; pyFoam, which allows all the various OF commands to be run from within Python and is essentially a scripting facility for the OpenFOAM code. It allows all features and commands available in OpenFOAM to be automated, such as scaling, running solvers and analysing log files. Another useful library is Numpy, which is a fundamental package for manipulating data and scientific computing and is most useful for array manipulation and automated post-processing. There are also many libraries that are instrumental in the implementation of an optimiser, thus making the language of choice for this project.

3.1.6 OpenFOAM

OpenFOAM is an open source numerical flow solver written in C++. It has a range of solvers for various flow types as well as utilities that can perform post and pre-processing tasks. It is a cell centred code that is known to have a bias to accuracy over robustness and is more sensitive to the flow settings than other commercial codes. It also lends itself well to being automated within a system and is the solver of choice for this project.

3.1.7 Catmull-Clark

The Catmull-Clark algorithm method is an algorithm that creates sub-division curves and surfaces from a set of vertices. It is an iterative procedure for which smooth curves can be between a number of positional control points. It will be used within this project as a method of representing and altering the shape.

3.2 Methodology

Throughout the project the main focus was to, at least, be able to mesh splines consisting of three control points. This was decided as it would provide a large enough design space to find a reasonable improvement, as well as not be too computationally expensive and be of achievable in terms of meshing.

3.2.1 Heel Mesh

During the optimisation of the draft tube, the geometry of the heel section was kept constant. Therefore, it would be rather unnecessarily complex and expensive to re-mesh the heel section for each evaluation, especially since the main aim was to create a framework for the optimisation process. The heel mesh was therefore created manually to maintain maximum control over the mesh. This was also beneficial as it allowed Burns [40] to use blockMesh to generate meshes for the diffuser section and stitch them to the heel grid. It was deemed achievable to create a structured mesh in this section, as although the geometry is not excessively complex, it contains a few features that can be challenging to produce a structured mesh for. As explained previously the advantages offered by a structured mesh, especially within an optimiser, were decided to be worth the extra effort in terms of time. First, the circular inlet was to be meshed. It is commonly known that one of the best ways to mesh a circular inlet is through an O-H topology mesh. It is also one of the only ways in which an internal structured grid can be mapped from a circular to a rectangular cross section. An H-grid is a typical structured grid and is used for rectangular sections containing four separate edges, wherein each boundary grid point is connected to a similar point on the opposite boundary of the geometry, whereas an O-grid is a circular grid in which each grid line is connected from the outer boundary to a boundary in the centre of the domain. Figure 4a shows a typical H grid applied to a circular cross section, as can be seen the mesh quality is poor at the points where the four edges that the circle has been divided into, meet. Figure 3b shows an O-H grid in which the O mesh propagates from the outer boundary to an inner boundary of the interior H-grid, thus allowing the grid quality to remain high, showing limited skew or non-orthogonality.

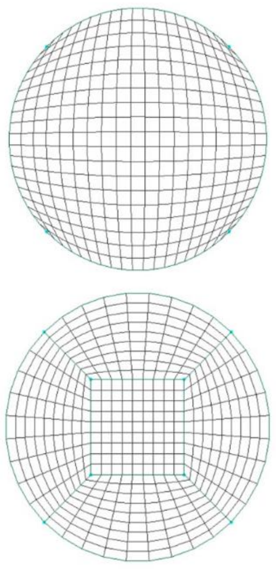


Figure 3 a) 'H-grid', top b)'O-H grid',

To begin with, the O-H topology was generated at the inlet domain, this was then mapped down the inlet cone of the draft tube as can be seen in figure 4a. The O-H domain was then attempted to be mapped through to the end of the heel section of the draft tube, however, due to the sharp heel corner, the stretching and distorting of the mesh to accommodate this corner was too great to achieve an acceptable level of mesh quality. Since the floor of the heel section is essentially an elongated semi-circle, similar problems were seen when trying to apply an H-grid to that surface domain as are seen in figure 3a. This H-grid along the surface domain is, however, required in order to map the O-H topology throughout the domain, making it a rather tricky

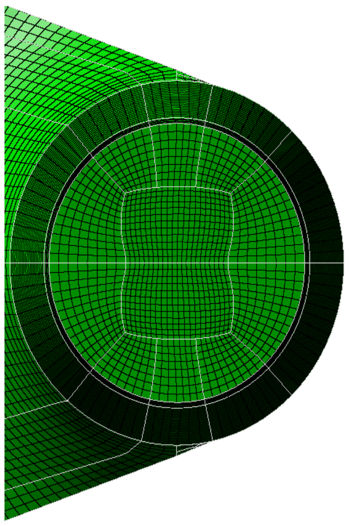
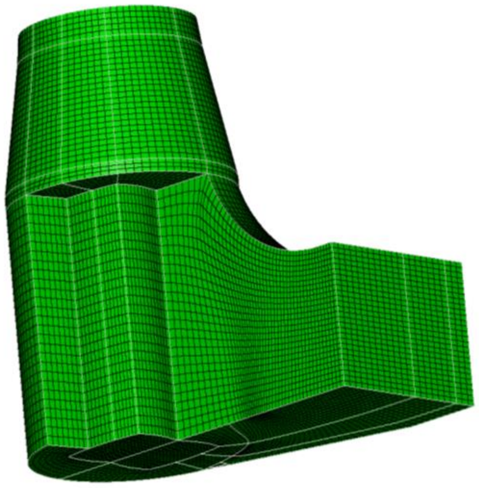



Figure 4: a) 'Top view of O-H mesh' b) 'Extrusion of O-H to heel floor'

geometry to mesh in a structured manner. It is worth noting that if the draft tube had a curved heel, as is logical and has been shown to be superior in terms of performance [5], mapping on O-H mesh through the domain would have been a relatively straightforward task. Since one of the only ways to mesh a circular section is with an O-H topology, several of the inlet cone sections were extruded downwards onto the floor of the tube in order to accommodate the semi-circular geometry, as is shown in figure 4b.

Due to the way in which the CAD model is built, a tangential curve joins the inlet cone to the diffuser section. This had to be adjusted slightly to allow a structured mesh of good quality to be applied to the surface of the geometry. The change was small enough that it would have a negligible effect on the solution in terms of the geometry but could have a large impact by allowing for higher quality cells to be produced in this region.

This mesh topology possibly shows better flow alignment than a fully mapped O-H topology might, as the flow mostly transfers down through the inlet cone and hits into the heel floor, rather than moving smoothly through the tube. In addition, the rectangular cross section of the tube can be meshed with a simple H-grid, allowing simpler and potentially higher quality meshing of the diffuser section.

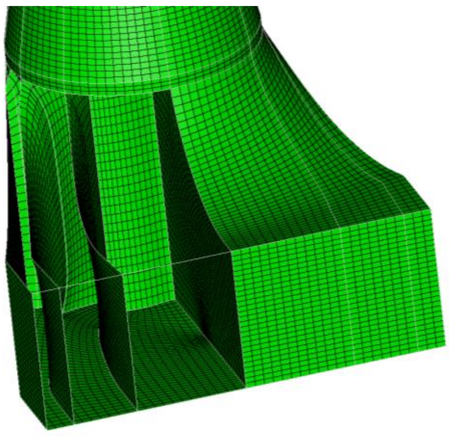
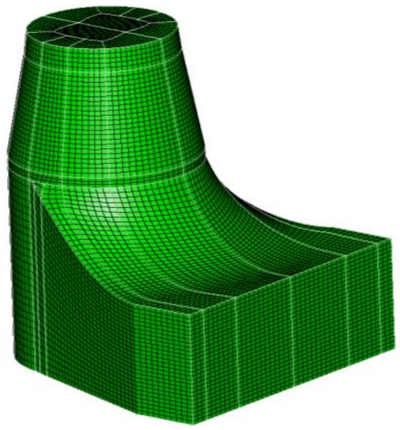



Figure 5: a) 'Showing domains to maintain grid control', left b) 'Final mesh', right

Domains were then created, connecting from the divisions of the remaining sections of the O-H domain to the rectangular section, to guide the mesh through, ensuring control over the distribution and quality of the resulting blocks. This is shown in figure 5a, with the final mesh shown in figure 5b.

The required first cell height to achieve $30 < y^+ < 300$, was estimated using equations based on flat plate boundary theory from [41]. The distribution of the grid points was subsequently based on this cell height. However, the boundary layer was not refined in comparison to the volume mesh as the first cell height was reasonably coarse and a larger cell size within the mesh would struggle to capture the geometry as well as much of the flow. This resulted in an underestimation of the y^+, due to changes in the flow giving a $y^+_{ave} \approx 20$. As can be seen from

Table 1 'Table of mesh quality metrics for the heel section

Metric	Extrema Values		Average	Threshold	Over threshold (%)	No. of cells
Equiangle Skewness	max	0.73	0.079	> 0.5	0.13	31265
Volume Ratio	max	2.59	1.13	> 1.5	25	
Aspect Ratio	max	5.95	1.6	> 2	56	
Min included angle	min	26°	77.7°	< 45°	1.6	
Max included angle	max	157°	102°	>135°	1.8	

the table 1, the average values for all the mesh quality metrics are highly encouraging. However, the extreme values are often seen to have a larger impact on the solution convergence and accuracy [10]. The maximum values for this mesh are not low, but they are mostly within the bounds of the threshold values that have been quoted in the theory section. The percentage values over the quoted thresholds for the skewness and included angles are very small and the extrema values are not high enough to cause concern. The values over the thresholds for the volume ratio and aspect ratio are higher. However, as mentioned in the theory section, these threshold values are in no way concrete and the effect on the solution is more based on the locations and types of flow and the effect cannot truly be told until it has been run.

When running the base draft tube geometry, using settings detailed by Hardy [42] in *3.6 Solution Setup*, with a $k - \varepsilon$ turbulence model, the solution does not explicitly converge. Although, the values of the pressure residuals fall below 1×10^{-3}, stabilising to a roughly flat line with some very small, repeated fluctuations often associated with a slightly transient flow. Although these residual errors are not as low as would be ideal, the pressure difference value between the inlet and outlet converges to 2 decimal places on a value that turned out to show very good agreement with the experimental results [43]. Some of the slightly lower quality cells within the mesh, specifically the non-orthogonality, may be contributing to the residual errors being slightly higher than ideal.

3.2.2 Geometry Import and generalisation

Since Pointwise is a so-called *bottom-up* meshing software, the best meshes are produced by beginning with the base meshing elements, which are the curves and lines that represent the geometry, on which the mesh distribution can be defined. Therefore, from the perspective of importing the geometry into the meshing software, it was decided that the geometry would be best imported in the format of a single curve or spline. Therefore, a methodology had to be created to transform a set of co-ordinates from an array into a file that could easily be imported into the mesh generation software. The chosen method was a segment file format, in which each segment of a geometry is represented by the coordinates, and on import into Pointwise, arc segments are fitted between the points between each appropriate section. The pseudo representation of the format is shown in figure 6.

A section of code was then written to transform the coordinates of the spline from an n-dimensional Numpy array to a file in the segment file format, this is detailed in Appendix A.

N_1
$x_n \quad y_n \quad z_n$
$x_{n+1} \quad y_{n+1} \quad z_{n+1}$
. . .
$x_{N1} \quad y_{N1} \quad z_{N1}$
N_2
$x_n \quad y_n \quad z_n$
$x_{n+1} \quad y_{n+1} \quad z_{n+1}$
. . .
$x_{N2} \quad y_{N2} \quad z_{N2}$
...
N_n
$x_n \quad y_n \quad z_n$
$x_{n+1} \quad y_{n+1} \quad z_{n+1}$
. . .
$x_{Nn} \quad y_{Nn} \quad z_{Nn}$

Figure 6 'Pseudo file format representation'

Where, N_1 is the number of points in segment 1
N_2 is the number of points in segment 2
N_n is the number of points in segment n
And x_n, y_n, z_n are the coordinates of the nth point of that segment

To perform an action on a particular object within Pointwise glyph scripting, the name of that object must be called. Each object is named by assigning a number and type of domain e.g. '*dom-18*', however, the numbering process does not appear to be fully sequential and sometimes appears to be random. This presents somewhat of a challenge to generalise the process for an automatic meshing procedure in which different geometries can be imported for each evaluation. Therefore, the spline was imported first and re-named to avoid any object naming issues. Once this was done the sections of the diffuser geometry that remained constant where imported and correctly scaled.

3.2.3 Diffuser Mesh Generation

The glyph script was written, such that the spline was then translated into the correct position on the geometry on import, and the distribution of mesh points was then applied to the spline and fixed boundaries. Since the TFI method of structured mesh generation can be unstable for concave regions of the boundary, which will regularly be generated during the geometric alterations, the side domain was split into equally distributed sections and the domains then created, as shown in figure 7. This solved some issues that could not be rectified by use of the elliptic solver. The four domains were then joined, as it was seen that solving the mesh with one domain, as opposed to four, showed better continuity and allowed the solver more freedom when adjusting the interior mesh distribution. Solving the elliptic PDEs for the two-dimensional domain proved far less computationally expensive as only 3807 cells had to be adjusted as opposed to the many more that are within the three-dimensional mesh. The three-dimensional volume grid is then created using a parabolic extrusion method in which the mesh

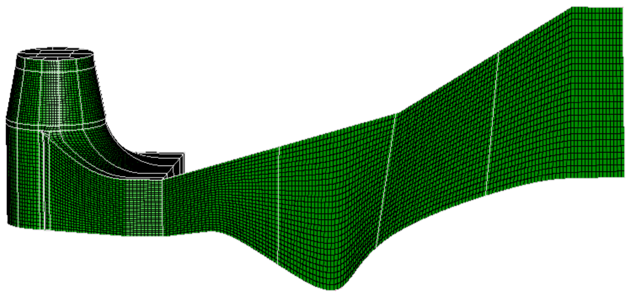


Figure 7 'Showing domains split across diffuser section'

quality of the two-dimensional domain is almost exactly reflected in its three-dimensional counterpart. This process is shown in figure 9. Within Pointwise, the boundary conditions can then be applied in accordance with various solvers, of which OpenFOAM is one; the patch types and names were applied as detailed by Hardy [42]. All in all, this meshing process, once fully automated takes $\approx 10\ seconds$ to import the geometry, apply scaling, generate and solve the mesh as well as applying boundary conditions and exporting the mesh to the desired folder in the correct, OpenFOAM format. The glyph script for this meshing procedure can be seen in Appendix B.

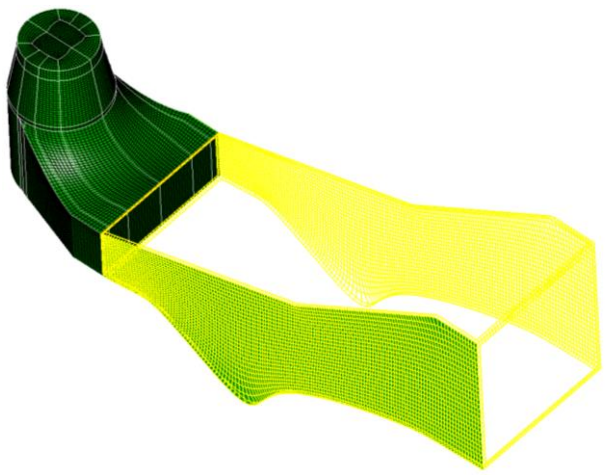


Figure 8 'Domain extrusion method'

3.2.4 Curvature

It was desired to monitor the curvature of the splines that were produced to determine what, if any, influence this has on the quality of the structured meshes that are produced. There are functions within some libraries of Python that can be used to find the gradient of a curve from its set of points. These however, rely on the points being equally spaced in the x direction. This is not the case when using Catmull-Clark splines, as the number of points increases in areas of high curvature. It was therefore necessary to write a function that would allow the gradient of any set of points to be calculated. This was done simply by calculating the change in y divided by change in x, using the differences between adjacent points. The implementation of this can be seen in Appendix C. This function relies on the points being sequentially ordered and cannot be applied to splines that go back on themselves. Splines of this nature were rarely seen and only for a number of control points greater than ≈ 7.

3.2.5 Automation and interfacing

Since Python is an object-oriented programming language, each of the meshing processes was written as an individual function that could be inserted into the optimiser code of [28]. These were written for both cfMesh and Pointwise. For cfMesh, the flow diagram representation of the function is shown below in figure 9a. This function required the control points to be inputted, from which an STL surface of the spline could be made, using a function written by [9]. This STL surface then had to be transformed from binary to ASCII format, so the boundary

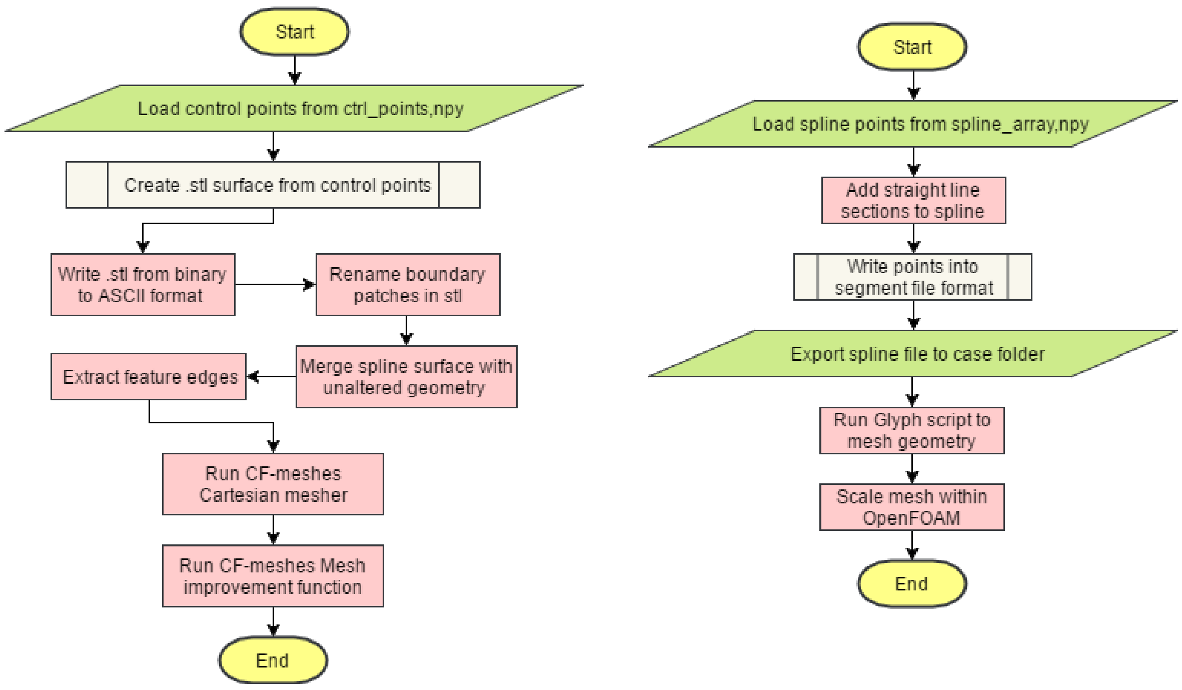

Start
Load control points from ctrl_points,npy
Create .stl surface from control points
Write .stl from binary to ASCII format
Rename boundary patches in stl
Merge spline surface with unaltered geometry
Extract feature edges
Run CF-meshes Cartesian mesher
Run CF-meshes Mesh improvement function
End
Start
Load spline points from spline_array,npy
Add straight line sections to spline
Write points into segment file format
Export spline file to case folder
Run Glyph script to mesh geometry
Scale mesh within OpenFOAM
End

Figure 9 'Flow diagrams showing simplified steps for Functions of a) cfMesh, left and b) Pointwise, right'

patches could be renamed and the STL merged with the other, constant sections of the geometry. The cfMesh application was then called using functions from the PyFoam library, and a mesh improvement function run. The code for this section can be found in Appendix D. Similarly, the Pointwise function required the input of the spline points, from which, extra sections of the geometry could be added in order to account for a possible change in size of the bounding box. This process, for cfMesh was implemented within the STL function. These points where then written into the Segment file format and the spline saved as a file within the case folder. This file was then used by the glyph script which is called from within the function; to create and mesh the geometry within Pointwise. The points were then scaled correctly, using the OpenFOAM application `transformPoints`. Once these functions had finished, a function was called to run the solution and then report the pressure drop. This implementation can be found in Appendix E.

The object-oriented fashion in which the overall optimiser code is written, helps to make the interfacing of the CFD with an optimiser simpler. Despite this, interfacing the meshing tools and CFD with the optimiser was one of the most time-consuming and challenging aspects of the project. Since the overall optimiser code sums to several thousand lines of code across 5 programme files, there was a significant amount of debugging to be done before the first evaluations could be performed. As both the meshing tools had different requirements as inputs as well as the use of the lengthy glyph scripts within the Pointwise function, this meant that each of the implementations were rather different. Transferring information across programme files and between sections of the optimiser and meshing tools presented somewhat of a challenge.

3.2.5.1 Recording Mesh Quality

As mentioned previously, the mesh quality has a large influence on the accuracy of the solution and therefore the performance of the optimiser. Therefore, it is important to develop a method of monitoring the mesh quality automatically within the optimiser, to prevent solutions becoming inaccurate. There are applications within OpenFOAM that can report the mesh quality, such as `checkMesh` and `checkMeshModified`, however, these are limited to a few metrics of mesh quality and would require programming in C++ in order to customise these to record the quality in the desired format. It was seen that Pointwise has a more comprehensive and accessible set of metrics and is highly regarded as one of the best methods for assessing mesh quality. Hence, a glyph script was developed, to use functionalities within Pointwise for determining the mesh quality.

The values of all the cells within the mesh can be recorded with Tcl commands for finding the maximum, minimum and average values as well as the number of values within certain ranges. A script was then written that reports the extreme values and the average values for each metric, as well as a function to record the values of the quality into a histogram format of bins and frequencies for more detailed analysis of the distribution of values. In order to transfer these values from the Glyph script to the Python optimiser, the values were exported as a file in the comma separated values format and a function written in Python to read in these mesh quality statistics as variables to be assessed. These values can then be used to penalise geometries with poor mesh quality and inhibit these solutions from misleading the optimiser.

3.2.6 Mesh Quality Thresholds

Determining threshold values of mesh quality, for which a solution is deemed inaccurate is somewhat open to interpretation. Without full analysis of the ranges and distributions of the various mesh quality, as would be done when producing a mesh manually, it is tricky to determine how the quality can affect the solution. Even with full analysis of the quality, as well as analysis of areas of high gradients within the flow, some preliminary simulations would usually be run to determine the true influence. As this is not a luxury that can be afforded within an automated optimiser, thresholds must be decided with care.

It was decided that the extreme values of the mesh quality would be used as the thresholds, as even one highly non-orthogonal or skewed cell can be sufficient to destroy convergence and therefore the accuracy of the solution [10]. Whilst the average values can be used to describe the mesh quality, they could be misleading and the extreme values were deemed to be more appropriate.

For the skewness, the threshold value was set to 0.65, based on the recommended literature values [44] and the fact that OpenFOAM is known to be slightly more sensitive to the effects of highly skewed cells [8]. The maximum and minimum included angles were set to be between $40° < \theta < 150°$ [32], equating to a rough non-orthogonality of 50°. The threshold value was for aspect ratio was set to 10. Although high aspect ratio cells in acceptable orientation and in acceptable regions of flow can be much higher than this [13], since this information was not available, the threshold was kept low. The maximum area ratio was also set to be 5.

There is a school of thought that says, CFD simulations within optimisers do not need to be 100% accurate, and higher levels of residual error are acceptable. As long as the flow is still physical and the trends are similar to that of the real life case, the optimiser will still move in the same direction and the same geometry will still be considered optimal. However, this has not been considered within this project and mesh quality values are set to levels that allow accurate simulation.

3.2.7 Elliptic solver settings

A brief study into the various functions available to solve the elliptic PDEs was performed to improve the grid quality. In order to find the most appropriate combination of functions for the optimiser, the performance of the solver would ideally be measured over a range of geometries, as some functions are known to perform best for differing geometries; some being better for high curvature and some best for convex regions. Therefore, to avoid finding settings that were overly suited to one particular test geometry, a code was written to repeatedly mesh a random

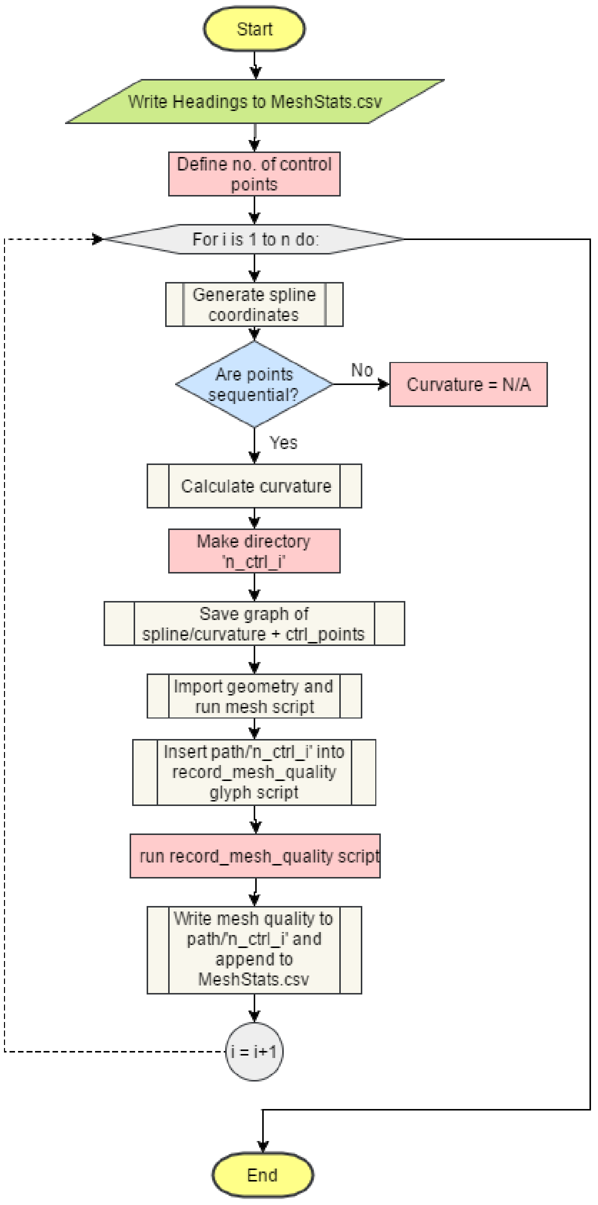

Start
Write Headings to MeshStats.csv
Define no. of control points
For i is 1 to n do:
Generate spline coordinates
Are points sequential?
No
Curvature = N/A
Yes
Calculate curvature
Make directory 'n_ctrl_i'
Save graph of spline/curvature + ctrl_points
Import geometry and run mesh script
Insert path/'n_ctrl_i' into record_mesh_quality glyph script
run record_mesh_quality script
Write mesh quality to path/'n_ctrl_i' and append to MeshStats.csv
i = i+1
End

Figure 10 'A Flow diagram representation of the script for examining mesh quality over many splines '

distribution of splines and report the mesh quality and various other aspects. In order to gain a full insight into the problems experienced for certain geometries an individual folder was created for each spline that was meshed, including all the mesh quality values, a plot of the spline and it's curvature as well as the control points for the spline, so the mesh could be recreated on demand. As dealing with two scripts, one calling the other, in separate languages, the path names in the glyph script had to be altered for every spline to save the data to separate folders each time. One main file was also appended to, collecting the mesh quality for all the geometries, from which analysis could be performed. The flow chart for this script can be seen in figure 10. This section of code is seen in Appendix F.

4 Presentation of analytical results and description of final constructed product

4.1 Results

4.1.1 Elliptic Solver Study

In the first section of results, the quality values are reported as averages over a random sample of 100 splines, and for the elliptic solver, all the settings are run for 40 iterations as this was

Table 2 'Comparison of mesh quality for interior control functions'

	Skew		Aspect Ratio		Area Ratio		Max Include Angle		Min Included Angle	
	Average	Max	Average	Max	Average	Max	Average	Max	Average	Min
Thom-Middle	0.21	0.58	1.72	4.80	1.03	1.39	108.4	142.1	71.7	39.2
Laplace	0.20	0.60	1.69	5.60	1.03	1.42	107.8	143.7	72.3	37.7

Table 3 'Comparison of mesh quality for boundary fixing functions'

	Skew		Aspect Ratio		Area Ratio		Max Include Angle		Min Included Angle	
	Average	Max	Average	Max	Average	Max	Average	Max	Average	Min
von-Lav-Hilg-White	0.20	0.61	1.72	17.30	1.04	3.00	108.1	144.9	72.0	36.4
Steger Sorenson	0.21	0.58	1.72	4.80	1.03	1.39	108.4	142.1	71.7	39.2

deemed appropriate from initial studies.

As can be seen when comparing the values of mesh quality in table 2, it seems that the Laplace interior function results in slightly superior values when looking at the averages of each metric. Whereas, the Thomas-Middlecoff function shows higher quality values for the extrema. Therefore, there is a slight trade-off between these two metrics, however, the differences between the values are almost negligible and it is highly unlikely that any difference in solution accuracy would be seen, when considering the values in table 3. Therefore, given the fact that the Laplace function gives little regard to the original distribution of grid points, the TM function is best suited to this scenario.

Table 3 shows the comparison between the two boundary fixing functions, that of Steger and Sorenson, and of Von-Lavante, Hilgenstock and White. The function of Steger and Sorenson is shown to be the more stable of the two, showing far better extreme values, especially for the aspect ratio and area ratio. When meshes are recreated, the LHW function performs well but is shown to be highly sensitive to the number of iterations. The quality appears to be good initially but quickly diverges for geometries with highly concave regions. But for an ideal number of iterations, which varies with the complexity of the geometry, the LHW function appears to produce higher quality elements. With poor mesh quality from these diverged solutions skewing the measures of extreme values, it would therefore seem that the trade-off between the two is in stability. Stability and robustness are highly advantageous qualities within an automatic mesher; for this reason, coupled with the frequency at which highly concave geometries can be produced within the optimiser, the function of Steger Sorenson will be used.

The method for solving the elliptic PDEs is compared in table 4, as can be seen, the SOR method shows very similar performance to the multigrid method when analysing the average

Table 4 'Comparison of mesh quality for solver method'

	Skew		Aspect Ratio		Area Ratio		Max Include Angle		Min Included Angle	
	Average	Max	Average	Max	Average	Max	Average	Max	Average	Min
SOR	0.20	0.64	1.65	5.68	1.03	1.78	108.0	147.5	72.1	33.6
MultiGrid	0.21	0.58	1.72	4.80	1.03	1.39	108.4	142.1	71.7	39.2

Table 5 'Comparison of mesh quality for cell boundary conditions'

	Skew		Aspect Ratio		Area Ratio		Max Include Angle		Min Included Angle	
	Average	Max	Average	Max	Average	Max	Average	Max	Average	Min
Fixed	0.21	0.58	1.72	4.80	1.03	1.39	108.4	142.1	71.7	39.2
Floating	0.20	0.57	1.66	13.13	1.03	3.32	108.1	141.3	72.0	40.0
Orthogonal	0.14	0.42	1.53	2.45	1.03	3.12	102.8	126.0	77.3	53.0

values. However, for the extreme values, the Multigrid method shows significantly better performance, across all quality metrics. Hence, it is an obvious choice and the MultiGrid method will be used within the optimiser.

Table 5 and figure 12 highlight the mesh quality differences between the boundary conditions applied to the first layer of elements. As can be seen, the average values for the fixed and floating boundary functions are very similar. The same can be said for the extreme values of skew and included angle values as well as the distribution of the skewness. However, the floating boundary condition shows higher extreme values for the aspect and area ratio. Floating grid points can be converted into fixed points if a unique neighbour is not found for each grid point; this is the reason for the similarity between the two results. The orthogonal condition shows a noticeably better distribution of skewness with far more cells within the lower bins.

The performance for the included angle is also distinctly better, this is unsurprising since this metric represents the non-orthogonality and this will therefore be the selected boundary function. It does however, perform slightly worse than the fixed boundary condition, when considering the area ratio. This is unsurprising as the fixed condition does not alter the boundary grid distribution which is originally set.

Table 4 'Comparison of mesh quality with no. iterations'

No of iterations	Skew		Aspect Ratio		Area Ratio		Max Include Angle		Min Included Angle	
	Average	Max	Average	Max	Average	Max	Average	Max	Average	Min
20	0.17	0.48	1.58	2.54	1.03	2.69	104.9	132.8	75.2	47.7
40	0.14	0.43	1.56	2.59	1.03	3.13	102.8	128.1	77.2	51.7
60	0.12	0.41	1.52	2.57	1.03	3.16	101.0	125.6	79.1	53.5
80	0.11	0.44	1.49	2.77	1.04	3.67	100.1	128.4	80.0	51.0
100	0.10	0.43	1.46	2.56	1.04	3.37	98.5	128.2	81.5	51.5

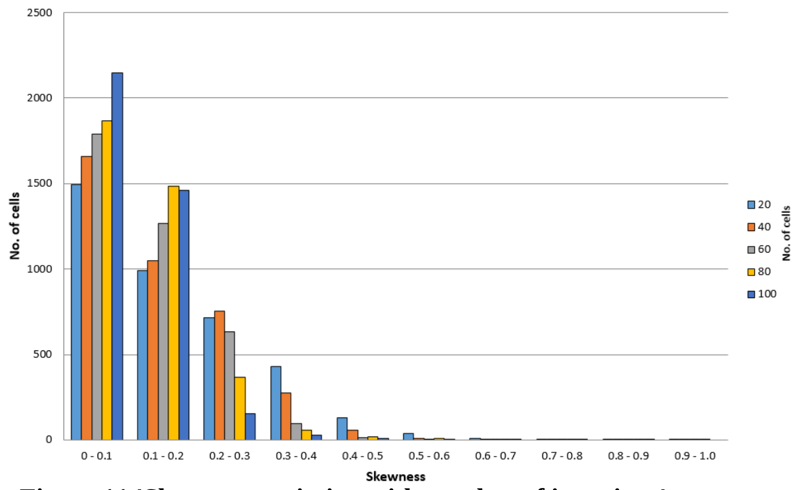


Figure 11 'Skewness variation with number of iterations'

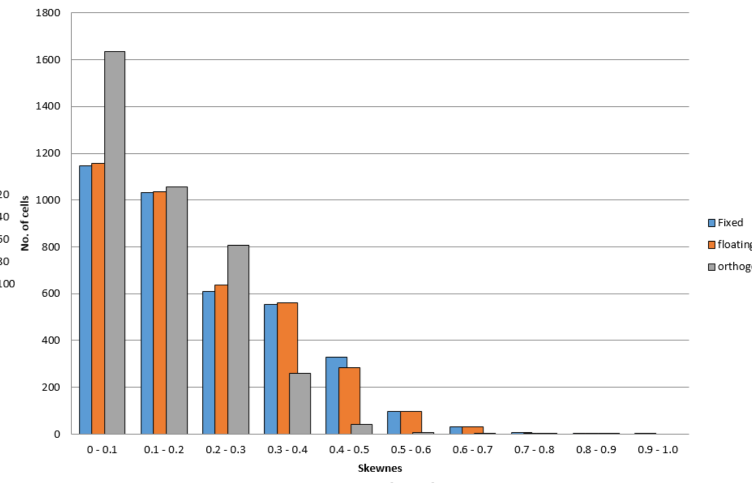


Figure 12 'Skewness variation for first cell boundary condition'

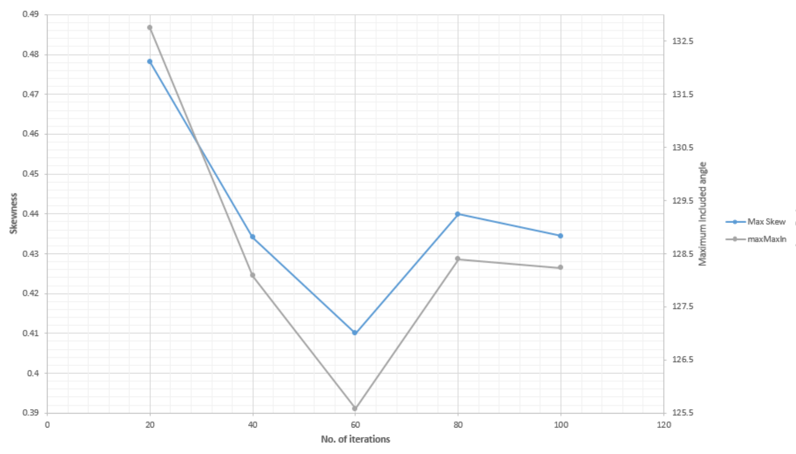


Figure 13 'Skewness and maximum included angle with no of iterations'

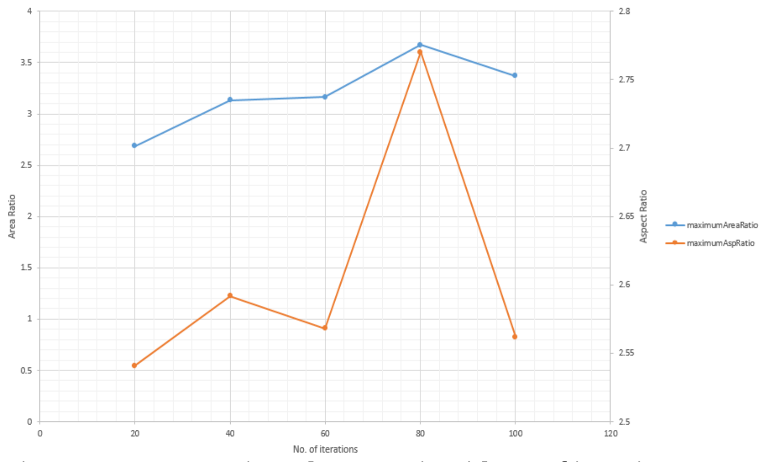


Figure 14 'Aspect ratio and Area ratio with no. of iterations'

The best settings from each of the previous sub-studies have been combined, and the number of iterations altered to determine the relationship between the number of iterations and mesh quality over many splines. The results are shown in table 6. As can be seen in figure 11, the skewness distribution appears to move further into the lower bins as the number of iterations increases, thus increasing the quality of the grid with more iterations when looking at the averages over many splines. However, when looking at the extreme values of skew and maximum included angle in figure 13, the relationship is not linear; the extrema values decline until 60 iterations, then start to increase. This is because the grid can be ‘over-solved’; if a grid of high quality is solved for too many iterations, the grid quality can diminish. This process is difficult to capture over many geometries, as the initial grid quality is variable. In addition, for complex, highly concave splines the grid solution can diverge, in terms of quality. When looking at the figure 14, effects of increasing the number of iterations tends to decrease the quality of the grid with respect to the aspect ratio and area ratio. This is because the equations alter the distribution of points to increase the quality with more bias to the cell shape. The optimiser was run with 60 iterations of the PDE solver, to provide the best overall performance.

This study, however, is very brief and does not consider that certain combinations of settings may work better together, for example the vLHW boundary control function may be more stable when used with the SOR solver method. In order to find the optimum combination of settings, a GA could be run as has been done previously for settings of a meshing tool [10]. This would allow the most promising combinations to be tweaked to get the ideal settings, and could greatly improve the overall grid quality.

4.1.2 Curvature Study

In this section the relationships between the curvature and the quality of the mesh can be analysed, to understand the influence it has on the mesh quality. There may be potential for the curvature to be used to predict the quality or possibly alter certain parameters that may work better for higher curvature geometries. As can be seen in figure 15, there is a clear relationship between the curvature of the spline and the skewness, in this case the average curvature and

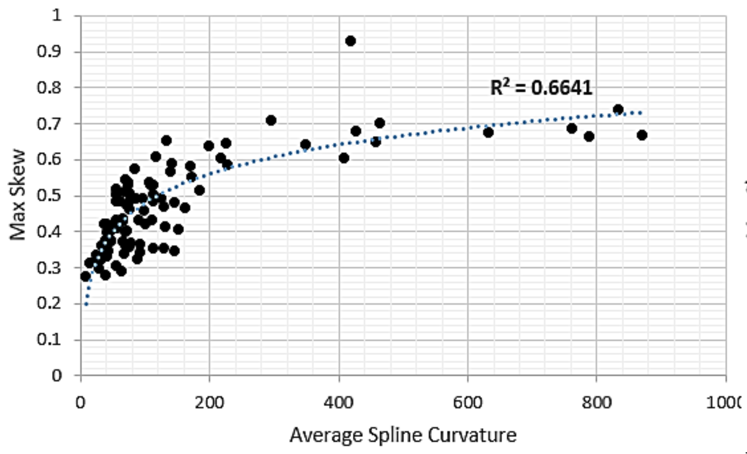


Figure 15 'Curvature against maximum skew for 20 iterations of the solver'

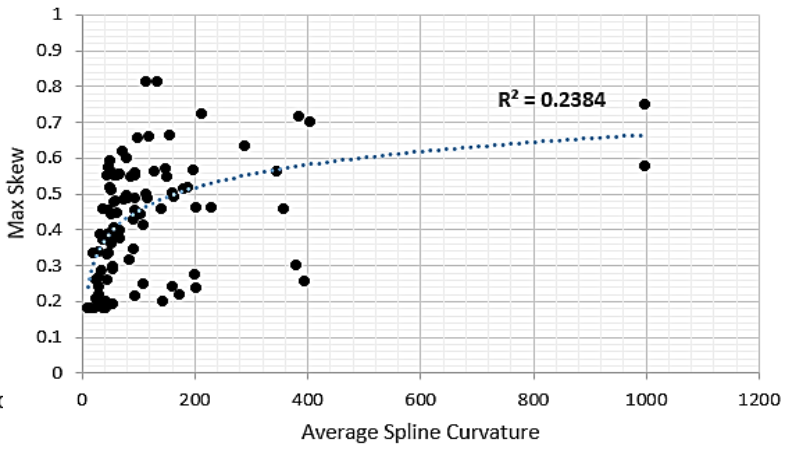


Figure 16 'Curvature against maximum skew for 100 iterations of the solver'

the maximum skew are displayed. The distribution is well fitted with a simple logarithmic model, with the initial stages being almost linear and then plateauing out as the curvature continues to increase, showing an R^2 of 0.66. All geometries with high-curvature exhibit poor mesh quality but not all geometries with low curvature have good quality, implying that it has a large influence but is not the only factor. The average curvature could therefore provide a rough estimate of the initial grid quality, allowing for potential customising the initial solver settings, dependent on this. As can be seen in figure 16 the number of iterations has been increased; the logarithmic function shows much worse regression with the distribution, the grid qualities dependence on curvature has decreased. Thus, highlighting the improvements the solver. Although in some cases the quality appears to have deteriorated for low curvature splines, indicating the presence of over solving on an already reasonable quality grid. It is believed that the trend in high curvature grids of diverging and over-solving, would be shown far more distinctly if the solver was run for many more iterations. This therefore shows that solving each mesh for a fixed number of iterations across all geometries is not ideal.

4.1.3 Potential Solutions dependant on findings

Some settings of the solver are believed to perform better for high curvature geometries, such as the Steger Sorenson boundary control. Therefore, after further investigation, a possible function could be written, such that the control functions may be selected accordingly, based on the curvature of the geometry. For example, the LHW method could be used for low curvature geometries as it has the potential to solve for the best quality grids but for scenarios where the curvature is high, the SS method could be used, as it is shown to be more stable for these geometries.

It has been seen but not shown in the above investigation, and is possibly the reason the correlation between curvature and quality is not better, that the nature of the spline, if it is convex or concave influences the quality also. Therefore, using the differential function defined in the methodology section, the double differential of the curvature could be found to determine if points of high curvature are maxima or minima and therefore if they are concave or convex regions. Certain settings could then also be customised for each individual geometry, as suggested above

4.2 *Final Implementation*

4.2.1 Final Results of the Structured Meshing tool

Figure 17 shows the distribution of the maximum skewness values for 100 different geometries sampled using the Latin Hypercube method for three control point splines. As can be seen, the majority of the values are below 0.6, therefore indicating that most of the grids produced are of acceptable quality in terms of skewness. The values of other metrics are shown in table 7 and are very encouraging, with very low values of average skewness and much improved values for the included angles. The values for the aspect ratio are seen to be good, with both the averages being very good and the maxima being acceptable. This highlights the ability of the mesher to produce quality meshes for 3 control point splines. That being said there are a few geometries for which it is not robust enough to produce acceptable values of skewness for. These geometries would be penalised when run within the optimiser and not allowed to influence the surrogate model.

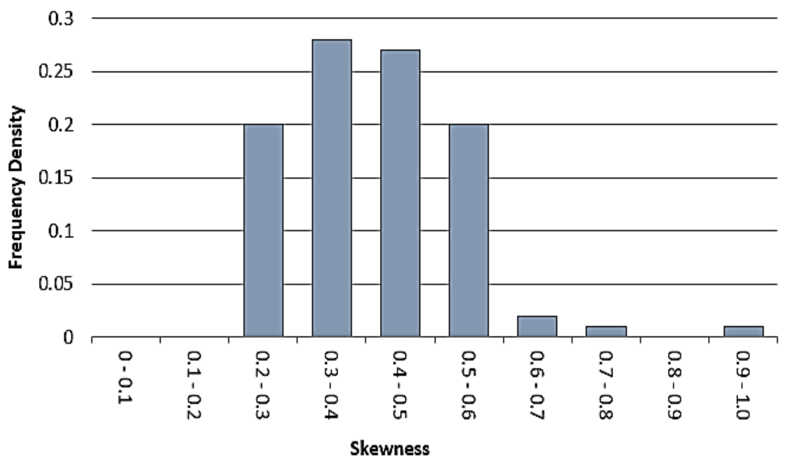


Figure 17 'Distribution of the maximum skewness values across 100 splines of 3 control points'

Table 5 'Table of mesh quality metrics over 100 splines for varying number of control points'

No. of ctrl points	Skew		Aspect Ratio		Area Ratio		Max Include Angle		Min Included Angle	
	Average	Max	Average	Max	Average	Max	Average	Max	Average	Min
1	0.11	0.30	1.43	2.15	1.02	2.22	99.6	116.3	80.5	64.0
2	0.11	0.36	1.49	2.24	1.03	2.72	100.2	122.1	79.9	57.6
3	0.12	0.41	1.52	2.57	1.03	3.16	101.0	125.6	79.1	53.5
4	0.13	0.49	1.53	2.83	1.04	3.82	101.9	131.9	78.2	47.4
5	0.15	0.60	1.58	3.85	1.05	5.34	102.9	143.2	77.2	38.5
6	0.16	0.70	1.60	4.48	1.05	6.66	104.3	151.3	75.9	30.3

As can be seen from table 7, across all metrics, the mesh quality decreases as the number of control points increases. The higher the number of control points the greater the chances of high-curvature geometries being produced, due to the large number of possible geometries available. More control points within a small design space means smooth geometries are less likely, when randomly sampled. For lower numbers of control points, the structured meshing tool is able to produce excellent values for both the average and extrema. For control points 3 and 4 the extrema are still within the quality thresholds defined in the methodology sections, with these values being pushed up slightly by outliers. This outlines the ability of the tool to create meshes for a range of reasonably complex geometries. For 6 and 7 control point geometries, the extrema values move much closer to the threshold limits, with a not insignificant number of meshes being of unacceptable quality. This is especially the case when looking at the values for non-orthogonality, this metric appears to suffer the most as the number of control points increases. Therefore, for this size bounding box, it would be infeasible to run an optimiser for any more than four control points when using this structured meshing tool. However, it is unlikely that large improvements would be seen for these numbers of control points, as smoothly varying pressure gradients would be expected for an optimum design.

4.2.2 Mesh Quality Control

The structured technique used to mesh the diffuser of the draft tube has been shown to produce high quality cells when used with an elliptic solver, it has also been shown to be reasonably robust, for a range of geometries. However, for some more extreme geometries the mesher fails to produce cells of acceptable quality, that if were run would produce inaccurate results and possibly diverge. As mentioned in the theory section, these inaccurate results could mislead the optimiser and cause it to converge on a false optimum. Hence, using the techniques for recording the mesh quality using Pointwise the values of mesh quality can be monitored from within the optimiser and those values that are shown to be over the thresholds will be discarded and penalised. This avoids these results being incorrectly considered by the optimiser.

This does however, limit the design space somewhat, as some geometries within the search space will not be tested due to poor mesh quality. In order to counter this, the work of this author and the use of the automated meshing tool cfMesh by Hardy [42] have been combined. cfMesh is an open source, automated, cut-cell meshing tool that is designed to be highly robust and be able to produce meshes for almost any geometries. This therefore, is an ideal complement to the Pointwise structured meshing tool, that has been shown to produce higher quality meshes but is less robust. This allows most of the

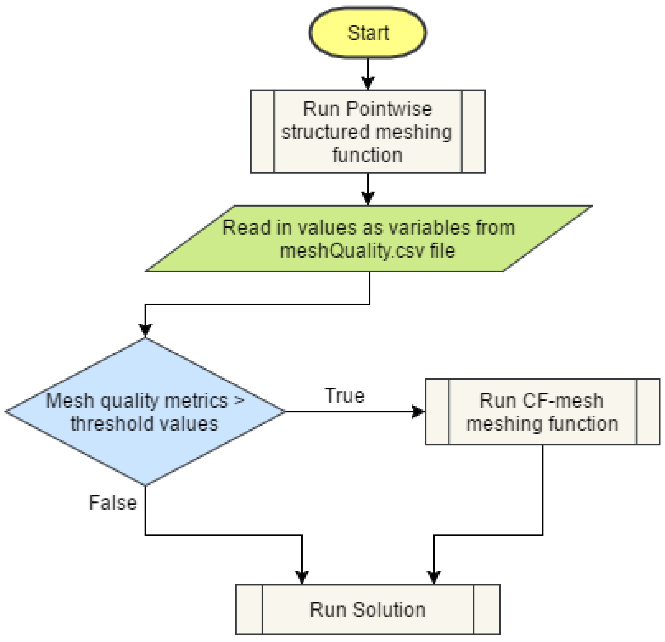

Start
Run Pointwise structured meshing function
Read in values as variables from meshQuality.csv file
Mesh quality metrics > threshold values
True
Run CF-mesh meshing function
False
Run Solution

Figure 18 'Implementation of CF mesh with Pointwise structured mesh.

geometries to be run using a good quality structured meshes, but for outliers that contain high curvature and poor mesh quality, cfMesh can be used. This allows all geometries queried by the optimiser to be meshed and a solution provided. There is, however, not currently a method for recording the mesh quality of CF mesh within this project, therefore there is no way to inhibit solutions with poor mesh quality if they are produced by cfMesh.

The implementation is shown in figure 18 by means of a flowchart. As can be seen, the geometry is attempted to be meshed using the structured meshing technique first as this has the potential to produce higher quality cells and is less computationally expensive as it contains less elements. If the mesh quality is below the thresholds, the solution is run on the Pointwise mesh. However, if the mesh is not of good enough quality, the geometry is passed to cfMesh to create the mesh, overwriting the previously produced structured Pointwise mesh.

A similar process was also developed that included an unstructured tetrahedral mesh, produced in Pointwise, using a glyph script similar to that defined in the methodology section. Unstructured meshes are known to be far easier to produce and can be applied to complex geometries more robustly. However, tetrahedral meshes have more elements than that of the complete structured Pointwise mesh and the hex dominant cfMesh. The advantage of using Pointwise as the more robust tool is that more control is provided over the distribution of cells and the ability to record the mesh quality. The simplified flow diagram of the code written to implement this is shown in figure 19.

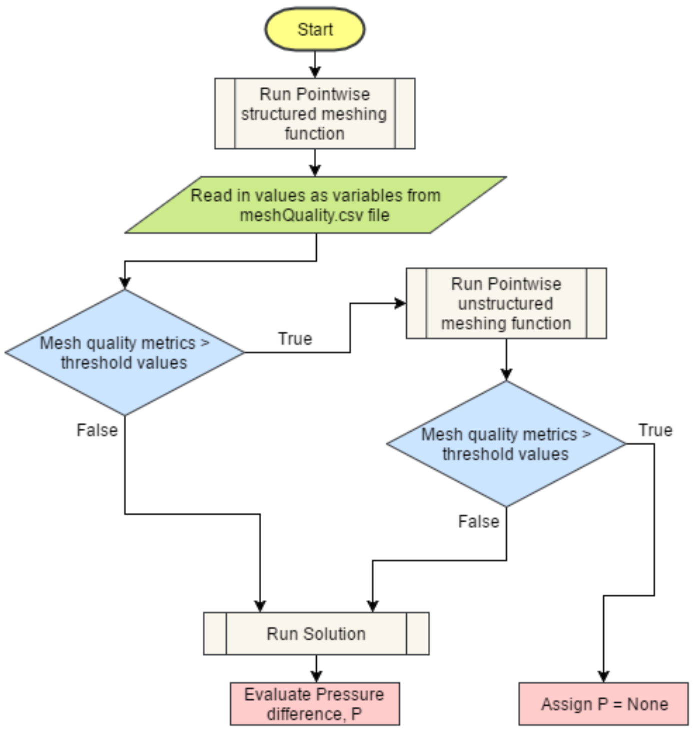
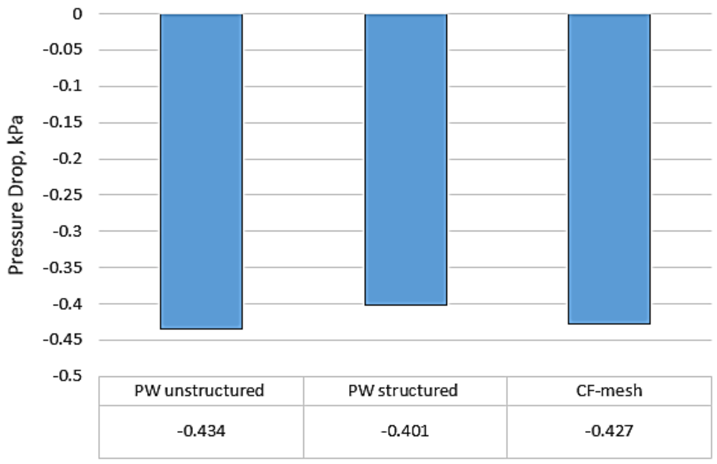

Start
Run Pointwise structured meshing function
Read in values as variables from meshQuality.csv file
Mesh quality metrics > threshold values
True
False
Run Pointwise unstructured meshing function
Mesh quality metrics > threshold values
True
False
Run Solution
Evaluate Pressure difference, P
Assign P = None

Figure 19 'Implementation of Pointwise structured and unstructured tools'

As can be seen, again, the structured mesh is first attempted and the mesh quality is assessed by the threshold values. If it fails the mesh quality test, the meshing is passed to the Pointwise unstructured meshing tool which produces a mesh. This must then be within the mesh quality thresholds before it can be passed to OpenFOAM to run the case. If the structured mesh does not produce a mesh with adequate quality, the geometry is penalised by assigning a none type object to the pressure difference.

4.2.3 Sensitivity of combined meshing tool

The method involving the combination of two meshing tools must be used with care. If there are discrepancies in the results due to the difference between the meshes produced by each tool, this may lead to inaccuracies in the optimiser. If one mesh consistently produces meshes that give a larger pressure difference, this will cause the optimiser to give a slight preference to geometries produced by one tool. This could lead to a non-optimal solution being produced. As can be seen in figure 20, the solution, when run using the $k - \varepsilon$ turbulence, is sensitive to the meshing tool. These discrepancies between the results are most likely due to differences in the y^+ values as well as mesh quality, and the settings for each tool must be tweaked to reduce this. This combination was implemented mainly to test the framework. For this reason along with the fact that appropriate experimental results have not been available for much of the project, work to reduce the sensitivity has not yet been done. With adjustments, the cfMesh tool is most likely to produce a mesh that gives similar values to the structured Pointwise mesh, as it produces a hybrid mesh that is hex dominant as opposed to a purely tetrahedral unstructured mesh.



PW unstructured	PW structured	CF-mesh
-0.434	-0.401	-0.427

Figure 10 'Sensitivity of the solution to the meshing tool'

4.2.4 Proposed Ideas

As concluded in the earlier section, the over-solving of a grid can damage the mesh quality rather than improve it. Therefore, some methodologies to correct for this process are proposed. One such method would be to define the number of iterations that the solver is run for, based on the initial quality of the grid. However, it is believed that the over-solving process can be related to more than just the initial grid quality, such as the curvature of the geometry and the stability of the functions used.

A more appropriate method to account for this over solving process is to run the solver in iterative blocks of 20 or so

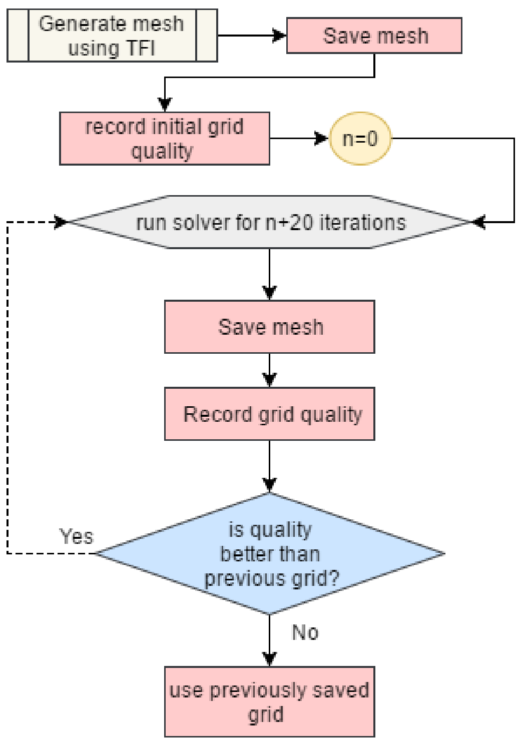

Generate mesh using TFI
Save mesh
record initial grid quality
n=0
run solver for n+20 iterations
Save mesh
Record grid quality
is quality better than previous grid?
Yes
No
use previously saved grid

Figure 21 'Methodology for preventing over-solving'

iterations at a time, with the grid quality of the previous step recorded and the quality of the current grid reported. As soon as the grid quality decreases from the previous step the process is stopped and reverts back to the mesh of the previous step. This process is detailed in figure 21 and is an outline of the proposed methodology.

5 Discussion and conclusions

5.1 Discussion

The structured meshing tool that has been implemented using Pointwise glyph scripting has been shown to produce good quality elements over a large range of varying geometries, using varying numbers of control points. It has been shown to give excellent results for 1 and 2 control point splines as well as good results for 3 and 4 control point splines. It does, however, struggle to mesh some of the more extreme geometries with high curvature and the values, recorded over many splines for 5 and 6 control points show a not insignificant number of meshes of unacceptable quality. Nevertheless, this is expected of a structured meshing tool. For some of the more complex geometries, even when meshed manually, it would be very challenging to produce structured grids of acceptable quality. For these geometries, unstructured grids would be able to achieve much higher quality meshes. Therefore, it is unrealistic to expect a purely structured tool to be completely robust for complex geometries. Although, with further refinement of the functions included within the elliptic solver, possibly with the use of an evolutionary algorithm, higher quality meshes may still be able to be produced.

In addition, within fluid dynamics it is considered that good design is usually characterised by smoothly-varying pressure distributions throughout the domain [4]. These distributions are almost certainly indicative of low-curvature geometries; therefore, it can be concluded that the most likely optima are not extreme geometries with very high curvature. Hence, most, if not all of the potential optima will be able to be meshed using a structured tool and the tools inability to mesh high-curvature geometries may be of little relevance in a fluid dynamics optimisation procedure. Counter to this argument, one advantage of an optimiser is it does not have pre-conceptions of what a ‘good design’ looks like and therefore may be able to uncover designs contrary to current beliefs and reasoning.

An implementation that makes use of the proposed iterative block technique to reduce over-solving of the mesh, as well as including the curvature of the spline to alter the solving function, could further increase the power of the tool.

Most other methods of meshing within an optimiser, outlined within the literature review, use a technique of mesh motion or distortion and it is said that re-meshing the geometry for each evaluation is too computationally expensive. However, the work done within this project presents an argument that entirely re-meshing the geometry can be computationally inexpensive and a viable option within an optimiser. The meshing procedure takes roughly 10 seconds to import the geometry, mesh it and assign the boundary conditions. Although the meshing of the diffuser section may be considered simple, as it is merely an extrusion of an initial domain. Many of the processes for meshing within Pointwise use a similar technique. Pointwise' use of a bottom-up meshing method allows the two-dimensional domains to be solved by the mesh smoothing algorithms, from which blocks are created that reflect the grid quality of the previous domain. Hence, solving the two-dimensional domains as opposed to an entire volume grid can drastically reduce the computational expense, therefore making this method a viable alternative.

Also since the exact distributions of points are known beforehand, the meshing algorithms do not have to decide the most appropriate distributions through the solving of algorithms. As the distributions and topology of the mesh are known beforehand. This is also one of the advantages of a structured mesh; a predefined topology.

Taking this work forward to be applied to a scenario in which the entire surface geometry can be altered, as has been investigated by Hutchings [45]. The entire surface mesh can be created using the structured TFI technique, from there the grid can be solved by the elliptic PDE's, this could then be extruded inwards in a similar manner to the method shown for this geometry to form an extended boundary layer style mesh. This forms the O part of an O-H grid, from which the interior H-grid can be simply created. Without the presence of the sharp heel this method could hopefully produce high quality grids; as are shown at the inlet of the sharp heel geometry shown in the methodology section.

Due to restraints on the size of the bounding box, control points mostly end up below the original spline geometry, this means the majority of the splines produced are concave and very few convex geometries have been tested. The meshing tool, and settings within it, has been tested on few convex geometries, therefore it's performance in these scenarios is not as well-known as for concave regions. It is however, known that concave boundaries are significantly more challenging to mesh due to the collision of grid points [15] and it could then be postulated that the meshing tool would produce reasonable results for convex boundaries also.

The use of a combination of meshing tools has been developed and allows for a structured meshing tool to be used within the optimiser without limiting the size of the design space to exclude high-curvature geometries. This therefore allows for less computationally expensive grids to be used for the majority of evaluations, thus reducing the computational expense of the problem as well as including the increased accuracy of good quality structured meshes. However, for this method to be used, the sensitivity between the meshes produced by the two separate tools must be as small as possible for an accurate optimiser. As yet, for this project, settings within the meshing tools have not been investigated to reduce this.

This meshing tool is also quite general and could be taken forward and applied to any two-dimensional cross section optimisation of an internal duct. With further time input and tweaking, it could be applied to a three-dimensional surface distortion optimiser. This highlights the power of the tool produced and of the Pointwise glyph scripting process.

5.2 Conclusion

A structured meshing tool has been presented that has been successfully automated within an optimisation procedure and has shown to produce meshes of good quality for geometries containing non-high curvature Catmull-Clark splines. The tool uses a transfinite interpolation method to create an initial two-dimensional domain which is then solved by Poisson’s elliptic partial differential equations to increase the cell quality, from which a three-dimensional volume mesh is created. There is still room for improvement here by introducing methods outlined in the results section. Frameworks have been created to monitor mesh quality and inhibit poor quality meshes misleading an optimiser. In addition, a novel combination of meshing tools has been developed to allow structured meshes to be used within a large design space that has potential for extreme geometries.

6 Project management

6.1 Project Management

This report presents one of eight individual reports that contribute towards a group objective; optimising the geometry of a hydraulic draft tube. Prior to undertaking any work, the risks of the project were assessed to determine what could be achieved within the time frame. Since the scope of the project was specific in that there was one main result; an optimised draft tube geometry, the project was initially broken up into three highly interconnected work packages with more than one individual in each. These were subsequently broken down into smaller work packages that were assigned to each individual. Since each of the larger work packages were highly dependent on one another, the management of the project was key and mitigating strategies were identified to reduce the impact of a work package not being achieved. For example, three meshing strategies produced to introduce a redundancy. To monitor the progress of each work package weekly meetings with our supervisor were completed with meeting minutes drawn up as well as a chair person to ensure all relevant topics were covered.

Potential risks in terms of producing a final optimum design were considered prior to creating the group and individual goals within the project. It was decided to begin with the simple case of optimising just the diffuser section of the draft tube, as well as not including the more complex swirling flow. In order to reduce duplication of work this author was assigned the task of learning programming within Python, in order to automate the meshing side of the project and produce coding within the CFD work package.

Due to mitigating circumstances of other group members', experimental data was heavily delayed until very close to the project deadline. This data is essential for performing numerical simulations, as it is one of the only ways to assess the accuracy. In order to mitigate this, this project was decided to be purely based on the quality metrics of a mesh as opposed to the effect that this mesh quality may have on the accuracy of the solution.

In terms of individual project management, a Gantt chart was initially drawn up. This included managing the time taken to learn new skills necessary for the project. Since this author had no previous programming experience, time had to be allotted to learn the overall principles and the specifics of coding within Python and to a lesser degree Tcl. Time also had to be taken to learn the process of mesh generation within Pointwise, which is a far more involved process than other techniques used previously by this author.

6.2 Sustainability

In terms of sustainability, the direct implications of this individual project were very little, since all the work performed was done computationally. However, the overall result of the project could have significant impacts upon the sustainability of energy generation. Since, an improved draft tube design improves the efficiency of a reaction turbine it can also increase the amount of energy generated within a hydroelectric power plant. This can help to reduce the dependency on fossil fuels and increase the sustainability of hydroelectric power generation.

7 Contribution to group functioning

In terms of general group functioning, the work done here provides an automatic meshing tool able to mesh a range of geometries that presented as an induvial function, to allow it to be integrated into the optimiser produced by Ng [28]. An automated tool that produces acceptable quality meshes is essential within CFD optimisation and has been used to generate optima shown in [46]. The interfacing of the meshing tools with the optimiser was also performed in conjunction with Ng [28] and Hardy [42]. Work was also done to automate the meshing process of cfMesh used by Hardy [42] as well as the production of the CAD model which was used across all CFD process and to print experimental models.

8 References

[1] F. Guibault, Y. Zhang, J. Dompierre and T. C. Vu, “Robust and Automatic CAD-based Structured Mesh Generation for Hydraulic Turbine Component Optimization,” in *23rd IAHR Symposium on Hydraulic Machinery and Systems,* Yokohama, 2006.

[2] J. C. Vassberg, N. A. Harrison and D. L. Roman, “A Systematic Study on tthe impact of Dimesnionality for a Two-Dimensional Aerodynamic Optimisation Model Problem,” in *29th AIAA Applied Aerodynamics Conference* , Honolulu, Hawaii, 2011.

[3] G. Carrier, D. Destarac, A. Dumont, M. Meheut, I. S. E. Din, J. Peter, S. B. Khelil, J. Brezillon and M. Pestana, “Gradient-Based Aerodynamic Optimization with the elsA Software,” in *52nd Aerospace Sciences Meeting,* National Harbor, Maryland, 2014.

[4] J. C. Vassberg and A. Jameson, “Influence of Shape Parameterization on Aerodynamic Shape Optimization,” in *Von Karman Institute Lecture-III,* Brussels, Belgium, 2014.

[5] B. Marjavaara and T. S. Lundstrom, “Redesign of a sharp heel draft tube by a validated CFD-optimisation,” *International Journal for Numerical Methods in Fluids ,* vol. 50, no. 8, pp. 911-924, 2006.

[6] J. McNabb, C. Devals, S. A. Kyriacou, N. Murry and B. F. Mullins, “CFD based draft tube hydraulic design optimization,” in *IOP Conference Series: Earth and Environmental Science,* Volume 22, 2014.

[7] R. Eisenger and A. Ruprecht, “Automatic shape optimization of hydro turbine components based on CFD,” *TASK QUARTERLY ,* vol. 1, no. 6, pp. 101-111, 2002.

[8] J. Rhoads, “Effects of grid quality on solution accuracy,” in *9th OpenFOAM Workshop,* Zagreb, 2014.

[9] S. Daniels, A. Rahat, G. Tabor, J. Fieldsend and R. Everson, “A review of shape distortion methods available in the OpenFOAM framework for automated design optimisation,” 2017.

[10] B. Fabritius and G. Tabor, “Improving the quality of finite volume meshes through genetic optimisation,” *Engineering with Computers,* vol. 32, no. 3, pp. 425-440, 2016.

[11] “Automate Your CFD Meshing with Scripting,” Pointwise, 2011. [Online]. Available: http://www.pointwise.com/pw/script.shtml.

[12] M. Grigoriev, T. Carrigan, D. Garlisch and J. Hitt, “Higher Order Finite Element Meshes for Centrifugal Impeller Blade Analyses Using Pointwise,” in *ASME Turbo Expo: Turbine Technical Conference and Exposition,* Düsseldorf, Germany, 2014.

[13] L. E. Eriksson, “Generation of boundary-conforming grids around wing-body configurations using transfinite interpolation,” *American Institute of Aeronautics and Astronautics,* vol. 20, no. 10, pp. 1313-1320, 1982.

[14] S. P. Spekreijse, B. B. Prananta and J. C. Kok, “A simple, robust and fast algorithm to compute deformations of multi-block structured grids,” *NATIONAAL LUCHT EN RUIMTEVAARTLABORATORIUM-PUBLICATIONS-NLR,* 2002.

[15] C. B. Allen, “Towards automatic structured multiblock mesh generation using improved transfinite interpolation,” *International Journal for Numerical Methods in Engineering* , vol. 74, no. 5, pp. 697-733, 2008.

[16] W. M. Chan, “Enhancements of a three-dimensional hyperbolic grid generation scheme,” *Applied Mathematics and Computation,* vol. 51, no. 2-3, pp. 181-205, 1991.

[17] S. Nakamure and M. Suzuki, “Noniterative three-dimensional grid generation using a parabolic-hyperbolic hybrid scheme,” in *25th AIAA Aerospace Sciences Meetin,* Reno, 1987.

[18] P. D. Thomas, “Composite Three-Dimensional Grids Generated by Elliptic Systems,” *AIAA Journal,* vol. 20, no. 9, pp. 1195-1202, 1982.

[19] W. Schwarz, “Elliptic grid generation system for three-dimensional configurations using Poisson’s equations.,” in *Proceedings of 1st International Conference on Numerical Grid Generation in CFD,* Swansea, 1986.

[20] S. S. Dhanabalan and H. M. Tsai, “An Algebraic Elliptic Grid Generator with Grid Spacing Control,” in *44th AIAA Aerospace Sciences Meeting and Exhibit,* Reno, 2006.

[21] T. E. Tezduyar, S. Sathe, M. Schwaab and B. S. Conklin, “Arterial fluid mechanics modeling with the stabilized space time fluid structure interaction technique,” *Numerical Methods in Fluids,* vol. 57, no. 5, pp. 601-629, 2008.

[22] K. Stein, T. E. Tezduyar and R. Benney, “Automatic mesh update with the solid-extension mesh moving technique,” *Computer Methods in Applied Mechanics and Engineering,* vol. 193, no. 21-22, pp. 2019-2032, 2004.

[23] H. Jasak, “Dynamic Mesh Handling in OpenFOAM,” in *47th AIAA Aerospace Sciences Meeting* , Zagreb, 2009.

[24] J. Hrvoje, R. Damir and T. Željko, “Design and implementation of Immersed Boundary Method with discrete forcing approach for boundary conditions,” in *6th European Congress on Computational Fluid Dynamics,* Barcelona, 2014.

[25] T. Carrigan and S. Barr, “Big Wave Surfboard Optimization Using Pointwise & CRUNCH CFD,” Poinwise, 2017. [Online]. Available: http://www.pointwise.com/webinar/2017-04/Big-Wave-Surfboard-Optimization-Pointwise-CRUNCH-CFD.shtml.

[26] Y. Jin, M. Olhofer and B. Sendhoff, “On evolutionary optimization with approximate fitness functions,” in *Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation,* Las Vegas, 2000.

[27] R. P. Liem, C. A. Mader and J. Martins, “Surrogate models and mixtures of experts in aerodynamic performance prediction for aircraft mission analysis,” *Aerospace Science and Technology,* vol. 43, p. 126–151, 2015.

[28] C. Ng, “Implementation of Bayesian Optimiser on CFD Modelled Draft Tube,” Exeter, 2017.

[29] G. D. Santis, M. D. Beule, P. Segers, P. Verdonck and B. Verhegghe, “Patient-specific computational haemodynamics: generation of structured and conformal hexahedral meshes from triangulated surfaces of vascular bifurcations,” *Computer Methods in Biomechanics and Biomedical Engineering* , vol. 14, no. 9, pp. 792-802, 2011.

[30] Y. Zang, R. L. Street and J. R. Koseff, “A Non-staggered Grid, Fractional Step Method for Time-Dependent Incompressible Navier-Stokes Equations in Curvilinear Coordinates,” *Journal of Computational Physics,* vol. 114, no. 1, pp. 18-33, 1994.

[31] “OpenFOAM User Guide: 4.4 Numerical schemes,” CFD-direct, 2015. [Online]. Available: https://cfd.direct/openfoam/user-guide/fvschemes/.

[32] M. Goelke, “Element Quality and Checks,” Altair University, 2014. [Online]. Available: http://www.altairuniversity.com/modeling/element-quality/.

[33] “Quality Meshes for Converged and Accurate CFD,” Pointwise, 2015. [Online]. Available: http://www.pointwise.com/pw/quality.shtml.

[34] “How Meshing is Better with Pointwise:,” CFD Technologies, [Online]. Available: http://www.cfd-technologies.co.uk/Pointwise.htm.

[35] “Pointwise User Manual,” 2016. [Online]. Available: http://www.pointwise.com.

[36] J. F. Thompson, F. C. Thames and C. W. Mastin, “Boundary-fitted curvlinear coordinate systems for solution of partial differential differencial eqautions on fields containing any number of arbitrary two-dimensional bodies,” NASA, Washington, 1977.

[37] R. L. Sorenson and J. L. Steger, “Numerical Generation of two-dimensional grids by the use of poisson equations with grid control,” NASA. Langley Research Center Numerical Grid Generation Tech, United States, 1980.

[38] “A Fast Method For The Elliptic Generation of Three-Dimensional Grids With Full Boundary Control,” *Numerical Grid Generation in Computational Fluid Mechanics,* pp. 137-146, 1988.

[39] P. D. Thomas and J. F. Middlecoff, “Direct Control of the Grid Point Distribution in Meshes Generated by Elliptic Equations,” *AIAA Journal,* vol. 18, pp. 652-656, 1979.

[40] P. Burns, “The development of an automatic numerical draft tube model,” Exeter, 2017.

[41] F. M. White, “Fluid Mechanics 5th edition,” Boston, McGraw-Hill, 2003, p. 467.

[42] S. Hardy, “Automatic Mesh Generation for Shape Optimisation of a Draft Tube Using cfMesh,” Exeter, 2017.

[43] J. Angus, “Experimental Investigation of Flow Inside Draft Tubes with Varied Diffuser Geometry,” Exeter, 2017.

[44] J. Rhoads, “How Do You Define a Good Grid?,” Pointwise, November 2014. [Online]. Available: http://www.pointwise.com/theconnector/November-2014/What-Defines-a-Good-Grid.shtml.

[45] S. Hutchings, “Surface Representation for Optimisation using Catmull-Clark Subdivision,” Exeter, 2017.

[46] J. Gowans, S. Hardy, P. Burns, R. Gilbert, C. Ng, J. Angus, S. Hutchings and T. Dye, “G2 - An investigation into the Optimisation of a Draft Tube,” Exeter, 2017.

[47] S. Hutchings, “Surface Representation for Optimisation using Catmull-Clark Subdivision,” Exeter, 2017.

9 Appendix

Please follow the link to access the following:
https://universityofexeteruk-my.sharepoint.com/personal/jg432_exeter_ac_uk/_layouts/15/guestaccess.aspx?folderid=0012740bed4ec4e4cb370fcaef26a9cbb&authkey=AQ8mtTpkOzR-iqmnb5kqs70&e=5%3Af52f5b2ceb294ccda9d8026e045ee8dc

A: lines 103-116 of meshingWithinOptimiser.py

B: structuredPointwiseGlyphScript.glf

C: gradientFunction.py

D: lines 130-184 of meshingWithinOptimiser.py

E: lines 88-126 of meshingWithinOptimiser.py

F: evaluateQualitySettings.py

G: lines 220-260 of meshingWithinOptimiser.py